

TECNOLÓGICO DE ESTUDIOS SUPERIORES DE COACALCO

“Sistema de información web con PHP y Firebase para la
gestión y control de inventario de pinturas para CC
Camaleón Color S.A. de C.V.”

Informe Técnico de Residencias Profesionales

PARA OBTENER EL TÍTULO EN
INGENIERÍA EN SISTEMAS COMPUTACIONALES.

Especialidad

Ingeniería de Software

Presenta(n):

Cortez Mercado Marco Alexis.

Hernández de la Cruz Diana Ivonne.

Coacalco de Berriozábal, septiembre/2024

AGRADECIMIENTOS

Presente lector, es un honor y un privilegio dedicar este espacio para expresar los más sinceros agradecimientos inicialmente a esta comunidad educativa el Tecnológico de Estudios Superiores de Coacalco por el crecimiento académico que no habría sido posible sin el apoyo de la docencia a cargo, institución la cual no solo ha sido un lugar de aprendizaje, sino también un hogar donde se fomenta la excelencia, la amistad y el desarrollo integral. Cada uno de los docentes deja un aprendizaje en la experiencia educativa. Al Dr. José Roberto Pérez Torres, por su invaluable contribución a nuestro desarrollo académico, gracias a su guía y por compartir sus conocimientos en las materias y ser parte de este proyecto sin duda un mentor comprometido con el éxito.

Por otro lado, nos es un honor dirigirnos a la empresa CC Camaleón Color S.A de C.V. con agradecimiento por la oportunidad y la confianza para la realización de este proyecto, por permitir innovar en el área asignada principalmente al Lic. Mauro Reyes Ortiz, Director general de esta empresa agradecemos la disposición constante y la de todo su equipo de profesionales en el área designada, la cual es Calidad y Almacén, ha sido una experiencia pilar para el impulso de nuestro desarrollo académico y laboral.

Igual de importante aprovechamos este espacio para expresar nuestros agradecimientos a nuestra querida familia por el apoyo durante este proceso de desarrollo académico, gracias a todos los que formaron parte, amigos y familia por creer en este proyecto.

RESUMEN

El presente proyecto enfoca en el desarrollo de un sistema de información web de inventario utilizando PHP y Firebase con el objetivo de optimizar la gestión completa de los productos y herramientas en CC Camaleón Color S.A. de C.V.

El sistema abarca diferentes funciones que van desde el control de inventario hasta el seguimiento de salidas, pasando por la gestión detallada de proveedores y usuarios. El sistema se enfoca principalmente en mejorar el proceso de almacenado de productos y salidas de estos, para lo cual se integra un acceso controlado de usuarios mediante una plataforma web vinculada a una base de datos en la nube, almacenando los productos, usuarios, proveedores, etc.

En el núcleo del sistema, PHP será empleado para la programación del lado del servidor, asegurando una implementación robusta y eficaz de la lógica empresarial. La elección de Firebase como plataforma de base de datos proporcionará un entorno seguro, escalable y en tiempo real para el almacenamiento de datos, facilitando así la sincronización de información crítica en todo el sistema.

A su vez el sistema tiene la posibilidad de búsqueda, eliminar, actualizar e ingresar información específica de cada uno de los procesos con lo que cuenta el sistema, las diferentes áreas con las que cuenta el sistema brindan un mayor control y flexibilidad sobre el uso e ingreso de información, por lo que el proyecto busca gestionar y agilizar el manejo de inventario mejorando las salidas de productos para CC Camaleón Color S.A de C.V.

ÍNDICE DE CONTENIDOS

AGRADECIMIENTOS	2
RESUMEN	3
CAPÍTULO I. GENERALIDADES DEL PROYECTO	25
1. INTRODUCCIÓN.....	26
1.2. DESCRIPCIÓN DE LA EMPRESA	27
- Misión:	27
- Visión:	27
- Área directa de trabajo: Calidad y almacén	27
1.2.1. ORGANIGRAMA	28
1.3. PLANTEAMIENTO DEL PROBLEMA	29
1.4. JUSTIFICACIÓN	30
1.5. OBJETIVOS.....	31
1.5.1. OBJETIVO GENERAL	31
1.5.2. OBJETIVOS ESPECÍFICOS	31
CAPÍTULO II. MARCO TEÓRICO.....	32
2. Escenario teórico de investigación	33
2.1. Ingeniería de software	33
2.1.1. Arquitectura de software.....	33
2.2. Ingeniería de requisitos	33
2.2.1. Definición de requisito	34
2.2.2. Tipos de requisitos	34
2.3. Diagramas UML	35
2.3.1. Diagramas de casos de uso en UML	35
2.3.2. Elementos de diagrama de caso de uso	36
2.4. Internet	36
2.4.1. Protocolos	36
2.4.2. Modelo Cliente- Servidor.....	37
2.4.3. Servidor.....	37
2.4.4. Cliente.....	37
2.4.5. Programa Servidor	38
2.4.6. Programa cliente	38
2.5. Página web.....	38
2.5.1. Tipos de páginas web	38
2.5.2. Página web de entrada	39

2.5.3.	Página de bienvenida	39
2.5.4.	Página web principal	39
2.5.5.	Diseño de páginas web	39
2.6.	Lenguaje de programación	40
2.6.1.	JavaScript	40
2.6.2.	HTML	40
2.6.3.	CSS.....	41
2.6.4.	PHP	41
2.6.5.	XHTML	42
2.7.	Firebase	42
2.7.1.	Mobile Backend as a Service (MBaaS)	43
2.7.2.	Los servicios de Firebase	43
2.7.3.	Herramientas de desarrollo.....	44
2.7.4.	Herramientas de análisis e interacción de usuarios con Firebase.....	45
2.7.5.	Herramientas de comprobación.....	45
2.8.	Base de datos.....	45
2.8.1.	Antecedentes de las bases de datos	46
2.8.2.	Origen de las bases de datos	46
2.8.3.	El Data minig en la actualidad	47
2.9.	Bases de datos no relacionales	47
2.9.1.	Definición de NoSQL	47
2.9.2.	¿Cuándo elegir una base de datos no relacional?	48
2.9.3.	Ventajas de una base de datos no relacional.....	48
2.10.	Características de las bases de datos	50
2.10.1.	Integridad.....	50
2.10.2.	Redundancia.....	51
2.11.	Procesos operativos	51
2.11.1.	Inventario.....	51
2.11.2.	Stock	53
2.11.3.	Control de inventarios.....	53
2.12.	MARCO METODOLÓGICO	53
2.12.1.	ETAPAS DE DESARROLLO DE LA PROGRAMACIÓN EXTREMA	55
CAPÍTULO III. DESARROLLO		58
3.	Descripción General	59
3.1.	Perspectiva del producto	59
3.2.	Funciones del producto.....	59
3.3.	Características del usuario.....	61

3.4.	Restricciones	62
3.5.	Suposiciones y dependencias.....	63
3.6.	Requisitos futuros	65
3.7.	Interfaces externas	65
3.8.	Funciones	70
3.8.1.	Requisitos funcionales	70
	Los requisitos funcionales identificados en las reuniones con la empresa se presentan a continuación.	70
3.8.1.1.	Requisitos no funcionales.....	73
3.9.	Requisitos específicos	74
3.9.1.	Caso de uso general.....	75
3.10.	DESARROLLO DE CODIGO	120
3.10.1.	Creación de Base de datos.....	120
CAPÍTULO IV: RESULTADOS		277
4.	Resultados	278
4.1.	Prueba de Funcionalidad.....	278
4.2.	Objetivo de la prueba	278
4.3.	Pruebas de seguridad	283
4.3.1.	Prueba de penetración – OWASP.....	283
4.3.2.	Prueba de Nmap.....	286
CONCLUSIONES		288
COMPETENCIAS DESARROLLADAS		289
ANEXOS		293
MANUAL DE USUARIO		294
	Introducción.....	294
	Alcances	294
	Guía de uso	296
	Sección de solución de problemas.....	310
EVIDENCIAS DE IMPLEMENTACIÓN		312

ÍNDICE DE FIGURAS

Figura. 1 Organigrama de CC Camaleón Color. Fuente. Elaboración propia.	28
Figura. 2. Mapa de ubicación de CC Camaleón Color. Fuente Google Maps.....	28
Figura. 3. Graphical NoSQL Database Cloud Service – Real Time Throughput. Fuente. Oracle 2023.	49
Figura. 4 Diagrama de caso de uso general del sistema. Creatly Elaboración propia.....	75
Figura. 5 Caso de uso específico autenticar usuario. Fuente. Elaboración propia.	76

Figura. 6 Pantalla de inicio de sesión. Fuente. Elaboración propia.	77
Figura. 7 Pantalla de ingreso de datos. Fuente. Elaboración propia	78
Figura. 8 Ingreso al sistema. Fuente. Elaboración propia.	78
Figura. 9 Menú principal del sistema de inventario. Fuente. Elaboración propia.....	78
Figura. 10 Caso de uso específico de registro de nuevo usuario. Fuente. Creately, elaboración propia.....	79
Figura. 11 Ingreso al menú principal. Fuente. Elaboración propia.	80
Figura. 12 Lista desplegable de usuarios. Fuente. Elaboración propia.	81
Figura. 13 Formulario de registro de nuevo usuario. Fuente. Elaboración propia.....	81
Figura. 14 Ejemplo de llenado incorrecto. Fuente. Elaboración propia.	82
Figura. 15 Mensaje de alerta de datos faltantes. Fuente. Elaboración propia.....	82
Figura. 16 Mensaje de agregar usuario exitoso. Fuente. Elaboración propia.	82
Figura. 17 Caso de uso editar datos. Fuente. Creately. Elaboración propia.	83
Figura. 18 Entrar a la sección de editar en lista. Fuente. Elaboración propia.....	84
Figura. 19 Lista de usuarios. Fuente. Elaboración propia.	85
Figura. 20 Icono de editar usuario. Fuente. Elaboración propia.	85
Figura. 21 Edición de administrador. Fuente. Elaboración propia.	85
Figura. 22 Cambio de correo. Fuente. Elaboración propia.....	85
Figura. 23 Caso de uso específico eliminar cuenta de usuario. Fuente. Creately, elaboración propia.....	86
Figura. 24 Ingreso a lista de usuarios. Fuente. Elaboración propia.	87
Figura. 25 Lista de usuarios. Fuente. Elaboración propia.	88
Figura. 26 Icono de eliminar. Fuente. Elaboración propia.	88
Figura. 27 Caso de uso Agregar producto. Fuente. Elaboración propia.....	89
Figura. 28 Caso de uso editar usuario. Fuente. Creately, elaboración propia.....	90
Figura. 29 Menú principal. Fuente. Elaboración propia.....	92
Figura. 30 Lista para editar. Fuente. Elaboración propia.	92
Figura. 31 formulario para editar producto. Fuente. Elaboración propia.	93
Figura. 32 Cambio de nombre de producto. Elaboración propia.	93
Figura. 33 Lista actualizada de productos. Fuente. Elaboración propia.....	93
Figura. 34 Caso de uso específico de eliminar producto. Fuente. Creately, elaboración propia.	94
Figura. 35 Módulo de lista de productos. Fuente. Elaboración propia.....	95
Figura. 36 Confirmación de eliminación de producto. Fuente. Elaboración propia.	95
Figura. 37 Lista de productos actualizada. Fuente. Elaboración propia.....	96
Figura. 38 Caso específico de agregar nuevas compras. Fuente Creately Elaboración propia.....	96
Figura. 39 Página principal y módulo de compras. Fuente. Elaboración propia.	98

Figura. 40 Módulo de nueva compra. Fuente. Elaboración propia.	98
Figura. 41 Agregar nueva compra. Fuente. Elaboración propia.	98
Figura. 42 Compra agregada a la lista de compras. Fuente. Elaboración propia,	99
Figura. 43 Actualización en inventario de la nueva compra. Fuente. Elaboración propia.	99
Figura. 44 Ejemplo de datos faltantes. Fuentes. Elaboración propia.	99
Figura. 45 Ejemplo de advertencia de campo faltante. Fuente. Elaboración propia.	99
Figura. 46 Caso de uso específico de consultar productos. Fuente. Elaboración propia.	100
Figura. 47 Pantalla principal para ingreso a inventario. Fuente. Elaboración Propia.	102
Figura. 48 Lista de inventario. Fuente. Elaboración propia.	102
Figura. 49 Ejemplo de búsqueda por proyecto. Fuente. Elaboración propia.	102
Figura. 50 Caso de uso específico Imprime/ genera reporte. Fuente. Elaboración propia.	103
Figura. 51 Módulo de inventario. Fuente. Elaboración propia.	104
Figura. 52 Botón de exportar datos. Fuente. Elaboración propia.	104
Figura. 53 Descarga de archivo .xlsx. Fuente. Elaboración propia.	105
Figura. 54 Archivo en Excel. Fuente. Elaboración propia.	105
Figura. 55 Caso de uso específico de registrar proveedores. Fuente. Elaboración propia.	105
Figura. 56 Selección de administración para agregar proveedores. Fuente. Elaboración propia.	107
Figura. 57 Módulo y formulario de agregar proveedor. Fuente. Elaboración propia.	107
Figura. 58 Agregar datos de proveedor nuevo. Fuente. Elaboración propia.	107
Figura. 59 actualización de lista de proveedores. Fuente Elaboración propia.	108
Figura. 60 Caso de uso específico de Editar y eliminar proveedor. Fuente. Elaboración propia.	108
Figura. 61 Lista de proveedores. Fuente. Elaboración propia.	110
Figura. 62 Icono para editar proveedor.	110
Figura. 63 Editar datos de proveedor. Fuente. Elaboración propia.	110
Figura. 64 Actualización de datos de proveedores. Fuente. Elaboración propia.	111
Figura. 65 Selección de borrar registro de proveedor. Fuente. Elaboración propia.	111
Figura. 66 Confirmación de eliminación. Fuente. Elaboración propia.	111
Figura. 67 Lista actualizada de datos eliminados de proveedores. Fuente. Elaboración propia.	111
Figura. 68 Caso de uso de registro de salidas. Fuente. Creately, Elaboración propia.	112
Figura. 69 Seleccionar icono para realizar salida de producto. Fuente. Elaboración propia.	113
Figura. 70 Dar salida al producto que selecciono. Fuente. Elaboración propia.	114
Figura. 71 Interfaz de registro de salidas de productos. Fuente. Elaboración propia.	114
Figura. 72 Ingreso de cantidades negativas, Fuente. Elaboración propia.	115
Figura. 73 Advertencia por el ingreso de cantidades negativas, Fuente. Elaboración propia.	116

Figura. 74 Advertencia de campo vacío. Fuente. Elaboración propia	116
Figura. 75 Ingreso de una operación correcta, Fuente: Elaboración propia	116
Figura. 76 Registros de tabla administradores, Fuente. Elaboración propia	117
Figura. 77 Datos modificados de tabla administradores, Fuente. Elaboración propia	117
Figura. 78 Advertencia por ingreso erróneo, Fuente. Elaboración propia.....	118
Figura. 79 Ejemplo de formato invalido, Fuente. Elaboración propia.	118
Figura. 80 Vista del sistema desde un ordenador, Fuente. Elaboración propia	119
Figura. 81 Vista desde dispositivo móvil, Fuente. Elaboración propia.....	119
Figura. 82 Comienzo a Firebase. Fuente. Fuente. Elaboración propia.....	120
Figura. 83 Acceder a la cuenta de Google para crear proyecto en Firebase.....	120
Figura. 84 Nombre del nuevo proyecto Fuente. Elaboración propia,	121
Figura. 85 Interfaz de configuración Fuente Elaboración propia.	121
Figura. 86 URL de conexión a Firebase. Fuente. Elaboración propia.	122
Figura. 87 Estructura de conexión a la Base de Datos, Fuente. Elaboración propia.....	122
Figura. 88 Interfaz del editor de texto Sublime Text, Fuente. Elaboración propia.	123
Figura. 89 Cuerpo de la carpeta Proyecto Camaleón. Fuente. Elaboración propia.....	123
Figura. 90 Estructura de lenguaje HTML e Idioma del archivo index.html. Fuente. Elaboración propia.	124
Figura. 91 Ingreso de título y enlaces de funciones del archivo index.html. Fuente. Elaboración propia.	124
Figura. 92 Cuerpo de la carpeta img. Fuente. Elaboración propia.....	125
Figura. 93 Configuración del archivo estilos.css. Fuente. Elaboración propia.....	125
Figura. 94 Creación de contenedores de autenticación. Fuente. Elaboración propia.....	126
Figura. 95 Creación de formulario de autenticación (administradores). Fuente. Elaboración propia.....	127
Figura. 96 Creación de formulario de autenticación para personal. Fuente. Elaboración propia.	127
Figura. 97 Ingreso de animaciones mediante el archivo script.js. Fuente. Elaboración propia.	128
Figura. 98 Configuración de las animaciones para los formularios de administrador y personal. Fuente. Elaboración propia.....	128
Figura. 99 Configuración para la visualización de la contraseña en formularios. Fuente. Elaboración propia.	129
Figura. 100 Ingreso de archivos de conexión a Firebase del formulario administrador. Fuente. Elaboración propia.....	130
Figura. 101 Evaluación de contenido de campos del formulario administrador. Fuente. Elaboración propia.	130
Figura. 102 Configuración de ingreso de datos a la tabla usuarios. Fuente. Elaboración propia.	131
Figura. 103 Ingreso de archivos de conexión a Firebase del formulario personal. Fuente. Elaboración	

propia.	131
Figura. 104 Configuración de ingreso de datos a la tabla personal. Fuente. Elaboración propia.	132
Figura. 105 Configuración de lenguaje y tiempo de inactividad del archivo home.php. Fuente. Elaboración propia.	132
Figura. 106 Estructura de lenguaje HTML, enlaces e Idioma del archivo home.php. Fuente. Elaboración propia.	133
Figura. 107 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.	133
Figura. 108 Codificación de la barra de navegación lateral pt1. Fuente. Elaboración propia.	134
Figura. 109 Codificación de la barra de navegación lateral pt2. Fuente. Elaboración propia.	135
Figura. 110 Codificación de la barra de navegación lateral pt3. Fuente. Elaboración propia.	135
Figura. 111 Codificación de la barra de navegación lateral pt4. Fuente. Elaboración propia.	136
Figura. 112 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	136
Figura. 113 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	137
Figura. 114 Creación de azulejos de visualización de registros. Fuente. Elaboración propia.	137
Figura. 115 Codificación del conteo de registros para el uso de azulejos, Fuente. Elaboración propia.	138
Figura. 116 Configuración de lenguaje y tiempo de inactividad del archivo admin.php. Fuente. Elaboración propia.	139
Figura. 117 Ingreso de Título y enlaces de funciones del archivo admin.php, Fuente. Elaboración propia.	140
Figura. 118 Codificación del nombre y avatar del usuario, Fuente. Elaboración propia.	140
Figura. 119 Codificación de la barra de navegación lateral pt1, Fuente. Elaboración propia.	141
Figura. 120 Codificación de la barra de navegación lateral pt2, Fuente. Elaboración propia.	141
Figura. 121 Codificación de la barra de navegación lateral pt3, Fuente. Elaboración propia.	142
Figura. 122 Codificación de la barra de navegación lateral pt4, Fuente. Elaboración propia.	142
Figura. 123 Creación de los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.	143
Figura. 124 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente, Elaboración propia.	144
Figura. 125 Codificación de enlaces de edición y eliminación, Fuente. Elaboración propia.	144
Figura. 126 Definición de título del formulario para administradores, Fuente. Elaboración propia.	145
Figura. 127 Creación de los campos para el formulario de administradores, Fuente. Elaboración propia.	145
Figura. 128 Codificación de la estructura de los campos para el formulario, Fuente. Elaboración propia.	145
Figura. 129 Creación del botón agregar administrador, Fuente. Elaboración propia.	146

Figura. 130 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente, Elaboración propia.	146
Figura. 131 Codificación para el llamado de tablas en Firebase, Fuente. Elaboración propia.	147
Figura. 132 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia.	147
Figura. 133 Codificación de ingreso de datos a la tabla usuarios, Fuente. Elaboración propia.	148
Figura. 134 Creación de la tabla de registros para administradores, Fuente. Elaboración propia.	148
Figura. 135 Creación y codificación para la visualización de registros en la tabla de administradores, Fuente. Elaboración propia.	149
Figura. 136 Creación y codificación del botón de conversión a formato Excel, Fuente. Elaboración propia.	149
Figura. 137 Creación y configuración del archivo editar.php, Fuente. Elaboración propia.	150
Figura. 138 Codificación y llamado de registros en los campos del formulario administrador, Fuente. Elaboración propia.	150
Figura. 139 Botón editar administrador, Fuente. Elaboración propia.	151
Figura. 140 Codificación de tiempo de inactividad del archivo action_editar.php. Fuente, Elaboración propia.	151
Figura. 141 Conexión a la base de datos y verificación para datos repetidos de los campos, Fuente. Elaboración propia.	152
Figura. 142 Codificación de ingreso de datos actualizados a la tabla usuarios, Fuente. Elaboración propia.	152
Figura. 143 Creación de la tabla de registros actualizados para administradores, Fuente. Elaboración propia.	153
Figura. 144 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente, Elaboración propia.	153
Figura. 145 Codificación y eliminación de registros de la tabla usuario, Fuente. Elaboración propia.	154
Figura. 146 Codificación de enlaces de edición y eliminación, Fuente. Elaboración propia.	154
Figura. 147 Contenido de la carpeta assets, Fuente. Elaboración propia.	155
Figura. 148 Contenido de la carpeta CSS, Fuente. Elaboración propia.	155
Figura. 149 Contenido de la carpeta fonts, Fuente. Elaboración propia.	156
Figura. 150 Contenido de la carpeta icons, Fuente. Elaboración propia.	156
Figura. 151 Contenido de la carpeta img., Fuente. Elaboración propia.	156
Figura. 152 Contenido de la carpeta js, Fuente. Elaboración propia.	157
Figura. 153 Configuración de lenguaje y tiempo de inactividad del archivo categorias.php, Fuente.	

Elaboración propia.....	157
Figura. 154 Ingreso de Título y enlaces de funciones del archivo categorías.php, Fuente. Elaboración propia.....	158
Figura. 155 Codificación del nombre y avatar del usuario, Fuente. Elaboración propia.....	159
Figura. 156 Codificación de la barra de navegación lateral pt1, Fuente. Elaboración propia.....	159
Figura. 157 Codificación de la barra de navegación lateral pt2, Fuente. Elaboración propia.....	160
Figura. 158 Codificación de la barra de navegación lateral pt3, Fuente. Elaboración propia.....	160
Figura. 159 Botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.....	161
Figura. 160 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.....	161
Figura. 161 Creación de contenedores del formulario categorías, Fuente. Elaboración propia.....	162
Figura. 162 Creación de título del formulario e indicaciones, Fuente. Elaboración propia.....	162
Figura. 163 Creación de título del formulario e indicaciones, Fuente. Elaboración propia.....	163
Figura. 164 Creación de campos para el formulario de categorías, Fuente, Elaboración propia.....	163
Figura. 165 Codificación de la estructura de los campos para el formulario, Fuente. Elaboración propia.....	164
Figura. 166 Creación del botón agregar categoría, Fuente. Elaboración propia.....	164
Figura. 167 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente, Elaboración propia.....	164
Figura. 168 Codificación para el llamado de tablas en Firebase, Fuente. Elaboración propia.....	165
Figura. 169 Codificación de ingreso de datos a la tabla categorías, Fuente. Elaboración propia.....	165
Figura. 170 Almacena en categorías. Fuente. Elaboración propia.....	165
Figura. 171 Creación de la tabla de registros para las categorías, Fuente. Elaboración propia.....	166
Figura. 172 Visualización de registro. Fuente. Elaboración propia.....	166
Figura. 173 Configuración y llamado a la base de datos de categorías, Fuente. Elaboración propia.....	167
Figura. 174 Codificación de registros en el formulario de categorías, Fuente. Elaboración propia.....	167
Figura. 175 Creación del botón agregar categoría, Fuente. Elaboración propia.....	168
Figura. 176 Codificación de tiempo de inactividad del archivo action_editar.php. Fuente, Elaboración propia.....	168
Figura. 177 Conexión a la base de datos y verificación de campos, Fuente. Elaboración propia.....	169
Figura. 178 Codificación de actualización de registros de la tabla categorías, Fuente. Elaboración propia.....	169
Figura. 179 Creación de la tabla de registros actualizados para categorías, Fuente. Elaboración propia.....	170
Figura. 180 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente, Elaboración propia.....	170

Figura. 181 Codificación y eliminación de registros de la tabla categorías, Fuente. Elaboración propia.	171
Figura. 182 Codificación de enlaces de edición y eliminación, Fuente. Elaboración propia.	171
Figura. 183 Configuración de lenguaje y tiempo de inactividad del archivo compras.php, Fuente. Elaboración propia.	172
Figura. 184 Estructura de lenguaje HTML e Idioma del archivo compras.php. Fuente. Elaboración propia.	173
Figura. 185 Configuración de la barra de navegación lateral pt1. Fuente. Elaboración propia.	174
Figura. 186 Configuración de la barra de navegación lateral pt2. Fuente. Elaboración propia.	174
Figura. 187 Configuración de la barra de navegación lateral pt3. Fuente. Elaboración propia.	175
Figura. 188 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	175
Figura. 189 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	176
Figura. 190 Creación de contenedor de indicaciones del formulario. Fuente. Elaboración propia.	176
Figura. 191 Creación de enlaces navegadores del archivo compras.php. Fuente. Elaboración propia.	177
Figura. 192 Creación de título del formulario e indicaciones. Fuente. Elaboración propia.	177
Figura. 193 Creación de los campos para el formulario de compras. Fuente. Elaboración propia.	177
Figura. 194 Codificación de la estructura de los campos para el formulario. Fuente. Elaboración propia.	178
Figura. 195 Codificación de la estructura de listas de selección. Fuente. Elaboración propia.	178
Figura. 196 Creación del botón agregar producto. Fuente. Elaboración propia.	178
Figura. 197 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente. Elaboración propia.	179
Figura. 198 Codificación para el llamado de tablas en Firebase. Fuente. Elaboración propia.	179
Figura. 199 Codificación de verificación para datos repetidos de los campos. Fuente. Elaboración propia.	180
Figura. 200 Actualización de cantidades e ingreso de datos a las tablas. Fuente. Elaboración propia.	180
Figura. 201 Creación de la tabla de registros para compras. Fuente. Elaboración propia.	181
Figura. 202 Codificación de llenado de la tabla para la tabla compras. Fuente. Elaboración propia.	181
Figura. 203 Codificación del cálculo del precio con IVA. Fuente. Elaboración propia.	182
Figura. 204 Codificación de descarga y conversión a formato Excel. Fuente. Elaboración propia.	182
Figura. 205 Configuración de lenguaje y tiempo de inactividad del archivo inventario.php, Fuente. Elaboración propia.	183
Figura. 206 Estructura de lenguaje HTML e Idioma del archivo invntario.php. Fuente. Elaboración propia.	184
Figura. 207 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.	184

Figura. 208 Configuración de la barra de navegación lateral pt1. Fuente. Elaboración propia.	185
Figura. 209 Configuración de la barra de navegación lateral pt2. Fuente. Elaboración propia.	185
Figura. 210 Configuración de la barra de navegación lateral pt3. Fuente. Elaboración propia.	186
Figura. 211 Configuración de la barra de navegación lateral pt4. Fuente. Elaboración propia.	186
Figura. 212 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	187
Figura. 213 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	187
Figura. 214 Creación del contenedor de indicaciones de la tabla inventario. Fuente. Elaboración propia.	187
Figura. 215 Creación de la tabla de registros para el inventario. Fuente. Elaboración propia.	188
Figura. 216 Codificación y evaluación del ingreso de datos a la tabla inventario. Fuente. Elaboración propia.	188
Figura. 217 Codificación y conversión de registros a formato Excel. Fuente. Elaboración propia.	189
Figura. 218 Creación y configuración del archivo editar.php. Fuente. Elaboración propia.	189
Figura. 219 Codificación y llamado de registros en los campos del formulario. Fuente. Elaboración propia.	190
Figura. 220 Creación del botón agregar producto. Fuente. Elaboración propia.	190
Figura. 221 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente. Elaboración propia.	191
Figura. 222 Codificación de ingreso de datos actualizados a la tabla productos. Fuente. Elaboración propia.	191
Figura. 223 Creación de la tabla de registros actualizados para el inventario. Fuente. Elaboración propia.	192
Figura. 224 Codificación de enlaces de edición y eliminación. Fuente. Elaboración propia.	192
Figura. 225 Configuración de lenguaje y tiempo de inactividad del archivo personal.php. Fuente. Elaboración propia.	193
Figura. 226 Estructura de lenguaje HTML e Idioma del archivo personal.php. Fuente. Elaboración propia.	194
Figura. 227 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.	194
Figura. 228 Configuración de la barra de navegación lateral pt1. Fuente. Elaboración propia.	195
Figura. 229 Configuración de la barra de navegación lateral pt2. Fuente. Elaboración propia.	195
Figura. 230 Configuración de la barra de navegación lateral pt3. Fuente. Elaboración propia.	196
Figura. 231 Configuración de la barra de navegación lateral pt4. Fuente. Elaboración propia.	196
Figura. 232 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	197
Figura. 233 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente.	

Elaboración propia.....	197
Figura. 234 Creación del contenedor de indicaciones para el formulario. Fuente. Elaboración propia.....	198
Figura. 235 Creación de enlaces navegadores del archivo personal.php. Fuente. Elaboración propia.	198
Figura. 236 Creación de título del formulario e indicaciones. Fuente. Elaboración propia.	198
Figura. 237 Creación de los campos para el formulario de personal. Fuente. Elaboración propia.	199
Figura. 238 Codificación de la estructura de los campos para el formulario. Fuente. Elaboración propia.	199
Figura. 239 Creación del botón agregar personal. Fuente. Elaboración propia.	199
Figura. 240 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente. Elaboración propia	200
Figura. 241 Codificación para el llamado de tablas en Firebase. Fuente. Elaboración propia.	200
Figura. 242 Codificación de verificación para datos repetidos de los campos. Fuente. Elaboración propia.	201
Figura. 243 Codificación de ingreso de datos a la tabla personal. Fuente. Elaboración propia.	201
Figura. 244 Creación de la tabla de registros para el personal, Fuente. Elaboración propia.	202
Figura. 245 Configuración para la visualización de los registros de la tabla de personal, Fuente. Elaboración propia.	202
Figura. 246 Codificación y conversión de registros a formato Excel, Fuente. Elaboración propia.	203
Figura. 247 Creación y configuración del archivo editar.php, Fuente. Elaboración propia.	203
Figura. 248 Codificación y llamado de registros en los campos del formulario personal, Fuente. Elaboración propia.	204
Figura. 249 Creación del botón agregar personal, Fuente. Elaboración propia.	204
Figura. 250 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente, Elaboración propia.	205
Figura. 251 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia.	205
Figura. 252 Codificación de ingreso de datos actualizados a la tabla personal, Fuente. Elaboración propia.	206
Figura. 253 Creación de la tabla de registros actualizados para el personal, Fuente. Elaboración propia.	206
Figura. 254 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente, Elaboración propia.	207
Figura. 255 Codificación y eliminación de registros de la tabla personal, Fuente. Elaboración propia.	207
Figura. 256 Codificación de enlaces de edición y eliminación, Fuente. Elaboración propia.	208
Figura. 257 Configuración de lenguaje y tiempo de inactividad del archivo productos.php, Fuente. Elaboración propia.	208

Figura. 258 Ingreso de Título y enlaces de funciones del archivo productos.php. Fuente. Elaboración propia.	209
Figura. 259 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.	209
Figura. 260 Configuración de la barra de navegación lateral pt1. Fuente. Elaboración propia.	210
Figura. 261 Configuración de la barra de navegación lateral pt2. Fuente. Elaboración propia.	210
Figura. 262 Configuración de la barra de navegación lateral pt3. Fuente. Elaboración propia.	211
Figura. 263 Configuración de la barra de navegación lateral pt4, Fuente. Elaboración propia.	211
Figura. 264 Creación de los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.	211
Figura. 265 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	212
Figura. 266 Creación del contenedor de indicaciones para el formulario, Fuente. Elaboración propia.	212
Figura. 267 Creación de enlaces navegadores del archivo productos.php. Fuente. Elaboración propia.	212
Figura. 268 Creación de título del formulario e indicaciones. Fuente. Elaboración propia.	213
Figura. 269 Creación de los campos para el formulario de productos, Fuente. Elaboración propia.	213
Figura. 270 Codificación de la estructura de los campos para el formulario, Fuente. Elaboración propia.	214
Figura. 271 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente. Elaboración propia.	214
Figura. 272 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente, Elaboración propia.	214
Figura. 273 Codificación para el llamado de tablas en Firebase, Fuente. Elaboración propia.	215
Figura. 274 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia.	215
Figura. 275 Codificación de ingreso de datos a la tabla productos, Fuente. Elaboración propia.	215
Figura. 276 Creación de la tabla de registros para productos, Fuente. Elaboración propia.	216
Figura. 277 Conexión a la base de datos e incorporación de registros de la tabla productos, Fuente. Elaboración propia.	216
Figura. 278 Codificación y conversión de registros a formato Excel. Fuente. Elaboración propia.	217
Figura. 279 Creación y configuración del archivo editar.php, Fuente. Elaboración propia.	217
Figura. 280 Codificación y llamado de registros en los campos del formulario productos. Fuente. Elaboración propia.	218
Figura. 281 Creación del botón agregar producto, Fuente. Elaboración propia.	218
Figura. 282 Codificación de tiempo de inactividad del archivo action_editar.php. Fuente. Elaboración propia.	218
Figura. 283 Conexión a la base de datos y verificación de campos. Fuente. Elaboración propia.	219

Figura. 284 Codificación de actualización de registros de la tabla productos, Fuente. Elaboración propia.	219
Figura. 285 Configuración para la visualización de los registros de la tabla de productos. Fuente. Elaboración propia.	220
Figura. 286 Codificación de tiempo de inactividad del archivo action_eliminar.php, Fuente, Elaboración propia.	220
Figura. 287 Codificación y eliminación de registros de la tabla usuario, Fuente. Elaboración propia.	221
Figura. 288 Codificación de vínculos para los botones eliminar y editar, Fuente. Elaboración propia.	221
Figura. 289 Configuración de lenguaje y tiempo de inactividad del archivo proveedores.php. Fuente. Elaboración propia.	222
Figura. 290 Estructura de lenguaje HTML e Idioma del archivo proveedores.php. Fuente. Elaboración propia.	222
Figura. 291 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.	223
Figura. 292 Configuración de la barra de navegación lateral pt1. Fuente. Elaboración propia.	223
Figura. 293 Configuración de la barra de navegación lateral pt2. Fuente. Elaboración propia.	224
Figura. 294 Configuración de la barra de navegación lateral pt3. Fuente. Elaboración propia.	224
Figura. 295 Configuración de la barra de navegación lateral pt4. Fuente. Elaboración propia.	225
Figura. 296 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	225
Figura. 297 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.	226
Figura. 298 Creación del contenedor de indicaciones para el formulario, Fuente. Elaboración propia.	226
Figura. 299 Creación de enlaces navegadores del archivo proveedores.php. Fuente. Elaboración propia.	227
Figura. 300 Creación de título del formulario e indicaciones, Fuente. Elaboración propia.	227
Figura. 301 Codificación de la estructura de listas de selección. Fuente. Elaboración propia.	227
Figura. 302 Codificación de la estructura de los campos para el formulario, Fuente. Elaboración propia.	228
Figura. 303 Creación del botón agregar proveedor. Fuente. Elaboración propia.	228
Figura. 304 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente, Elaboración propia.	228
Figura. 305 Conexión a la base de datos y almacenamiento de información de listas, Fuente. Elaboración propia.	229
Figura. 306 Codificación para el llamado de tablas en Firebase, Fuente. Elaboración propia.	229
Figura. 307 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia.	230
Figura. 308 Codificación de ingreso de datos a la tabla proveedores, Fuente. Elaboración propia.	230

Figura. 309 Creación de la tabla de registros para proveedores, Fuente. Elaboración propia.	231
Figura. 310 Conexión a la base de datos e incorporación de registros de la tabla proveedores, Fuente. Elaboración propia.	231
Figura. 311 Codificación y conversión de registros a formato Excel, Fuente. Elaboración propia.	232
Figura. 312 Creación y configuración del archivo editar.php. Fuente. Elaboración propia.	232
Figura. 313 Codificación y llamado de registros en los campos del formulario proveedor, Fuente. Elaboración propia.	233
Figura. 314 Creación del botón guardar cambio, Fuente. Elaboración propia.	233
Figura. 315 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente. Elaboración propia.	233
Figura. 316 Conexión a la base de datos y verificación de campos, Fuente. Elaboración propia.	234
Figura. 317 Codificación de ingreso de datos a la tabla productos, Fuente. Elaboración propia.	234
Figura. 318 Creación de la tabla de registros actualizados para proveedores. Fuente. Elaboración propia.	235
Figura. 319 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente. Elaboración propia.	235
Figura. 320 Codificación y eliminación de registros de la tabla proveedores, Fuente. Elaboración propia.	236
Figura. 321 Codificación de vínculos para los botones eliminar y editar, Fuente. Elaboración propia.	236
Figura. 322 Configuración de lenguaje y tiempo de inactividad del archivo proyectos.php. Fuente. Elaboración propia.	237
Figura. 323 Estructura de lenguaje HTML e Idioma del archivo proyectos.php, Fuente. Elaboración propia.	238
Figura. 324 Codificación del nombre y avatar del usuario, Fuente. Elaboración propia.	238
Figura. 325 Configuración de la barra de navegación lateral pt1, Fuente. Elaboración propia.	239
Figura. 326 Configuración de la barra de navegación lateral pt2, Fuente. Elaboración propia.	239
Figura. 327 Configuración de la barra de navegación lateral pt3, Fuente. Elaboración propia.	240
Figura. 328 Configuración de la barra de navegación lateral pt4. Fuente. Elaboración propia.	240
Figura. 329 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	241
Figura. 330 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	241
Figura. 331 Creación del contenedor de indicaciones para el formulario, Fuente. Elaboración propia.	242
Figura. 332 Creación de enlaces navegadores del archivo proyectos.php. Fuente. Elaboración propia.	242
Figura. 333 Creación de título del formulario e indicaciones, Fuente. Elaboración propia.	242

Figura. 334 Creación de los campos para el formulario de proyectos, Fuente. Elaboración propia.	243
Figura. 335 Codificación de la estructura de los campos para el formulario. Fuente. Elaboración propia.	243
Figura. 336 Creación del botón agregar proyecto, Fuente. Elaboración propia.	244
Figura. 337 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente, Elaboración propia.	244
Figura. 338 Conexión a la base de datos e incorporación de registros en listas del formulario, Fuente. Elaboración propia.	245
Figura. 339 Codificación de verificación para datos repetidos de los campos. Fuente. Elaboración propia.	245
Figura. 340 Codificación de ingreso de datos a la tabla proyectos, Fuente. Elaboración propia.	246
Figura. 341 Creación de la tabla de registros para proyectos. Fuente. Elaboración propia.	246
Figura. 342 Conexión a la base de datos e incorporación de registros en la tabla proyectos, Fuente. Elaboración propia.	246
Figura. 343 Codificación y conversión de registros a formato Excel, Fuente. Elaboración propia.	247
Figura. 344 Creación y configuración del archivo editar.php. Fuente. Elaboración propia.	247
Figura. 345 Codificación y llamado de registros en los campos del formulario proyecto, Fuente. Elaboración propia.	248
Figura. 346 Creación del botón agregar proyecto, Fuente. Elaboración propia.	248
Figura. 347 Codificación de tiempo de inactividad del archivo action_editar.php. Fuente, Elaboración propia.	249
Figura. 348 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia.	249
Figura. 349 Codificación de ingreso de datos a la tabla proyecto, Fuente. Elaboración propia.	250
Figura. 350 Creación de la tabla de registros actualizados para proyectos, Fuente. Elaboración propia.	250
Figura. 351 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente, Elaboración propia.	251
Figura. 352 Codificación y eliminación de registros de la tabla proyecto, Fuente. Elaboración propia.	251
Figura. 353 Codificación de vínculos para los botones eliminar y editar, Fuente. Elaboración propia.	252
Figura. 354 Configuración de lenguaje y tiempo de inactividad del archivo salidas.php. Fuente. Elaboración propia.	252
Figura. 355 Estructura de lenguaje HTML e Idioma del archivo salidas.php, Fuente. Elaboración propia.	253
Figura. 356 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.	254
Figura. 357 Configuración de la barra de navegación lateral pt1, Fuente. Elaboración propia.	254

Figura. 358 Configuración de la barra de navegación lateral pt2, Fuente. Elaboración propia.	255
Figura. 359 Configuración de la barra de navegación lateral pt3, Fuente. Elaboración propia.	255
Figura. 360 Configuración de la barra de navegación lateral pt4. Fuente. Elaboración propia.	256
Figura. 361 Creación de los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.	256
Figura. 362 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.	257
Figura. 363 Creación de la tabla de registros para salidas. Fuente. Elaboración propia.	257
Figura. 364 Codificación de tiempo de inactividad del archivo action_editar.php, Fuente, Elaboración propia.	258
Figura. 365 Conexión a la base de datos e incorporación de registros en listas del formulario. Fuente. Elaboración propia.	258
Figura. 366 Codificación de ingreso de datos a la tabla productos, Fuente. Elaboración propia.	259
Figura. 367 Creación de la tabla de registros actualizados para las salidas, Fuente. Elaboración propia.	259
Figura. 368 Codificación de ingreso de datos a la tabla de salidas, Fuente. Elaboración propia.	260
Figura. 369 Codificación y conversión de registros a formato Excel, Fuente. Elaboración propia.	260
Figura. 370 Codificación y llamado de registros en los campos del formulario de salidas, Fuente. Elaboración propia.	261
Figura. 371 Codificación de enlaces de navegación y conexión a la base de datos, Fuente. Elaboración propia.	261
Figura. 372 Conexión a la base de datos y codificación de consulta de datos, Fuente. Elaboración propia.	262
Figura. 373 Configuración de lenguaje y tiempo de inactividad del archivo ubicaciones.php, Fuente. Elaboración propia.	262
Figura. 374 Estructura de lenguaje HTML e Idioma del archivo ubicaciones.php, Fuente. Elaboración propia.	263
Figura. 375 Codificación del nombre y avatar del usuario, Fuente. Elaboración propia.	263
Figura. 376 Configuración de la barra de navegación lateral pt1, Fuente. Elaboración propia.	264
Figura. 377 Configuración de la barra de navegación lateral pt2, Fuente. Elaboración propia.	264
Figura. 378 Configuración de la barra de navegación lateral pt3, Fuente. Elaboración propia.	265
Figura. 379 Configuración de la barra de navegación lateral pt4, Fuente. Elaboración propia.	265
Figura. 380 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.	266
Figura. 381 Creación del contenedor de indicaciones para el formulario. Fuente. Elaboración propia.	266
Figura. 382 Creación del contenedor de indicaciones para el formulario. Fuente. Elaboración propia.	266
Figura. 383 Creación de enlaces navegadores del archivo ubicaciones.php. Fuente. Elaboración propia.	267

Figura. 384 Creación de título del formulario e indicaciones, Fuente. Elaboración propia.	267
Figura. 385 Creación de los campos para el formulario de ubicaciones, Fuente. Elaboración propia.	267
Figura. 386 Codificación de la estructura de los campos para el formulario, Fuente. Elaboración propia.	268
Figura. 387 Creación del botón agregar ubicación, Fuente. Elaboración propia.	268
Figura. 388 Codificación de tiempo de inactividad del archivo action_agregar.php, Fuente. Elaboración propia.	269
Figura. 389 Conexión a la base de datos e incorporación de registros en listas del formulario, Fuente. Elaboración propia.	269
Figura. 390 Conexión a la base de datos y verificación de campos, Fuente. Elaboración propia.	269
Figura. 391 Codificación de ingreso de datos a la tabla ubicación. Fuente. Elaboración propia.	270
Figura. 392 Creación de la tabla de registros actualizados para ubicaciones, Fuente. Elaboración propia.	270
Figura. 393 Conexión a la base de datos y incorporación de registros en la tabla ubicaciones, Fuente. Elaboración propia.	271
Figura. 394 Codificación y conversión de registros a formato Excel, Fuente. Elaboración propia.	271
Figura. 395 Codificación de tiempo de inactividad del archivo nuevo_nivel.php. Fuente. Elaboración propia.	272
Figura. 396 Codificación de tiempo de inactividad del archivo action_nivel.php. Fuente, Elaboración propia.	272
Figura. 397 Creación y configuración del archivo editar.php, Fuente. Elaboración propia.	273
Figura. 398 Creación del botón guardar cambio, Fuente. Elaboración propia.	273
Figura. 399 Codificación de tiempo de inactividad del archivo action_editar.php. Fuente. Elaboración propia.	273
Figura. 400 Codificación de verificación para datos repetidos de los campos. Fuente. Elaboración propia.	274
Figura. 401 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia.	274
Figura. 402 Codificación de ingreso de datos a la tabla ubicaciones, Fuente. Elaboración propia.	274
Figura. 403 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente. Elaboración propia.	275
Figura. 404 Codificación y eliminación de registros de la tabla ubicaciones. Fuente. Elaboración propia.	275
Figura. 405 Codificación de vínculos para los botones eliminar y editar. Fuente. Elaboración propia.	275
Figura. 406 Contenido de la carpeta usuario. Fuente. Elaboración propia.	276
Figura. 407 Prueba de inicio de sesión. Fuente elaboración propia.	280

Figura. 408 Pruebas de registro de producto. Fuente. Elaboración propia.....	281
Figura. 409 Prueba de lista de productos agregados. Fuente. Elaboración propia.....	281
Figura. 410 Prueba de editar producto. Fuente. Elaboración propia.....	282
Figura. 411 Prueba de cambio de proyecto. Fuente. Elaboración propia.....	283
Figura. 412 Interfaz de OWASP.....	284
Figura. 413 Interfaz de escaneo automatizado	284
Figura. 414 URL a escanear	284
Figura. 415 Historial de análisis	285
Figura. 416 URL detectadas	285
Figura. 417 Pestaña de análisis de alertas.	285
Figura. 418 Resultados de vulnerabilidades. Fuente. Elaboración propia.....	286
Figura. 419 Topología de host,Fuente .Elaboración propia	286
Figura. 420 Detalles del Host, Fuente. Elaboración propia.....	287
Figura. 421 Inicio de sesión- administrador, Fuente. Elaboración propia.....	296
Figura. 422 Inicio de sesión- usuario operativo, Fuente. Elaboración propia	296
Figura. 423 Inicio de sesión, Fuente. Elaboración propia	297
Figura. 424 Alerta de fallo de inicio se sesión, Fuente. Elaboración propia	297
Figura. 425 Inicio de sesión v3, Fuente. Elaboración propia	298
Figura. 426 Alerta contraseña incorrecta, Fuente. Elaboración propia	298
Figura. 427 Inicio de sesión v4, Fuente. Elaboración propia	299
Figura. 428 Alerta de correo. Fuente. Elaboración propia	299
Figura. 429 Alerta de correo incorrecto. Fuente. Elaboración propia	299
Figura. 430 Inicio de sesión v5, Fuente Elaboración propia	300
Figura. 431 Pantalla de inicio, Fuente. Elaboración propia.....	300
Figura. 432 Pantalla de menú, Fuente. Elaboración propia.....	301
Figura. 433 Cinta de opciones del menú, Fuente. Elaboración propia.....	301
Figura. 434 Cinta de opciones del menú 2, Fuente. Elaboración propia.....	302
Figura. 435 Registro de usuarios (Administrador), Fuente. Elaboración propia	302
Figura. 436 Registro de usuarios v2, Fuente. Elaboración propia	303
Figura. 437 Ingreso de correo, Fuente. Elaboración propia	303
Figura. 438 Lista de usuarios registrados, Fuente. Elaboración propia.....	304
Figura. 439 Eliminar usuario, Fuente. Elaboración propia.....	304
Figura. 440 Alerta de datos eliminados, Fuente. Elaboración propia.	305
Figura. 441 Editar usuario, Fuente. Elaboración propia.....	305

Figura. 442 Editar usuario formulario, Fuente. Elaboración propia	305
Figura. 443 Lista de usuarios administradores, Fuente. Elaboración propia.....	306
Figura. 444 Registro de proveedores, Fuente. Elaboración propia	306
Figura. 445 Sección nuevo proveedor, Fuente. Elaboración propia	307
Figura. 446 Dirección, Fuente. Elaboración propia	307
Figura. 447 Contactos, Fuente Elaboración propia.....	307
Figura. 448 Lista de proveedores, Fuente. Elaboración propia	308
Figura. 449 Eliminar registro, Fuente. Elaboración propia	308
Figura. 450 Alerta de datos eliminados.....	309
Figura. 451 Descarga y genera reporte.....	309
Figura. 452 Reporte descargado	309
Figura. 453 Cerrar sesión	310
Figura. 454 Confirmación de cerrar sesión	310
Figura. 455 Presentación img1 Figura. 456. Presentación img2	312
Figura. 457 Anaqueles1 Figura. 458 Anaqueles2.....	313
Figura. 459 Maderas Figura. 460 Anaqueles 3.....	313
Figura. 461 Anaqueles4	314
Figura. 462 Espacio de trabajo Figura. 463 Espacio de trabajo	314
Figura. 464 Espacio de trabajo	314

ÍNDICE DE TABLAS

Tabla 1 Comparación entre Baas. Fuente Tomás,J. (2019).....	43
Tabla 2. Cuadro metodológico.	57
Tabla 3 Características de usuarios.....	61
Tabla 4 Escenario de requisito funcional 1	76
Tabla 5. Escenario de requisito funcional 2	79
Tabla 6 Escenario de requisito funcional 3. Elaboración propia	83
Tabla 7 Escenario de requisito funcional 4. Elaboración propia.	86
Tabla 8 Escenario de requisito funcional 5. Elaboración propia.	89
Tabla 9 Escenario de requisito funcional 6 Elaboración propia.	91
Tabla 10 Escenario de requisito funcional 7. Elaboración propia.	94
Tabla 11 Escenario de requisito funcional 8. Elaboración propia.	97
Tabla 12 Escenario de requisito funcional 9. Elaboración propia.	101
Tabla 13 Escenario de requisito funcional 10. Fuente. Elaboración propia	103
Tabla 14 Escenario de requisito funcional 11. Elaboración propia.	106
Tabla 15 Escenario de requisito funcional 12. Elaboración propia.	109
Tabla 16 Escenario de requisito funcional 13. Elaboración propia.	112
Tabla 17 Prueba de autenticarse. Elaboración propia.	279
Tabla 18 Prueba, registro de producto. Fuente. Elaboración propia.	280
Tabla 19 Prueba de editar producto. Elaboración propia.	282

CAPÍTULO I. GENERALIDADES DEL PROYECTO

1. INTRODUCCIÓN

Una de las partes más importantes de una empresa es el uso y manejo de su inventario de productos, esto con el propósito de no tener pérdidas principalmente económicas y errores de planeación y retraso de producción, siendo así una parte esencial para mantener eficiencia operativa, reducción de costos, asegurando tener los productos necesarios disponibles para llevar a cabo el trabajo sin atrasar tiempos de producción.

Siendo así que este proyecto se centra en explorar y mejorar las prácticas de gestión de inventario con el propósito de brindar una visión clara y detallada de la problemática y la solución técnica que se abordará la cual integra el proceso de desarrollo de software que comprende de varias etapas, incluyendo el análisis de requerimientos, el diseño del sistema, la programación y la implementación, así como la capacitación al usuario. Dada la situación en la que se enfrenta la empresa CC Camaleón Color en cuanto a las complicaciones en la gestión de su inventario, abordando así la problemática mediante el uso de tecnologías actualmente funcionales para llevar a cabo esta solución técnica, este proyecto presenta un paso importante para la respuesta a una solución de la necesidad que tiene la empresa específicamente en el área asignada, calidad y almacén área en la que se trabaja de la mano.

Mencionando así también que la implementación de nuevas tecnologías proporciona seguridad de datos, innovación, entre otras puesto que la adopción de tecnologías modernas es fundamental para el éxito a largo plazo en el desarrollo de software.

1.2. DESCRIPCIÓN DE LA EMPRESA

CC camaleón color S.A. De C.V. es una empresa mexicana dedicada a crear herramientas de color y producción de muestrarios.

Especializada en la igualación de colores de pintura, y en la aplicación de pigmentos de color real en cualquier sustrato por medio de serigrafía y/o impresión directa de color (depósito) de pintura.

- **Misión:**

Crear herramientas publicitarias que muestren el color real de una manera innovadora, con calidad y profesionalismo, que puedan ofrecer una mejor experiencia del color.

- **Visión:**

Ser los mejores en la industria del color, brindando servicios de excelencia en todas nuestras herramientas del color.

- **Área directa de trabajo: Calidad y almacén**

Responsable de recibir y registrar los materiales de entrada y salida de la empresa, así como los proyectos que se entregan a clientes lo que implica la verificación de la calidad de productos recibidos y a entregar.

1.2.1. ORGANIGRAMA

En la figura 1 se representa el organigrama de la empresa CC Camaleón Color

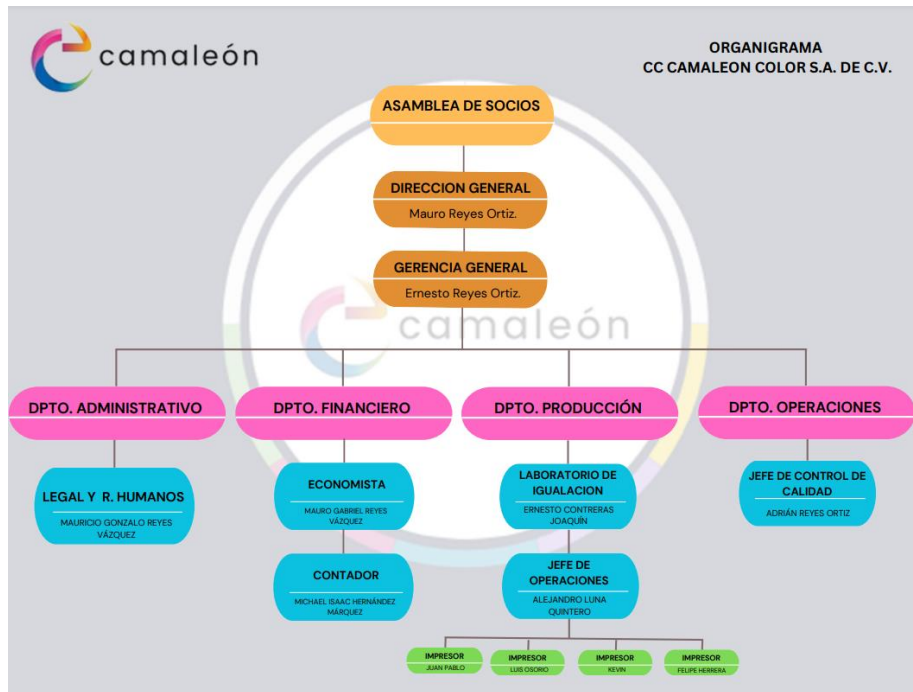


Figura. 1 Organigrama de CC Camaleón Color. Fuente. Elaboración propia.

Empresa ubicada en el domicilio Calle Azucena Mza. 5, Lt.8 Paraje San Juan, Iztapalapa. Dirección gráfica en la figura 2.



Figura. 2. Mapa de ubicación de CC Camaleón Color. Fuente Google Maps.

1.3. PLANTEAMIENTO DEL PROBLEMA

La empresa CC Camaleón color actualmente opera en el sector de la serigrafía, herramientas de color y producción de muestrarios teniendo un crecimiento significativo desde hace varios en el mercado. Sin embargo, se identificó que a pesar de este crecimiento que han tenido, la empresa carece de un sistema de inventario adecuado para gestionar el control de disponibilidad de su material. La falta de un control de inventario genera problemas en el orden de las entradas y salidas de productos de pintura y otros materiales necesarios para hacer cada proyecto, por otro lado, la empresa ha presentado pérdidas económicas ya que no se prevé la compra anticipada de productos necesarios para la producción de muestrarios, esto debido a que no se lleva un control y no se da seguimiento de lo que se tiene en existencia y lo que falta por comprar, por lo que ha llevado a costos adicionales o de emergencia.

Actualmente se realiza un seguimiento manual del inventario, propiciando pérdida de tiempo, pérdida en precisión y exactitud de datos, así como mayor probabilidad de errores de pérdida de información, lo que ha llevado a una ineficiencia operativa significativa por la falta de un sistema de inventario, lo que le ha dificultado a la empresa la toma de decisiones estratégicas, que es principalmente la identificación de productos que deben ser discontinuados o la planificación de buenas compras.

1.4. JUSTIFICACIÓN

La implementación de un sistema de inventario le proporcionará a la empresa CC Camaleón color y al área de almacén una visión de su inventario disponible al momento, reducir la inconsistencia de datos, duplicidad de datos, y apoyar al manejo en tiempo eficiente de la transacción de datos entre usuarios, así como optimización de inventario, mejora de la planificación de producción, reducción de costos de almacenamiento, mejora de toma de decisiones, reducción de pérdidas y robos, A su vez el sistema estará conformado por: los lenguajes de alto nivel de programación (PHP y JavaScript) que conformarán la parte de interfaz de usuario y lógica del sistema , una base de datos ubicada en la nube para el almacenamiento de información (Firebase), seguridad de cifrado de contraseña.

La implementación de estas tecnologías como lo es Firebase la cual es una plataforma de desarrollo en la nube que ofrece herramientas para sitios web proporcionando servicios de gestión y almacenamiento de información, escalabilidad y confiabilidad.

Promoviendo estas nuevas tecnologías de almacenamiento en la nube, a su vez se implementan los lenguajes de PHP y JavaScript los cuales permiten el desarrollo de interfaces gráficas y funciones de métodos lógicos para el desarrollo de procesos estructurados de programación, su amplia rama de paquetería proporciona una gama de diversas opciones de estructuras y diseños para los sistemas.

1.5. OBJETIVOS

1.5.1. OBJETIVO GENERAL

Desarrollar un sistema web de inventario para gestionar las entradas, salidas y compras de pinturas en el departamento de calidad y almacén de CC camaleón color.

1.5.2. OBJETIVOS ESPECÍFICOS

1. Obtener información necesaria para el diseño de los requisitos para el desarrollo del sistema web de inventario.
2. Diseño de los requisitos del sistema web de inventario, clasificando información, maquetado de las interfaces y diseño de la base de datos (tablas).
3. Programación de los módulos (Autenticarse y registrar, interfaz, compras, agregar, eliminar y editar) del sistema web, codificar y conectar la base de datos por Firebase y PHP.
4. Probar los requisitos funcionales y no funcionales del sistema web para inventario.
5. Realizar escaneo de puertos en el servidor del sitio web con la herramienta Nmap, así como escaneo de vulnerabilidades con OWASP-ZAP analizando las URL pertenecientes al sistema web, prueba de funciones principales (CRUD), elaboración de manuales (Usuario-Técnico) siguiendo un mantenimiento y monitoreo.

CAPÍTULO II. MARCO TEÓRICO

2. Escenario teórico de investigación

En este capítulo se detalla el marco teórico con temas relevantes respecto al proyecto, integrando conceptos que sirven para el desarrollo del proyecto, explorando investigaciones que permiten identificar temas relevantes sobre el proyecto y se tenga una comprensión más profunda al desarrollo del proyecto.

2.1.Ingeniería de software

Disciplina que se encarga de todos los aspectos del desarrollo del software se incluyen actividades de ingeniería de requisitos, modelado y técnicas (Sommerville, 2003).

Según (Presman, 2006) el software es algo que se desarrolla, no se crea a partir de una materia prima, un producto que no se fabrica, si no se construye a través de un proceso.

2.1.1. Arquitectura de software

La arquitectura de software lleva a cabo para el desarrollo de software por la cual se podrá analizar el impacto que tienen los posibles cambios a lo largo del desarrollo.

La arquitectura de software ayuda a identificar elementos y componentes que se van a necesitar desarrollar.

2.2.Ingeniería de requisitos

Según (Ramos C.) en requisito surge a partir de una necesidad y la idea de desarrollar un software que cumpla esas necesidades, este proceso impacta principalmente en la estimación de software.

Se identifica por (Pressman, 2006) que la ingeniería de requisitos es una actividad en la ingeniería de software la cual busca comprender el sistema que está a desarrollar. Menciona también que los requisitos deben ser gestionados e identificados en todo momento en el proyecto esto con el fin de reflejar al cliente como se aplicaron sus necesidades al sistema el cual debe

comprender por medio de estos requisitos

2.2.1. Definición de requisito

(Sommerville 2003) identifica que un requisito puede definirse como una visión de una función. Por lo que partiendo de esa definición el autor menciona que existen diferentes niveles de detallar los requisitos dividiéndolo en:

- **Requisitos del usuario**

Se proporciona sentencias en lenguaje natural esto acerca de las funciones que el sistema deba resolver.

- **Requisitos del sistema**

Se detallan funciones y restricciones del sistema de la mano con el desarrollo de software.

- **Especificación del proyecto de software**

Es una descripción del proyecto agregando a detalle la solución a los requisitos del sistema.

2.2.2. Tipos de requisitos

Dentro de los requisitos hay ciertas clasificaciones las cuales son requisitos funcionales y no funcionales.

Requisitos funcionales: según (Sommerville,2003) son aquellos que están relacionados directamente con las funciones que el sistema deberá proporcionar de acuerdo con la entrevista que se tuvo con el cliente y sus necesidades.

Requisitos no funcionales: Estos requisitos son restricciones tales como seguridad, integridad, limitaciones de tiempo etc. Los cuales son importantes considerar ya que afectan a las estimaciones de proyecto.

2.3.Diagramas UML

Los diagramas UML según (Fontela, 2017) es una notación de modelado visual lo cual vca a apoyar al desarrollador y al cliente a mostrar distintas funciones y/o aspectos del sistema, en el caso de uso general o caso de uso por requisitos funcionales.

Surge en el año 1995 por Grady Booch, James Rumbaugh e Ivar Jacobson ingenieros de software, contaban con un avance en sus anotaciones e modelado, por lo que UML se define como “unificado” ya que surgió de los mejores elementos de las anotaciones previas con las que contaban los autores.

Tras pasar los años ya en el año 2000 UML se convirtió en el estándar de fato para la modelización de software. Actualmente UML es un lenguaje de visualización especificación y documentación de software el cual constan de trece tipos de diagramas.

2.3.1. Diagramas de casos de uso en UML

En (Natsys, 2017) se menciona que un caso de uso es una interacción entre el usuario y el sistema, el cual se podrá visualizar de manera grafica.

Por lo que un diagrama de caso de uso está formado principalmente por:

Un **actor** este elemento es importante ya que es el que interactúa con el sistema directamente, aunque un actor no necesariamente tiene que ser un humano, esto se refiera a que puede ser utilizado por otro sistema externo que podría tomar información del sistema actual.

Una **relación** <<extend>> la cual se va a utilizar en especificar una característica que tenga una función opcional o este reflejado en algún otro caso de uso.

Se considera igualmente <<include>> la cual va a representar una característica que tenga que ver con otro caso de uso y obligatoriamente se debe usar porque lo incluye.

En términos generales un modelo de casos de uso es de los más adaptables y amigables para el análisis del desarrollo de software puesto que se va construyendo durante un proceso iterativo con la recolección de la información mediante reuniones con el cliente y/o usuarios finales.

Lo interesante del diagrama de caso de uso es que se identifican de manera grafica las acciones y el comportamiento del sistema, así como el comportamiento de los usuarios en el sistema.

2.3.2. Elementos de diagrama de caso de uso

Detallando los demás elementos que conforma los diagramas de casos de uso el cual contiene los requisitos funcionales existentes, se cuenta con un dialogo entre el desarrollador y el cliente. Entre sus elementos según (Debrauwer, L., Van der Heyde, F. 2016) se encuentran:

- Actores:

Un actor representa al usuario externo al sistema este elemento está representado como actor el cual puede realizar diferentes funciones en un solo sistema el cual cumple con un rol.

- Relación de inclusión:

La cual tiene el propósito de compartir una función entre varios casos de uso.

2.4. Internet

Es la red de redes de ordenadores de cualquier tipo los cuales se entrelazan mediante un lenguaje denominado como el protocolo TC/IP. Esa primera cualidad de la heterogeneidad de los equipos conectados es la clave para comprender el funcionamiento de cualquier servicio de la red, así mismo también entender la necesidad de la portabilidad en los ámbitos que hagan uso de este (Cobo, 2005).

2.4.1. Protocolos

Los protocolos permiten compartir los recursos de las computadoras de una red en la que estén conectadas. Los protocolos fueron desarrollados en 1972 por el departamento de defensa de estados unidos para poder conectar los recursos de la red ARPANET, con el pasar de los años se han convertido en parte fundamental de la internet en la cual se conocen los famosos protocolos TC/IP los cuales están englobados dentro del conjunto genérico.

El uso de los protocolos TC/IP transmiten información en forma de diagramas los cuales dentro de estos contiene una cantidad de información la cual se quiere transmitir, por lo que cada uno de estos diagramas son enviados individualmente a la red para que una vez llegada a su destino esta pueda ser reconstruida de manera original y muestre la información (E, 2008).

2.4.2. Modelo Cliente- Servidor

En (Cobo, 2005) el autor también menciona que cualquier servicio que ofrezca internet donde se resalta el servicio web, es basado en la conexión de cliente/servidor, es esencial comprender esta conexión para posteriormente comprender los lenguajes de la programación web.

2.4.3. Servidor

Es el conjunto de ordenadores que ofrecen sus funcionalidades a los demás equipos conectados. Estos frecuentan tener una presencia estable en la red, lo que determina que tengan direcciones IP de manera permanente.

2.4.4. Cliente

Equipos individuales utilizados por los usuarios para vincularse a la red y solicitar servicios a los servidores. Conforme al tiempo de la conexión tienen un acceso físico en la red, comúnmente los proveedores de acceso a internet asignan una IP a estos equipos, dicha conexión es variable la cual puede cambiar de unas conexiones a otras totalmente distintas.

2.4.5. Programa Servidor

Es aquel que deberá de estar ejecutándose dentro del equipo servidor para que este proporcione el servicio. Uno de los programas más usados es Apache, programa el cual pertenece al código libre (Open Source). Se conocen otros programas como lo es personal web server (PWS) a su vez otros servicios como lo son el correo electrónico o la transferencia de ficheros deberán contener los programas correspondientes del servidor.

2.4.6. Programa cliente

El equipo cliente requiere el uso de un software para tener acceso al servicio, algunos ejemplos son los navegadores de internet como: Internet Explorer, Mozilla los cuales representan a los clientes web, por otra parte, programas como Outlook es ejemplo de cliente de correo electrónico y para un cliente FTP los programas WS_FTP o CuteFTP son claro ejemplo de estos.

2.5. Página web

En (Powell, 2001) el concepto de página web es la parte más fundamental de un sitio web, en su forma más básica, una página web es lo que se muestra en las ventanas de un navegador por lo que, una página web es igual a una dirección URL.

Actualmente una dirección URL muestra varios documentos dentro de la ventana del navegador por lo que a su vez una página web está dividida en varias ventanas más pequeñas conocidas como frames. Una página web se puede almacenar en equipos de manera local o en servidores donde dentro de estos se restringe el acceso a redes privadas.

2.5.1. Tipos de páginas web

Las diferentes formas de clasificar a las páginas web son numerosas, esto se define con base a la función que realizan, se imprimen o se observan de manera online, a su vez se pueden

clasificar mediante páginas generadas dinámicamente y estáticas.

Por lo que la necesidad de clasificar estas páginas en diferentes tipos permite a los diseñadores tener un análisis más concreto y una fácil clasificación para páginas de entrada, navegación, contenido y salida.

2.5.2. Página web de entrada

Cualquier página de un sitio web puede hacer la función de entrada por lo que si el usuario tiene conocimiento de la URL de la página esta hará la función de entrada del sitio.

2.5.3. Página de bienvenida

Las páginas web de bienvenida son utilizadas como su nombre lo indica para dar la bienvenida al sitio web para cualquier usuario y de esta manera dar una impresión concreta del sitio, en algunas ocasiones a la página de bienvenida como un atractivo del sitio mostrando disposiciones gráficas y sonido, de esta manera se pueden implementar acciones para dar a conocer acerca de instrucciones de instalación de herramientas para el usuario y que este haga uso correcto del sitio o en su caso saber si es que el usuario cuenta ya con las herramientas necesarias.

2.5.4. Página web principal

Como se ha mencionado con anterioridad (Powell, 2001) la página principal es la primera página que el usuario observa al ingresar al sitio web, dando como referencia la calidad del sitio y los temas que se abordan, como página de referencia deberá tener rasgos característicos que la diferencien de las demás páginas.

2.5.5. Diseño de páginas web

Antes de comenzar a crear cualquier página web, así como lo menciona (Powell, 2001), se

debe de tomar en cuenta aspectos específicos de estas como lo son: Identificar audiencia, Evaluar el contenido y Diseñar la estructura.

La diferencia entre un buen contenido y una interfaz decorativa se define en que se incline a que se alcance el objetivo, por lo que se tiene que definir de manera concreta para que este se cumpla, por lo que es imposible saber el alcance de una página web sin haber definido un objetivo con anterioridad. Ya que el diseño web es sumamente disciplinario por lo que campos como las artes visuales, tecnología, finalidad y motivo son buenas ideas para organizar los diseños de gráficas o aplicaciones de software de las páginas.

2.6.Lenguaje de programación

Un lenguaje de programación se define como: “El conjunto de caracteres y reglas, instrucciones que permiten la realización de acciones” como lo menciona (Jiménez, 2022), Por lo que un lenguaje de programación contiene una sintaxis (forma de ser escrito) que lo diferencien de los demás tipos de lenguajes, a su vez contienen bloques y agrupaciones de reglas las cuales son específicas dando razón al lenguaje, esto es conocido como semántica.

2.6.1. JavaScript

JavaScript es un lenguaje de programación para ordenadores, se utiliza en la creación de sitios web para interactuar con los usuarios permitiendo una interacción más proactiva mostrando animaciones, fecha etc. Se especializa principalmente en apoyar a los desarrolladores web para la interacción entre la página web y el navegador, como su nombre lo indica JavaScript está basado en el lenguaje de programación Java, aunque no al cien por ciento si no es que al contar con estructuras de sintaxis parecidas no es considerada una versión de Java sino un lenguaje independiente siendo más útil en el desarrollo de entornos web (Dimes, 2015).

2.6.2. HTML

Las siglas HTML son las siglas designadas para definir el lenguaje de “Marcas de Hipertexto”,

Es un lenguaje basado en marcas que contiene el navegador web por lo que los archivos de texto tipo HTML no usan formatos y solo contiene código simple que se denomina “Etiquetas”, a su vez (Dimes, 2015) menciona que los archivos generados por este lenguaje siempre contendrán la extensión de .html.

2.6.2.1. Creación de archivos HTML

De acuerdo con (Mcfedries, 1996) la creación de un documento desarrollado en HTML se lleva a cabo mediante un editor de texto, así como el Bloc de Notas para la creación y una vez creado es necesario un intérprete para mostrar lo redactado en el documento con anterioridad por lo cual se hace uso de los navegadores. Por lo que los pasos para la creación de estas necesitan un orden específico para evitar complicaciones en cuestión al orden, el orden se define en:

- Diseñar la estructura de la página
- Agregar título.
- Ingresar texto y párrafos.
- Ingresar formato y encabezado.
- Agregar imágenes y vínculos.
- Publicar página.

2.6.3. CSS

Según (Soria, 2014) Las siglas CSS tienen el significado de “Cascade Style Sheet” o como se entendería en español “Hojas de Exilo en Cascada” ya que este lenguaje permite otorgar atributos a los documentos que están hechos en HTML a su vez CSS permite separar los diseños y los contenidos, debido a esto se puede comparar a CSS con HTML más sin embargo CSS tiene una gran gama para poder personalizar una página web con más facilidad.

2.6.4. PHP

PHP es un lenguaje tipo script el cual es ejecutado por el lado del servidor, este lenguaje se

encuentra dentro del apartado principal de la creación de una página web relanzada en HTML. PHP que a diferencia del lenguaje de JavaScript el cual es ejecutado desde el cliente (navegador), PHP es ejecutado en el servidor, por lo que el resultado se refleja dentro de la página de HTML.

A su vez cuando es necesaria la consulta de algún archivo PHP, se procesa directamente en el servidor, una vez se complete el proceso se manda el resultado en código de PHP directo a la página y este es reenviado al navegador. Sin embargo, PHP es un lenguaje que fue diseñado con la intención total para el ámbito web por lo que las características que contiene son específicas para este ambiente (Oliver, 2016).

2.6.5. XHTML

XHTML es un lenguaje de programación de marcas, la creación de XHTML como lo menciona (Oros, 2010) no es otra más que sustituir a HTML ya que las funciones con las que cuenta HTML son idénticas a las que maneja XHTML.

La finalidad de XHTML no es otra más que continuar con el proyecto W3C (Word Wide Web Consortium) el cual tiene como propósito la creación de páginas donde maneje por separado la información de la página y la manera en la que se muestra al usuario, por lo que se define que XHTML es un lenguaje semántico.

XHTML tiene la característica de transmitir información de un documento sin tomar en cuenta el aspecto y el diseño, ya que estos son específicamente dejados a las hojas de estilo (CSS) por lo que también las funcionalidades de la página son dejadas a JavaScript.

2.7. Firebase

Se menciona en (Tomás, J. ,2019) que Firebase es una de las nuevas plataformas de Google que permiten el trabajo en la nube, esta permite el almacenamiento de datos y gestión de Backend considera que al ser una herramienta de Google la hace más confiable y segura, además de ser compatible con otros lenguajes de programación. Lo que en la actualidad haría que las aplicaciones trabajen en la nube toda la información que se maneja de forma sencilla e integrada.

2.7.1. Mobile Backend as a Service (MBaaS)

Firestore se cataloga como una plataforma tipo mobile backend la cual llama la atención porque proporciona servicios en la nube, así como la posibilidad de gestionar una consola web, los MBaaS están enfocados para el desarrollo de aplicaciones web y de escritorio, del cual no solo enfocándose en móviles también adopta las siguientes siglas “BaaS” es una opción que ofrece todas las infraestructuras de acceso y de almacenamiento y seguridad de los datos. En (Tomás, J. ,2019) se menciona que actualmente existen varios BaaS entre ellos las principales empresas tecnológicas como Apple, Amazon, Google Microsoft tienen plataformas con este propósito, como lo son:

- AWS Mobile (Amazon)
- Azure (Microsoft)
- CloudKit (Apple)
- Parse (Facebook)

Se muestra en la Tabla 1 una comparación entre los diferentes BaaS.

Tabla 1 Comparación entre Baas. Fuente Tomás,J. (2019)

	Firestore	AWS	CloudKit	Azure	Parse
Empresa	Google	Amazon	Apple	Microsoft	Facebook
Código abierto	no	no	no	no	sí
API nativo	Android iOS Unity JavaScript REST C	REST (Java JavaScript Perl PHP Python Ruby)	iOS Mac	Android iOS Windows Mac	Android iOS Unity Ja- vaScript REST C Arduino .net PHP
Bases de datos	NoSQL	NoSQL /SQL	¿?	NoSQL /SQL	¿?

2.7.2. Los servicios de Firestore

Firestore integra distintos servicios en una sola plataforma las cuales se muestran a continuación.

2.7.3. Herramientas de desarrollo

Las herramientas de trabajo con las que cuenta Firebase son extensas comenzando por:

- **Authentication**

Según (Tomás, J. ,2019) Firebase resuelve el problema de autenticación de manera que está basada en correo y contraseña para iniciar sesión, así como la opción de poder iniciar sesión por medio de una cuenta de Google, Facebook, Github y gestionar los usuarios desde el backend.

- **Realtime Database**

En este caso el autor menciona que no se necesita crear un servidor con las bases de datos de manera que se accede por medio de servicios Web, ya que está fundada en NoSQL así que, se estructura en una estructura muy similar al de JSON, y además de esto son bases de datos en tiempo real la ventaja de esta herramienta es que si el usuario se quedara sin internet se podrá guardar en el caché para almacenar los datos ingresados siendo así que en el momento que el usuario tenga internet de nuevo los datos del caché.

- **Cloud functions**

Desde cloud functions se tiene acceso a todos los servicios de Firebase como lo son las bases de datos, notificaciones, se menciona que este código se ejecutara en un entorno de Node.js seguro y administrado.

- **Cloud Storage**

Permite almacenar ficheros con un máximo nivel de disponibilidad, el autor comenta que se aprovechan las ventajas de la autenticación para dar los permisos adecuados y garantiza una mínima latencia, incluso si se dispara el número de accesos simultáneos.

2.7.4. Herramientas de análisis e interacción de usuarios con Firebase

- **Google Analytics**

-

Esta herramienta analiza toda la información relacionada con la app o en su caso la plataforma web y así poder tomar mediciones y apoyar a la mejor toma de decisiones, Analytics integra otras herramientas como lo son AsWords, AdMob.

2.7.5. Herramientas de comprobación

Las herramientas están conformadas por:

- **Crashlytics**

El autor menciona que es una herramienta la cual descubre problemas que puedan ocurrir para cualquier dispositivo lo que quiere decir es que identifica diferentes tipos de dispositivos y poder corregir lo más pronto posible, además de que genera informes de fallos en tiempo real.

- **Performance Monitoring**

Esta descubre si los usuarios perciben fluida la plataforma o app obtiene estadísticas de rendimiento personalizadas o automáticas todo esto desde la consola, se podrán realizar los análisis de los resultados y así poder detectar un problema desde el origen.

2.8. Base de datos

Las bases de datos son las más adecuadas para almacenar datos según Pulido, Escobar, Núñez, 2019). Se considera que es debido a que tiene diversas características, las bases de datos son importantes para la seguridad y resguardo de datos recuperación a posibles fallos.

2.8.1. Antecedentes de las bases de datos

Las bases de datos en la actualidad son un elemento básico y fundamental en la informática, se menciona en (Pulido, Escobar, Núñez, 2019) son de utilidad para toda disciplina o área que tenga la necesidad de gestionar datos siendo de importancia para la recepción de datos día a día.

2.8.2. Origen de las bases de datos

Los orígenes de las bases de datos se remontan desde la antigüedad, considerando bibliotecas en donde ya existían todo tipo de registros, otro de los ejemplos más notables es la recolección de datos de los censos de población que como se sabe la búsqueda de esta era lenta y poco eficaz, ya que no se contaba con la tecnología para la automatización de estas búsquedas.

Por lo que en necesidad de almacenar grandes cantidades de información se desarrolló el uso de las bases de datos, es así como en 1884 Herman Hollerith crea la primera máquina automática de tarjetas por lo que como en esas épocas los censos se hacían de manera manual Hollerith trabajó en la máquina tabuladora o censadora lo que permite en 1890 sistemas mecánicos para poder procesar información y tabular resultados.

Para el siglo XXI ya tres grandes empresas lideraban el mercado de las bases de datos como lo son:

IBM, Oracle, Microsoft y por parte de Google se encarga de gestionar una gran parte de información en su campo que es el internet, actualmente ya existe una gran variedad de software que permiten la creación de bases de datos según la necesidad

Hoy en día los sistemas de bases de datos pasan por un proceso de actualización para la adaptación de nuevas tecnologías como lo son bases de datos en la nube y enfocándose más a web (Pulido, Escobar, Núñez, 2019).

2.8.3. El Data minig en la actualidad

Data mining o minería de datos es un conjunto de técnicas que de manera automática permiten explorar grandes cantidades de datos con el fin de encontrar patrones, tendencias o reglas que explique cierto comportamiento de datos.

2.9.Bases de datos no relacionales

Una base de datos no relacional según (Microsotf) se define como aquella que prescinde del esquema tabular de filas y columnas típicamente empleado en la mayoría de los sistemas de bases de datos tradicionales. En su lugar, estas bases de datos adoptan un modelo de almacenamiento optimizado según los requisitos específicos del tipo de datos almacenado.

2.9.1. Definición de NoSQL

Se refiere a tipos de bases de datos no relacionales que almacenan datos en un formato distinto a las tablas relacionales.

¿Para qué utilizar una base de datos NoSQL?

Las bases de datos NoSQL gozan de una amplia implementación en aplicaciones de tiempo real y entornos de big data, destacándose por sus notables niveles de escalabilidad y disponibilidad.

Estas bases de datos son altamente preferidas por los desarrolladores debido a su inherente propensión al desarrollo ágil, permitiendo una adaptación rápida a los requisitos en constante evolución. Su capacidad para almacenar datos de manera más intuitiva y alineada con la estructura de datos utilizada por las aplicaciones y minimiza la necesidad de transformaciones extensas durante el almacenamiento o recuperación de datos mediante interfaces API NoSQL. Además, las bases de datos NoSQL son altamente eficientes en la nube, mitigando por completo el tiempo de inactividad. (Oracle, 2023)

2.9.2. ¿Cuándo elegir una base de datos no relacional?

Según la empresa (Oracle,2023) la elección de una base de datos NoSQL se justifica en entornos empresariales y organizacionales donde la necesidad de innovación rápida es imperativa. La capacidad para mantener la agilidad operativa a escala es fundamental. Las bases de datos NoSQL se destacan por ofrecer esquemas flexibles y admitir diversos modelos de datos, siendo especialmente idóneas para el desarrollo de aplicaciones que demandan gestionar grandes volúmenes de datos con bajos tiempos de respuesta o latencia. Ejemplos de tales aplicaciones incluyen juegos en línea y plataformas de comercio electrónico en la web.

2.9.3. Ventajas de una base de datos no relacional

(Oracle,2023) menciona que entre sus ventajas se encuentra la notable velocidad y escalabilidad en la interacción digital y el consumo de datos, observados en las últimas dos décadas, han obligado a las empresas a adoptar un enfoque más moderno y dinámico en la gestión y acceso a los datos. En un contexto en el cual los usuarios a nivel global demandan un flujo ininterrumpido de contenidos y funcionalidades, la adaptación veloz de las bases de datos resulta comprensible. En este sentido, se presentan a continuación algunos de los motivos fundamentales por los cuales los desarrolladores optan por bases de datos NoSQL o no relacionales.

- Flexibilidad

La flexibilidad inherente a las bases de datos NoSQL contrasta con la estructura más rígida y predefinida de los datos en las bases de datos SQL. En el contexto de NoSQL, los datos pueden ser almacenados de manera más libre, sin la restricción impuesta por esquemas inflexibles. Este diseño fomenta la innovación y el desarrollo ágil de aplicaciones, permitiendo a los desarrolladores enfocarse en la creación de sistemas que mejoren el servicio al cliente sin la necesidad de preocuparse por esquemas predeterminados. Las bases de datos NoSQL tienen

la capacidad de gestionar sin dificultades diversos formatos de datos, incluyendo estructurados, semiestructurados y no estructurados, todo dentro de un único almacén de datos.

- Escalabilidad

En vez de optar por la escalabilidad vertical mediante la adición de servidores adicionales, las bases de datos NoSQL tienen la capacidad de escalar horizontalmente mediante la utilización de hardware básico. Este enfoque permite manejar mayores volúmenes de tráfico para cumplir con la demanda sin experimentar períodos de inactividad. El escalado horizontal posibilita que las bases de datos NoSQL crezcan en tamaño y potencia, convirtiéndose así en la elección preferida para la expansión de conjuntos de datos.

- Alto rendimiento

La arquitectura con escalabilidad horizontal de las bases de datos NoSQL se vuelve especialmente beneficiosa al incrementarse el volumen de datos o el tráfico.

Como se puede ver en el siguiente grafico representado en la Figura 3 la arquitectura muestra que garantiza tiempos de respuesta rápidos por esta razón las bases de datos no relacionales se utilizan en apps que recopilan información de datos todos los días a todo momento para que el usuario pueda tener una experiencia interactiva.

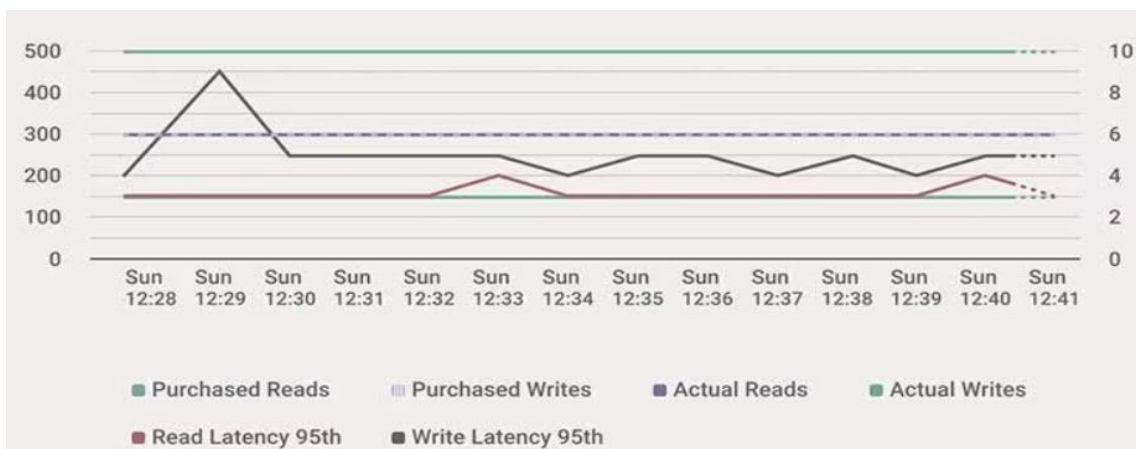


Figura. 3. Graphical NoSQL Database Cloud Service – Real Time Throughput. Fuente. Oracle 2023.

- Disponibilidad

Las bases de datos NoSQL implementan la replicación automática de datos en varios servidores, centros de datos o recursos en la nube, lo que resulta en una minimización de la latencia para los usuarios, independientemente de su ubicación geográfica. Esta funcionalidad no solo contribuye a la reducción de la carga asociada a la gestión de bases de datos, sino que también libera tiempo para enfocarse en otras prioridades.

- **Altamente funcional**

Las bases de datos NoSQL están concebidas para almacenes de datos distribuidos que demandan una capacidad de almacenamiento considerable. Esta característica posiciona a NoSQL como la opción predilecta para entornos de big data y aplicaciones web en tiempo real. (Oracle,2023)

2.10. Características de las bases de datos

Las bases de datos cuentan con tres puntos fundamentales para que se le considere base de datos la primera es:

- Independencia de los datos lo que significa que al existir los datos no dependen de ningún programa cualquier aplicación que necesite hacer uso de esos datos ya existentes se adapta
- Reducción de la redundancia: lo que se refiere a reducir la duplicidad de datos cuando esto se consigue se evitan inconsistencias entre los datos.
- Seguridad: En una base de datos se necesita la seguridad de datos, así como el manejo de los datos frente a usuarios autorizados y no autorizados.

2.10.1. Integridad

La integridad se conforma por la validez la consistencia de datos se compone de reglas que no se pueden violar, reglas que están establecidas que se aplican a los datos como a sus relaciones.

Esto ya definido, aunque es posible que se generen errores en los registros por el usuario

generando variaciones entre los datos la manera de poder evitar estos errores es que al crear una base de datos prestar atención a la integridad para lograr que un usuario se equivoque lo menos posible, un ejemplo de esto sería que si en un campo solo es para ingresar número, que solo se permita ingresar números sin tener la posibilidad de ingresar algún otro carácter diferente, finalmente mantener la integridad es asegurarse que los datos permanezcan intactos y sin cambios.

2.10.2. Redundancia

La redundancia se refiere al almacenamiento de los mismos datos varias veces en diferentes lugares lo que puede provocar problemas de espacio de almacenamiento, lo que provoca que en la búsqueda no haya datos precisos.

2.11. Procesos operativos

Según (Pilar, 2005) los procesos operativos son aquellos que transforman recursos para poder elaborar el producto o servicio de esta manera se aporta un alto valor a estos procesos, aunque estos procesos no pueden funcionar solos, si no que necesitan forzosamente de los recursos para cumplir con su objetivo.

2.11.1. Inventario

El inventario o Stock es aquel que lleva las existencias o cantidades de un sistema en producción, así como lo expresa (duda, 2018) contiene para su venta o utilización de producción, por lo que el objetivo que contiene un sistema de gestión de inventariado no es más que controlar los niveles que hay en el inventario y así determina cuáles niveles son los que deben de mantenerse, reposición de existencias etc., por lo que un inventario se clasifica de las siguientes maneras:

- Materias Primas: los cuales son materiales provenientes de proveedores externos y son utilizaos para la elaboración de los productos.

- Componentes: Son las partes que se utilizan para el proceso de ensamblaje de la empresa.
- Materiales en proceso: Aquí se encuentran los materiales que están cursando el proceso de producción.
- Productos terminados: Como su nombre lo indica aquí se encuentran los productos cuya elaboración ha sido culminada exitosamente. La finalidad del análisis de un inventario es saber cuándo se deben de pedir, ordenar los artículos dependiendo del lote esto se debe a:
- Stocks de seguridad: Si la demanda del producto es alta se deberá de contar con la producción pertinente dentro del alcance en el inventario, sin embargo, es necesario tener una reserva de seguridad ante cualquier imprevisto siendo de esta manera un seguro de tiempo para contar con más en Stock.
- Desequilibrios entre sectores de producción: Los desequilibrios que llegan a ver dentro de cada sector son captados por los inventarios facilitando y equilibrando esos tiempos.
- Producción por lotes: Cuando se llevan cambios de herramientas se opta por la creación de productos en lotes más extensos maximizando el uso de las máquinas.
- Producción nivelada: Si la demanda de la producción tiene un nivel acorde a la producción del momento, se opta por adelantar productos terminados y resguardarlos para un uso posterior.
- Compras de mayor tamaño: Si el precio por transporte de material prima va por unidad, las compras extensas generan un menor costo de unidad.

Por otra parte, como lo argumenta (Guevara, 2020) Inventario es la conexión que ay entre los bienes que una organización contiene, por lo que estos son comúnmente clasificados en familias categorías.

Las organizaciones tienen el deber de realizar un inventario ya que es de suma importancia que este se apegue a la realidad, sin embargo, si el inventariado no se acerca a la realidad y se colocan las cantidades mal, se corre el riesgo de tener una sobrevaloración la cual indicara que se tiene más o menos materia prima para el trabajo.

2.11.2. Stock

El Stock es el conjunto de existencias que contiene una organización ya sea materia prima de reserva o en su caso productos para venta.

Las funciones del Stock son 3 básicas las cuales son:

- Reguladora: Si el conocimiento del mercado es limitado la organización opta por hacer frente a posibles incrementos de ventas.
- Comercial: Facilita a la empresa los plazos de entrega en tiempo y forma.
- Económica: Si la empresa compra grandes cantidades de materia prima esta la posibilidad de manejar un mejor precio de compra (Guevara, 2020).

2.11.3. Control de inventarios

El control de inventarios se basa en el punto de reorden que, a su vez conforme a ventas, planeación de la producción y determinación de la distribución se busca maximizar las ventas a un punto donde sea una inversión ahorrativa para el inventario de las organizaciones ya que la mayor es resultado del control, pero a su vez también lo es un menor desperdicio o pérdida de materia prima (Taha, 1995).

2.12. MARCO METODOLÓGICO

En este apartado se detalla la realización del marco metodológico mencionando el modelo que fue elegido para el desarrollo del proyecto. El método utilizado para desarrollar el proyecto es el de programación extrema (XP).

En (Laínez, 2014) menciona que la comunicación entre el cliente y el equipo de desarrollo permite que ambas partes puedan dar seguimiento a detalles del proyecto.

Entre las buena practicas se encuentran:

- **Cliente presente**

Se considera que el cliente no necesita o incluso no debe estar presente durante el proceso de desarrollo, sin embargo, lo que busca la XP es precisamente hacer que la presencia del cliente si sea de vital importancia para el buen seguimiento y éxito del proyecto ya que es importante que el cliente participe activamente en el proceso del proyecto.

- **Planificación**

Importante punto de partida puesto que en el inicio de cada interacción el cliente debe ser invitado a describir los requisitos funcionales de acuerdo con sus necesidades y que estas funciones estén presentes en el sistema.

- **Desarrollo guiado por las pruebas**

Responsabilidad del desarrollador implementar las pruebas en las funcionalidades antes de codificar como realizar el diseño o diagramado.

- **Código estandarizado**

Siguiendo estas prácticas según (Laínez, 2014) los desarrolladores deben tener acceso al código en todo momento y pueden cambiar y actualizar aquello que se considere importante de modificar.

Por lo que, para la ejecución de este proyecto, se adopta la metodología de programación extrema (Extreme Programming, XP), disciplina perteneciente al ámbito de desarrollo ágil de software. Esta metodología se fundamenta en valores clave como la simplicidad, la comunicación efectiva, la retroalimentación constante, la valentía y el respeto. La eficacia de esta metodología radica en la integración de todo el equipo de desarrollo, involucrándolos en prácticas simples y proporcionando una retroalimentación adecuada. Este enfoque facilita la

adaptación de las prácticas a la situación específica del proyecto (Extreme Programming, 2021).

Esta metodología prioriza la práctica del desarrollo en parejas o grupos, involucrando activamente al cliente. Se enfoca en adoptar un diseño simplificado, realizar pruebas continuas del código y adherirse a un estilo de programación definido. El objetivo es lograr que el código sea comprensible para todos los miembros del equipo y permitir mejoras cuando sea necesario. Además, se aplican otras prácticas fundamentales en este enfoque.

El presente proyecto ofrece una introducción general a la metodología mencionada, resaltando sus áreas de aplicación primordiales, incluyendo su aplicabilidad en la enseñanza de la programación.

La integración de la programación extrema con los procedimientos de ejecución que buscamos desarrollar e implementar para el proyecto se manifiesta mediante las variables de ambiente concebidas por este modelo. Estas variables desempeñan un papel esencial en el desarrollo, particularmente en la fase de desarrollo y mantenimiento, alineándose de manera coherente con la visión inicial del proyecto (Programación Extrema, 2022)

Dentro de las Variables de la Programación Extrema se identifican las siguientes:

Mencionando que incluye cuatro variables para cualquier proyecto de software

- **Coste:** Alcance monetario del proyecto.
- **Tiempo:** Estimación y Fijación del Tiempo requerido para el desarrollo.
- **Calidad:** Cumplimiento de requerimientos a escala medible.
- **Alcance:** Futuras Implementaciones o crecimiento del Proyecto.

2.12.1. ETAPAS DE DESARROLLO DE LA PROGRAMACIÓN EXTREMA

- **Exploración:** Se examina el entorno, se conforma el equipo de trabajo valorando las habilidades de los miembros del equipo y se examinarán las tecnologías potencias que se necesitarán para desarrollar el sistema. El objetivo es lograr que el cliente refine un documento para poder determinar con eficiencia el tiempo que tomará construir la solución

- **Planeación:** Se establece una fecha en común acuerdo con el cliente para la entrega de la solución de los problemas más urgentes. Esta etapa debería durar unos pocos días.
- **Desarrollo:** Consta de iteraciones a la primera versión (ciclos de prueba, retroalimentación y cambios) dura aproximadamente tres semanas. Celebre siempre los avances por pequeños que sean pues forma parte de la cultura de motivación al grupo para que pongan todo su empeño en el desarrollo del proyecto. La filosofía que anima la programación extrema va más allá de la planeación y la administración de un proyecto de sistemas con métodos extremos, también tiene como base los valores y los principios.
- **Mantenimiento:** Es considerado como el estado normal del proyecto XP. Se pueden agregar nuevas características, se toman en cuenta las sugerencias más arriesgadas del cliente y se pueden cambiar o agregar nuevos miembros. (Extreme Programming, 2021).

Al considerar el que consiste esta metodología se representan las actividades a desarrollar por etapas considerando técnicas, herramientas y metas como se presenta en la tabla 2 del cuadro metodológico.

Tabla 2. Cuadro metodológico.

Marco metodológico			
Actividades	Técnicas	Herramientas	Metas
Planteamiento de problema	Análisis	Microsoft Word	Recolectar información de la empresa sobre la problemática a resolver para comenzar el desarrollo.
Análisis: 1. Objetivo general 2. Objetivos específicos 3. Justificación	- Investigación - Redacción - Observación - Encuestas	Microsoft Word	Demostrar la utilidad de la investigación así como mostrar un objetivo para la resolución el problema
Marcos: 1. Marco conceptual 2. Marco metodológico	- Investigación - Análisis - Recopilación bibliográfica - Definiciones - Tablas	Microsoft Word	Obtener una recopilación y análisis de información sobre así como definiciones técnicas para el entendimiento de palabras claves del proyecto.
Desarrollo: 1. Diseño 2. Maquetado 3. Desarrollo Front-end 4. Desarrollo Back-end 5. Pruebas	- Programación - Prototipado - Diseño - Diagramado	- PHP - CSS - JavaScript - Firebase - Creately para diagramar	Obtener planos de interfaces de diseños para una mejor comprensión, además de implementar programación adecuada y conexiones de bases de datos de acuerdo con el maquetado. Y a los requisitos.

CAPÍTULO III. DESARROLLO

3. Descripción General

En esta sección se detallan la perspectiva que tiene el proyecto, las funciones, así como las características de los usuarios, las restricciones y los futuros requisitos también, esto se detalla en cada uno de los puntos ya mencionados.

3.1.Perspectiva del producto

Se identificó el objetivo general y necesidades que tiene la empresa objetiva, así como que es lo que puede resolver el portal web para ayudar a satisfacer dicha necesidad.

La aplicación móvil para pisos de azulejo ofrecerá una amplia selección de diseños y colores para que los usuarios puedan encontrar el piso perfecto para cada espacio. Además, la función de visualización en realidad aumentada que permite a los usuarios ver cómo se vería el piso en su espacio antes de tomar una decisión de compra.

3.2.Funciones del producto

- Módulo de gestión de usuarios y roles.

Modulo que permite la creación de usuarios, permite la configuración de accesos limitados a registros específicos según el rol de usuario (administrador, operador). Por ningún motivo.

- Módulo de seguridad

Se controla el acceso por medio de contraseñas, por ningún motivo se puede hacer uso del sistema sin antes autenticarse, así el usuario tenga el URL de la página a la que quiera acceder el sistema pedirá autenticarse, control y cierre automático de sesiones inactivas después de 10 minutos para prevenir el acceso no autorizado en caso de olvido de cierre de sesión.

- Cifrado de datos

Datos almacenados en una base de datos para la protección de la información, como las compras, cantidades, y datos personales.

- Módulo de registro de cambios en el inventario

Registros detallados por medio de formularios almacenados en las bases de datos, se incluyen fechas de salidas para el control de salida de producto, así como quien fue el responsable de dicha salida.

- Módulo de gestión de existencias

Seguimiento en tiempo real de la existencia de cada producto, se describe la cantidad, ubicación y cuál es la cantidad mínima requerida mostrándose de color rojo como alerta.

- Módulo categorización y clasificación

Permite la clasificación de productos para que se puedan identificar de mejor manera, se categorizan también por productos en unidades de medida.

- Módulo de historial de compras

Mantiene un historial detallado de las compras que se hacen con el fin de mantener una transparencia y seguimiento de las compras.

- Módulo de gestión de entradas y salidas de productos

Permite contralorar quien realiza una salida o una entrada de productos detallando fechas y cantidades, lleva un seguimiento real de la actualización del inventario ya sea por compras, entradas o salidas, gestiona también las devoluciones de productos sobrantes actualizando directamente el inventario.

- Informe y/o reporte de inventario

Se genera un reporte sobre los productos que quiera consultar del inventario.

3.3.Características del usuario

En la tabla 3 se observa la descripción de las características de los usuarios involucrados en el sistema web.

Tabla 3 Características de usuarios

Usuarios	Características	Nivel educativo	Experiencia técnica
<i>Desarrollador</i>	Encargado de crear la estructura, diseño y comportamiento de las funciones y la interfaz. Utilizando herramientas tecnologías web necesarias. Integración de base de datos Configura y personaliza el contenido Encargado de la seguridad y protección de datos, Se encarga del soporte al usuario.	Nivel licenciatura/ingeniería.	Habilidades en herramientas tecnologías tales como: conocimiento en lenguajes de programación, gestión y manejo de bases de datos, es de datos, mantenimiento, así como conocimientos sobre resolución de problemas.
<i>Diseñador</i>	Encargado del análisis de requisitos, y traducirlos a diagramas UML. Diseña y crea diagramas UML para la comprensión de los requisitos funcionales del sistema	Nivel licenciatura/ingeniería.	Habilidades en herramientas de diseño UML e ingeniería de requerimientos.
<i>Gestor de proyecto</i>	Habilidades para tomar decisiones y planificar crear y gestionar ordenes de trabajo Gestiona recursos, Tiempo, Da seguimiento de control, Se encarga de la calidad contante.	Nivel licenciatura/ingeniería	experiencia en la gestión de proyectos, conocimientos en metodologías de desarrollo
<i>Usuario administrador</i>	Proactividad, habilidad de gestionar, comunicación, responsabilidad, uso efectivo del tiempo, toma de decisiones.	Encargado de área	Conocimiento previo del uso del sistema
<i>Usuario/cliente</i>	Usuario final tendrá como función agregar eliminar y editar sus compras, así como consultar el diseño del piso y generar un ticket.	No requerido	Conocimiento previo de uso del sistema

3.4. Restricciones

En este apartado se detallan e identifican las restricciones posibles en el uso del sistema de inventario web.

- Se identifican restricciones del uso del sistema a ciertos países o idiomas.
- Se Incluyen y se consideran las restricciones en una sección limitada del sistema o en los términos y condiciones de uso.
- El usuario deberá tener acceso a esta información de manera fácil y clara.

Políticas de la empresa:

- Tener en cuenta los requisitos legales y regulatorios en el sistema como lo son la ubicación, datos personales sean del usuario o de la empresa, es importante el cumplimiento de estas y se acople a como lo establezca la empresa ya que el manejo y uso de datos es fundamental por lo que se deberá asegurar que cumple con estos requisitos y respetar las políticas y regulaciones correspondientes.
- Adaptación al diseño de la empresa en cuanto a la interfaz de usuario.

Se toman en cuenta diseños hechos exclusivamente para el uso del sistema interno, siguiendo la misma estructura que tiene la empresa, considerando, logos, colores, e imagen.

- Cumplimiento de las políticas de seguridad.

Solo personal con una cuenta con la que se puedan autenticar podrá tener acceso al sistema.

- Se limita el uso del sistema solo a personal autorizado por la empresa;

Como lo son el encargado de almacén (administrador).

- Personal operativo (solo consulta, y registro de compras y salidas)

- Protocolos de comunicación.

Formatos de datos: Estructurar y codificar la información que se transmite. Los formatos para utilizar incluyen: JSON (JavaScript Object Notation), XML.

3.5.Suposiciones y dependencias

A continuación, se describe en general cada uno de los factores que afectan los requisitos declarados en el ERS, los siguientes puntos fueron considerados.

- Claridad en la definición de objetivos.
- Precisión en la planificación.
- Compromiso de los participantes.
- El presupuesto disponible.
- El tiempo de ejecución.

Por lo que los supuestos se hacen en un entorno establecido y definido como, por ejemplo;

- El sistema solo se ejecutará si la aplicación tiene una conexión a internet de lo contrario no cargara la imagen de inicio.

- Se asume que el usuario tiene o tendrá conocimientos básicos de cómo utilizar una computadora.

- Se asumen que los requisitos establecidos, durante el desarrollo del sistema sean estables y no muy cambiantes en su contexto.

- El algoritmo de autenticarse proporcionado es seguro y confiable.

- El registro se cumple de manera exitosa conforme a los requerimientos, se utiliza un usuario y contraseña para acceder ya previamente creado por el administrador de otra manera no accede al sistema.

- La barra de búsqueda para consultas en cada apartado proporciona la información correcta.

- Se registran datos de entradas y salidas de manera exitosa y segura para los usuarios.

- El sistema de inventario web requiere de un servidor en el cual almacena y procesa datos.

La infraestructura cloud donde se desplegará el sistema, deberá ofrecer toda su capacidad técnica y operativa para garantizar la disponibilidad de la información cuando se solicite.

- El sistema depende de una conexión a internet para acceder a ciertos servicios o funcionalidades.

Otros factores importantes de dependencia:

Se requiere constante actualización de datos, por la información proporcionada por parte de la empresa, ya que si no se mantiene comunicación constante entre la empresa y los desarrolladores el sistema no tendría actualización, por lo cual los datos a la vista del cliente final serían incorrectos.

Por lo que se toma en cuenta lo siguiente:

- 1.- Establecer un acuerdo de servicio con la empresa para que se encarguen de mantener y actualizar los datos, de esta manera los desarrolladores puedan aplicar las actualizaciones según sea necesario.
- 2.- Establecer posibles cronogramas de actualizaciones regulares con la empresa de manera que garantice que el sistema siempre se mantenga actualizado y funcionando correctamente.
- 3.- Establecer mecanismos de comunicación eficaces con la empresa para reportar cualquier problema o error que se detecte en el sistema.
- 4.- Proporcionar permisos específicos a la empresa (solo los necesarios para realizar las actualizaciones básicas), ningún permiso como desarrollador.
5. Hacer seguimiento regularmente para asegurarse de que se estén cumpliendo las condiciones planteadas.

3.6.Requisitos futuros

Esta subsección se detallan futuras mejoras al sistema, que podrán analizarse e implementarse en un futuro de ser necesario. Es importante tener en cuenta necesidades y funciones de los usuarios actuales para poder identificar sus características y otras funcionalidades que podrían ser útiles a futuro.

Se realizó una entrevista en la cual fue útil para evaluar los requisitos que se llevan a la programación final, así mismo se identificaron las restricciones posibles ya que estas pueden afectar al sistema, una vez definido esto los posibles requisitos futuros son:

- Implementar otra sección en el portal para otra administración, como la gestión de empleados exclusivamente para el área de RH.
- Agregar una función de un chat de soporte inmediato para resolver conflictos del sistema simples.
- Integrar alguna sección de tutoriales para el uso correcto del sistema.
- Gestión de fechas de caducidad para dar un seguimiento contralado sobre los productos con fechas de caducidad y mostrar alertas sobre productos próximos a vencer.
- Reserva de inventario para proyectos futuros.

3.7. Interfaces externas

El cliente/servidor del sistema de inventarios web utiliza PHP, la cual procesa solicitudes del cliente, procesa y realiza las operaciones en la base de datos alojada en Firebase Realtime Database como plataforma para el almacenamiento y sincronización de datos.

A continuación, se presenta una descripción de cómo se utiliza esta interfaz externa:

- **Conexión a Firebase**

El cliente del sistema se conecta al servidor de Firebase a través de las API's proporcionadas por la plataforma. Esto se logra mediante la inclusión de las bibliotecas y configuraciones necesarias en el código del cliente.

Para especificar a profundidad las interfaces externas que se necesitaron se explican a continuación:

- **Sublime Text**

Es un editor de texto el cual es caracterizado por sus amplias funcionalidades al momento de desarrollar software, tiene incorporado la estructura en la sintaxis de diferentes lenguajes de programación con una interfaz amigable, en el desarrollo del Sistema de información web con PHP y Firebase para la gestión y control de inventario de pinturas para CC Camaleón Color S.A. de C.V. se trabajó con esta herramienta de marcado ya que indica en donde está ubicado los errores de sintaxis y la codificación se maneja de forma dinámica teniendo en cuenta que se requieren de paquetes adicionales.

Sublime Text está disponible para OS X, Windows y Linux. Una licencia es todo lo que necesita para utilizar Sublime Text en todos los equipos de su propiedad, sin importar el sistema operativo que utiliza, también ofrece una licencia gratuita de prueba. Usa un conjunto de herramientas de interfaz de usuario personalizada, optimizado para la velocidad y la belleza, aprovechando al mismo tiempo la funcionalidad nativa en cada plataforma.

La versión usada para el desarrollo del proyecto es la versión gratuita, el cual funciona con una publicidad de compra del producto al momento de guardar los cambios realizados en el código, por lo tanto, no es una limitante para el desarrollo del sistema.

- **XAMPP**

XAMPP es un servidor web local multiplataforma que permite la creación y prueba de páginas web u otros elementos de programación. Sin embargo, XAPP integra una serie de herramientas que potencian y facilitan la experiencia al desarrollador. Por otra parte, XAAP cuenta con licencia gratuita de uso, ya que al ser open source no dispone de un precio al público.

Además, permite trabajar en un servidor local sin necesidad de tener conexión a Internet. Esto significa que se puede hacer uso de la computadora personal y trabajar en proyectos, aunque no se disponga de una conexión de ancho de banda. Por lo que al desarrollar el sistema web dio grandes beneficios y fácil acceso a configuraciones y desarrollo nuevo de funcionalidades gracias a que no es necesaria una conexión a internet.

- **Base de Datos (Firebase)**

En el desarrollo de sistemas web, la elección de la base de datos desempeña un papel crucial en el rendimiento, la escalabilidad y la eficiencia del sistema. Firebase, una plataforma de desarrollo de aplicaciones móviles y web la cual ofrece una base de datos no relacional que ha ganado popularidad debido a sus características únicas y beneficios.

Por lo que la aplicación de esta herramienta en el “Sistema de información web con PHP y Firebase para la gestión y control de inventario de pinturas para CC Camaleón Color S.A. de C.V.” ofrece un ámbito más flexible en cuanto a reglas de seguridad robustas y personalizables para garantizar la protección de los datos e integridad del sistema, A su vez la facilidad de uso de que provee para plataformas web simplifica el desarrollo y gestión del proyecto dentro de la plataforma.

Firebase al contener una base de datos no relacional no requiere un esquema fijo de datos permitiendo agregar, modificar y eliminar campos sin afectar la integridad de los datos existentes por lo que brinda beneficios significativos como el desarrollo, rendimiento y mantenimiento del sistema web.

- **DevTools de Google Chrome**

Chrome DevTools es un conjunto de herramientas para desarrolladores web integrado en el navegador Google Chrome. Con estas herramientas puede inspeccionar el HTML renderizado (DOM) y la actividad de red de sus páginas. Puede utilizar DevTools para solucionar problemas relacionados con el servicio de anuncios. Permite identificar puntos esenciales en errores de sintaxis y paquetería dentro del desarrollo del sistema que no podían ser visualizados por el IDE de desarrollo.

Las opciones son las que se encuentran en la barra de arriba del todo de la pantalla de devtools.

Las opciones se resumen en:

- Revisar código fuente
- Modo móvil o dispositivo
- Pestaña Elements

- Pestaña console
- Pestaña source
- Pestaña Network
- Pestaña performance
- Pestaña memory
- Pestaña application
- Pestaña security
- Pestaña Audits

Lenguajes de programación utilizados para el desarrollo:

Es esta sección se procede a explicar los lenguajes de programación empleados para el desarrollo del sistema:

- Java Script

Java Script es un lenguaje de programación de computadoras dinámico utilizado habitualmente en navegadores web para controlar el comportamiento de páginas web e interactuar con los usuarios. Permite comunicación asíncrona y puede actualizar partes de página web o incluso remplazar completamente su contenido.

Java Script se basa vagamente en el lenguaje de programación de java a pesar de tener una metodología de programación y sintaxis similar no puede ser considerada una versión ligera de java, sino que es un lenguaje completamente para navegadores el cual es compatible con todos los navegadores modernos y se ejecuta del lado del cliente, lo que significa que se ejecuta en el navegador web del usuario final. Además de su uso en el desarrollo web, Java Script también se utiliza en aplicaciones de servidor (con tecnologías como Node.js) y en el desarrollo de aplicaciones móviles y de escritorio. Su gran diversidad de funcionalidades se implementó en cuestión a animaciones y manejo de información en formularios del sistema.

- **HTML**

HTML es un lenguaje de programación que utiliza una serie de códigos llamados etiquetas que van definiendo los elementos que componen una página web: texto, Imágenes, etc. Esas etiquetas serán interpretadas por un programa navegador de internet que mostrada adecuadamente la página web al usuario.

A su vez se cuenta con HTML5 el cual no se limita solamente a interpretar etiquetas básicas, si no que este permite involucrar funciones como lo son CSS, CSS3 y Java Script entre otros que harán el proceso más complejo y detallado para el usuario y con estas se pueden crear páginas web más sofisticadas y útiles. Para la implementación del sistema web de inventarios el ocupar estas nuevas funciones permitieron agregar funcionalidades de animaciones y nuevas formas de procesamiento de información para un página más sofisticada y atractiva para el usuario final.

- **PHP (Hypertext Preprocessor)**

PHP es un lenguaje de programación de alto nivel que se ejecuta y es interpretado en el servidor. Se puede encontrar este lenguaje en páginas reconocidas como lo son: Facebook, Instagram, Wordpress o Wikipedia. Una de sus principales ventajas es que este lenguaje es ejecutado en donde están alojadas las páginas al contrario de otros lenguajes que son interpretados por el navegador, ya que al ser ejecutado por el servidor todas las páginas que contengan van a poder ser visualizadas en cualquier ordenador, independientemente del navegador que tenga.

El problema de que las instrucciones sean ejecutadas por el navegador es que en varias ocasiones este no puede entenderlas por completo causando una falta de visualización al resultado esperado.

Por lo que al implementarlo se evitó que hubiera fallos dentro del entendimiento de las instrucciones proporcionadas para el sistema web, facilitando las visualizaciones y actualizaciones de instrucciones de codificación.

REQUERIMIENTOS TÉCNICOS MINIMOS.

Requerimientos de Hardware:

- Procesador: Intel Procesador Core i5 // AMD - RYZEN 5 en adelante
- Memoria RAM: 8GB mínimo.
- Disco Duro: 500 GB mínimo.

Requerimientos de Software:

- Sistema Operativo: Windows 10 Pro/11 Pro /MacOs.
- Navegador Web Chrome/Safari/MicrosoftEdge/Opera/Mozilla Firefox.

3.8.Funciones

Al realizar el levantamiento de requerimientos mediante el proceso de recopilación de información a través de entrevistas se hace un análisis de las necesidades y expectativas de los usuarios para el sistema teniendo así las siguientes funciones para su usabilidad.

3.8.1. Requisitos funcionales

Los requisitos funcionales identificados en las reuniones con la empresa se presentan a continuación.

- **Requisito funcional 1: Autenticar cuenta de usuario administrador**

El cliente deberá autenticar su cuenta utilizando los datos proporcionados durante el registro en la sección de inicio de sesión. Para acceder al portal web, deberá ingresar su correo electrónico y contraseña previamente creadas por el administrador.

- **Requisito funcional 2: Registrarse como nuevo usuario**

Se requiere que el usuario administrador cree una cuenta solo el administrador tendrá privilegios de crear o eliminar una nueva cuenta, podrá crear cuentas de administrador o usuario común, para el registro de un nuevo usuario deberán ser llenados todos los campos que

se requieran por el sistema, nombre completo, correo, contraseña, de lo contrario no procederá un registro exitoso.

- **Requisito funcional 3: Editar datos de usuario**

El usuario administrador tiene privilegios de editar datos de un usuario, podrá editar el nombre en caso de haber faltado datos, o si correo en caso de cambio de correo, así como la contraseña proporcionada.

- **Requisito funcional 4: Eliminar usuario**

El usuario administrador podrá eliminar a los usuarios, esto para mejor seguridad en caso de alguna baja de personal, no tenga acceso al sistema.

- **Requisito funcional 5: Registrar nuevo producto**

El usuario con acceso al sistema como administrador podrá agregar productos, deberá llenar un formulario proporcionando la información requerida por el sistema, sin dejar ningún campo vacío, llevará campos como el nombre de producto, categoría, proyecto para el cual se utiliza el producto fecha de entrada de registro proveedor y la ubicación en donde se encontrará asignado el producto.

- **Requisito funcional 6: Editar producto**

El usuario administrador tiene los privilegios de editar algún producto que hayan metido en entradas ya existentes se podrá modificar en sus campos existentes excepto en el campo de ID.

- **Requisito funcional 7: Eliminar producto**

Si el usuario desea eliminar un producto en caso de ya no manejar ese producto solo el administrador tendrá los permisos de hacerlo mismo que se verá actualizado en la BD

- **Requisito funcional 8: Agregar nueva compra**

En sistema tiene el módulo de agregar nueva compra en el cual usuario operativo y usuario administrador podrá registrar las compras que se hacen con numero de factura, fecha de compra, descripción.

- **Requisito funcional 9: Consulta de productos en el módulo de inventario**

El sistema cuenta con una barra de búsqueda en la sección de inventario en la cual se podrá buscar por proyecto, por nombre o por ID según prefiera el usuario administrador o usuario operativo.

- **Requisito funcional 10: Imprime y/o genera reporte de Excel del inventario actual**

Se podrá generar y descargar un reporte de las listas de productos registrados generando un archivo de Excel incluyendo todos los datos del inventario.

- **Requisito funcional 11: Registrar proveedores**

Usuario con privilegios de administrador puede agregar una lista de proveedores, en este módulo se requiere llenar datos obligatoriamente como lo son el nombre del proveedor, datos de contacto como dirección, teléfono, correo electrónico, sin dejar ningún campo vacío de lo contrario no permitirá el registro.

- **Requisito funcional 12: Eliminar y edita datos de proveedores**

Se agrupan estas dos funciones ya que tiene privilegio el usuario administrador de actualizar y editar los datos del proveedor según sea conveniente, ya sea por cambio de dirección, razón social, cambio de teléfono etc., así como la eliminación de un proveedor en caso de ya o trabajar con ellos o asignarlo como inactivo.

- **Requisito funcional 13: Registro de salidas**

El inventario cuenta con una lista de los productos registrados y disponibles en el almacén, de los cuales cuando el usuario necesite hacer una salida al momento que necesiten un utilizar un producto, quedara registrado en el módulo de “salidas” esta función está disponible para ambos tipos de cuenta de usuario. Para realizar esta función deberá llenar datos importantes como la cantidad a retirar y seleccionar su nombre en usuario que retira el producto.

3.8.1.1.Requisitos no funcionales

Se tomaron en cuenta los siguientes requisitos no funcionales para el desarrollo del sistema.

- **Requisito no funcional 1: Integridad de datos en el registro**

En el sistema se implementan medidas para salvaguardar la integridad de datos mediante el registro y modificación de datos. Al realizar una validación en los campos numéricos, especialmente aquellos asociados con cantidades, para asegurar que cumplan con la restricción de una longitud máxima de 2 caracteres, y no se ingresen negativos en el caso de las compras, se pueda generar la operación por si sola al dar el total de la compra, se consideran en todos los campos de registro no hayan campos vacíos para evitar errores posibles al introducción la información de esta manera cuenta con la coherencia y consistencia durante la fase de registro y modificación de datos.

- **Requisito no funcional 2: No repudio en el registro**

Con el propósito de conseguir el no repudio en los procesos de editar y eliminar el sistema procederá de manera automatizada a generar un registro. Dicho registro incorporara información ingresada de manera precisa de la modificación que se hizo, así como los datos específicos que han sido objeto de alteración. Es necesario que el usuario ingrese datos correctos en los campos asignados de ser así el sistema generara un mensaje en pantalla de

registro exitoso, este mensaje se mostrara como una advertencia “El registro ha sido exitoso”, si los datos no cumplen con las características esperadas el mensaje no será generado hasta que el usuario ingrese los datos correctamente.

- **Requisito no funcional 3: Adaptabilidad en dispositivos**

Con la finalidad de que el sistema coexista de la mejor manera dentro de los diferentes dispositivos capaces de mostrar una página web, se opta por la implementación de la configuración para la visualización del contenido efectiva en una variedad de dispositivos, incluyendo computadoras de escritorio, laptops, tabletas y teléfonos móviles, proporcionando una experiencia de usuario consistente.

3.9.Requisitos específicos

Esta sección presenta los requisitos detallados a un nivel de detalle a los diseñadores para concebir un sistema que cumpla con estos requisitos. Además, permite planificar y ejecutar pruebas que evidencien la satisfacción o no del sistema en relación con los requisitos establecidos.

3.9.1. Caso de uso general

El siguiente caso de uso general representa mediante su notación en UML el sistema completo incluyendo los actores y funciones, así como es que interactúan dentro del software.

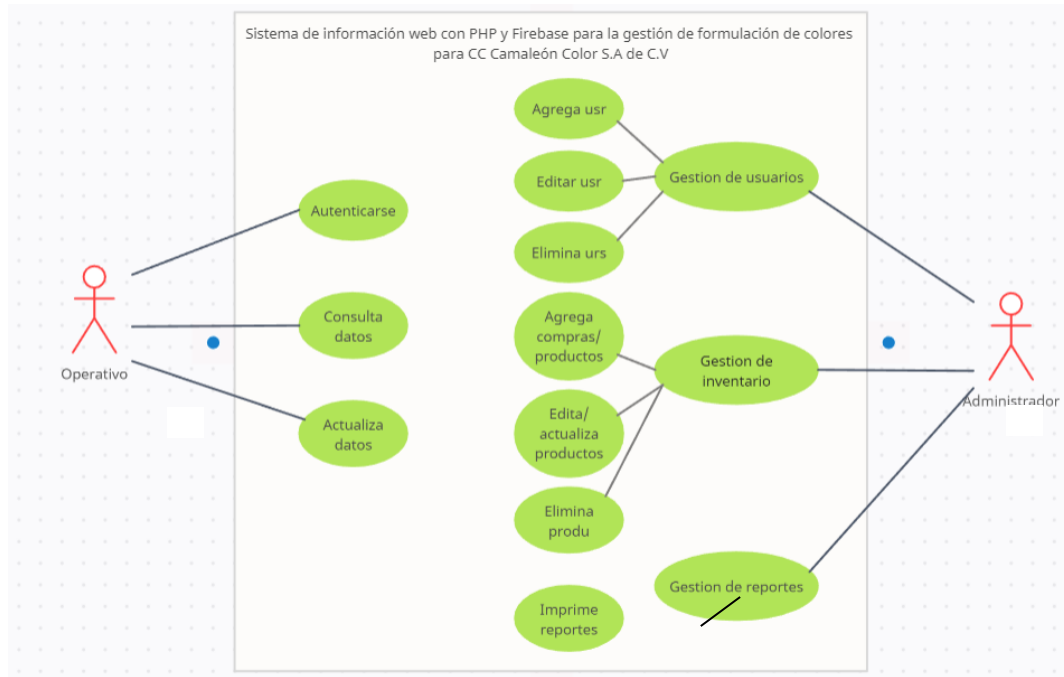


Figura. 4 Diagrama de caso de uso general del sistema. Creatly Elaboración propia

El diagrama que se observa en la figura.4 muestra las funciones generales del sistema web de inventario para un usuario administrador y un usuario operativo el cual tiene accesos limitados en el sistema.

- Requisito funcional 1: Autenticar cuenta de usuario administrador

El usuario deberá autenticar su cuenta utilizando los datos proporcionados durante el registro en la sección de inicio de sesión. Para acceder al portal web, deberá ingresar su correo electrónico y contraseña previamente creadas por el administrador.

- **Caso de uso específico de inicio de sesión (autenticar usuario)**

La figura 5. muestra el caso de uso específico de autenticarse:

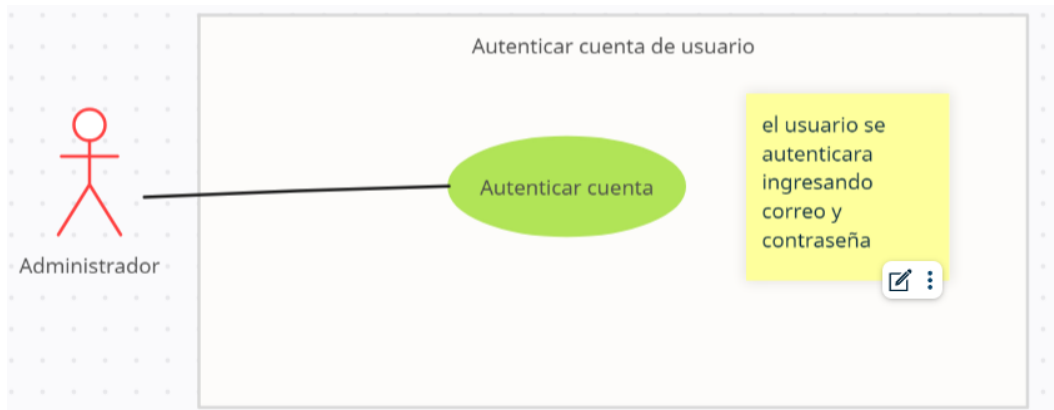


Figura. 5 Caso de uso específico autenticar usuario. Fuente. Elaboración propia.

- **Escenario de requisito funcional 1**

La tabla 4 muestra el escenario de registrar un nuevo producto la cual describe precondiciones, flujo normal y alterno, así como las post condiciones

Tabla 4 Escenario de requisito funcional 1

Número del requisito:	1
Nombre del requisito:	Autenticar cuenta de usuario.
Prioridad:	Alta
Descripción:	El cliente se autenticará por medio de su correo electrónico y contraseña dados en su registro.
Actores:	Administrador.
Precondiciones:	Es necesario que para realizar esta acción el usuario ya haya sido creado por el desarrollador previamente para que pueda autenticarse como administrador.
Flujo Normal:	1: El usuario deberá ingresar con el link del portal web. 2: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando con el formulario que se deberá llenar. 3: Se ingresa correo y contraseña previamente creados, hacer clic en entrar. 4: Al a hacer una autenticación exitosa el sistema arroja un mensaje con “loggeo exitoso” se da clic en aceptar mostrando así el menú principal.

Flujo Alternativo:	El usuario deberá ingresar y verificar que este correcto sus datos para que pueda autenticarse correctamente ya que si agrega un dato que no concuerde con lo que se ingresó en el registro le saldrá un mensaje de “Correo no registrado” o en su defecto si se ingresa una contraseña errónea mandara mensaje de “contraseña incorrecta”. De igual manera si no llena algún campo, le saldrá un mensaje, el sistema pedirá que ingrese correo o contraseña según sea el caso
Post condiciones:	Todos los datos ingresados en la parte del proceso de registro como el correo electrónico y contraseña son necesarios para poder autenticarse.

Pantallas de flujo normal Requisito funcional 1

En la primera pantalla que se muestra al cargar la página web en la figura 6 refleja la interfaz de inicio de sesión del portal web de inventario.

Figura. 6 Pantalla de inicio de sesión. Fuente. Elaboración propia.

Al ingresar los datos el usuario se autentica ya sea el caso de personal operativo o administrador.

Ingresa correo en el primero formulario y contraseña como se muestra en la figura. 7



¡Bienvenido de nuevo!
Soy camaleón y mi nombre es color.

Administrador

CORREO:

CONTRASEÑA:

Entrar

¿Eres Personal?

Inicia sesión para entrar en la página

Entra aquí

Figura. 7 Pantalla de ingreso de datos. Fuente. Elaboración propia

Mensaje de logeo exitoso en laFigura.8

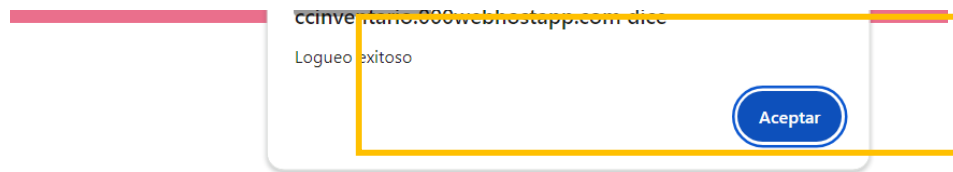


Figura. 8 Ingreso al sistema. Fuente. Elaboración propia.

Al dar clic en aceptar muestra la pantalla de menú principal como se muestra en la Figura 9.



Figura. 9 Menú principal del sistema de inventario. Fuente. Elaboración propia.

- **Requisito funcional 2: Registrarse como nuevo usuario**

Se requiere que el usuario administrador cree una cuenta solo el administrador tendrá privilegios de crear o eliminar una nueva cuenta, podrá crear cuentas de administrador o usuario común, para el registro de un nuevo usuario deberán ser llenados todos los campos que se requieran por el sistema, nombre completo, correo, contraseña, de lo contrario no procederá un registro exitoso.

- **Caso de uso específico de registro de usuario**

La figura 10 muestra el caso de uso específico de registrar un nuevo usuario.



Figura. 10 Caso de uso específico de registro de nuevo usuario. Fuente. Creately, elaboración propia.

- **Escenario de requisito funcional 2**

La tabla 5 muestra el escenario de registro de nuevo usuario, la cual describe precondiciones, flujo normal y alterno, así como las post condiciones

Tabla 5. Escenario de requisito funcional 2

Número del requisito:	2
Nombre del requisito:	Registro de nuevo usuario
Prioridad:	Alta
Descripción:	El usuario se autenticará por medio de su correo electrónico y contraseña dados en su registro.
Actores:	Administrador.

Precondiciones:	Es necesario que para realizar esta acción el usuario ya este registrado previamente por el desarrollador para dar inicio a nuevos usuarios
Flujo Normal:	1: El usuario deberá ingresar con el link del portal web. 2: En la primera interfaz que podrá ver es la pantalla de inicio de sesión mostrando con el formulario que se deberá llenar. 3: Se ingresa correo y contraseña previamente creados, hacer clic en entrar. 4: Al a hacer una autenticación exitosa el sistema mostrará un mensaje con “loggeo exitoso” se da clic en aceptar mostrando así el menú principal. 5: dará clic del lado izquierdo y desplegará la opción de usuarios. 6: desplegará una lista en la cual dará clic en el tipo de usuario que desee agregar ya se otro con privilegios de administrador o de usuario operativo. 7: El administrador iniciará registrar el usuario, con los campos Nombres, Apellidos, correo electrónico y contraseña para poder registrar un usuario con éxito
Flujo Alterno:	El usuario no debe saltarse ningún campo, de ser así saldrá una advertencia mencionando el campo que dejo vacío de esta manera el sistema no avanzará si no se ingresa el dato faltante o que se ingresen datos que no coincidan con el formato específico.
Post condiciones:	Tener una cuenta como administrador para poder agregar otros usuarios

Pantallas de flujo normal requisito funcional 2



Figura. 11 Ingreso al menú principal. Fuente. Elaboración propia.

Al ver en pantalla el siguiente menú posicionarse en la opción de USUARIOS como se ve en la Figura 11.

Al dar clic en el módulo de usuarios se desplegará una lista con las opciones de usuarios a elegir para crear un nuevo usuario como lo ve en la figura. 12 se muestran dos opciones después de dar clic en usuarios.



Figura. 12 Lista desplegable de usuarios. Fuente. Elaboración propia.

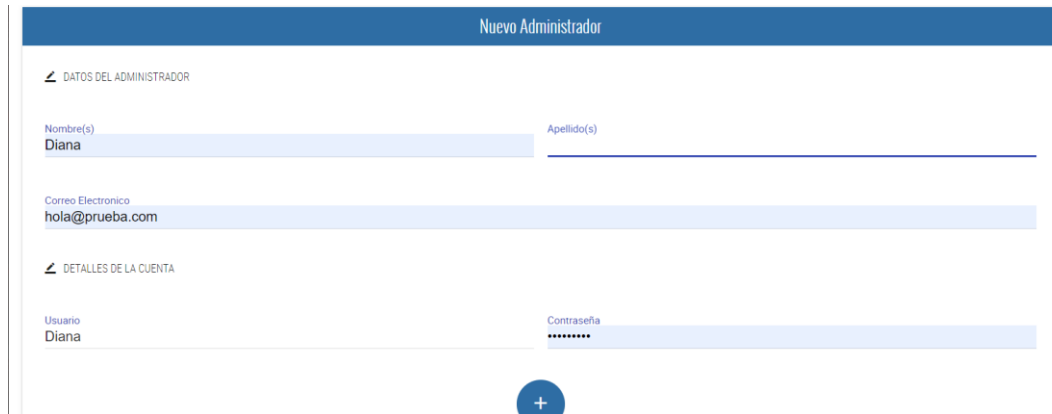
El usuario dará clic en el usuario que desee crear ya sea de usuario calificado que es lo mismo que usuario operativo o de administrador.

Una vez completada esta acción lo dirige a la siguiente pantalla mostrada en la figura. 13.

The screenshot shows the 'Nuevo Administrador' form. The sidebar menu is identical to the previous figure, but the 'ADMINISTRADORES' option is highlighted. The main area has a title bar 'Nuevo Administrador' with tabs for 'NUEVO' and 'LISTA'. Below the title bar, there are two sections: 'DATOS DEL ADMINISTRADOR' with fields for 'Nombre(s)', 'Apellido(s)', and 'Correo Electronico'; and 'DETALLES DE LA CUENTA' with fields for 'Usuario' and 'Contraseña'. A blue circular button with a white plus sign is at the bottom right.

Figura. 13 Formulario de registro de nuevo usuario. Fuente. Elaboración propia.

Cabe mencionar que se necesita llenar todos los campos que se muestran en el formulario en caso de lo contrario como se muestra en la figura. 14 aparecerá una alerta que pide ingrese el dato que haya faltado como se refleja en la figura 15.



The screenshot shows a web form titled "Nuevo Administrador". It is divided into two sections: "DATOS DEL ADMINISTRADOR" and "DETALLES DE LA CUENTA". In the first section, the "Nombre(s)" field contains "Diana" and the "Apellido(s)" field is empty. In the second section, the "Correo Electronico" field contains "hola@prueba.com", the "Usuario" field contains "Diana", and the "Contraseña" field is filled with asterisks. A blue circular button with a white plus sign is located at the bottom center of the form.

Figura. 14 Ejemplo de llenado incorrecto. Fuente. Elaboración propia.

Alerta del sistema de datos faltantes

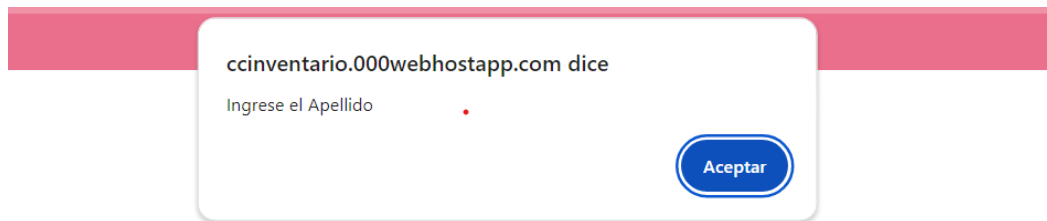


Figura. 15 Mensaje de alerta de datos faltantes. Fuente. Elaboración propia.

En la figura 16 se muestra un ejemplo del mensaje que aparece en interfaz cuando el usuario ingresa un usuario con ID, nombre o correo ya existente.

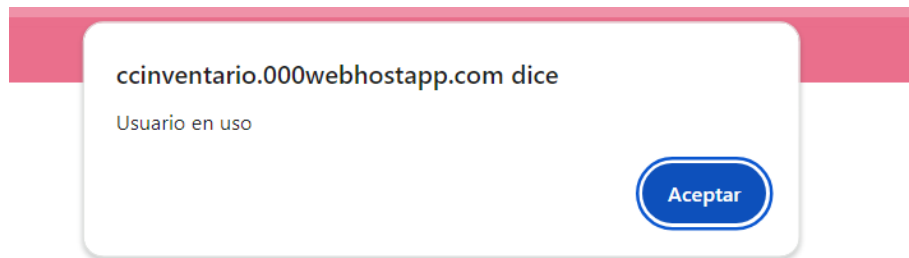


Figura. 16 Mensaje de agregar usuario exitoso. Fuente. Elaboración propia.

- **Requisito funcional 3: Editar datos de usuario**

El usuario administrador tiene privilegios de editar datos de un usuario, podrá editar el nombre en caso de haber faltado datos, o si correo en caso de cambio de correo, así como la contraseña proporcionada.

- **Caso de uso específico de requisito funcional 3**

La figura 17 muestra el caso de uso muestra la funcionalidad para editar información de un usuario existente.

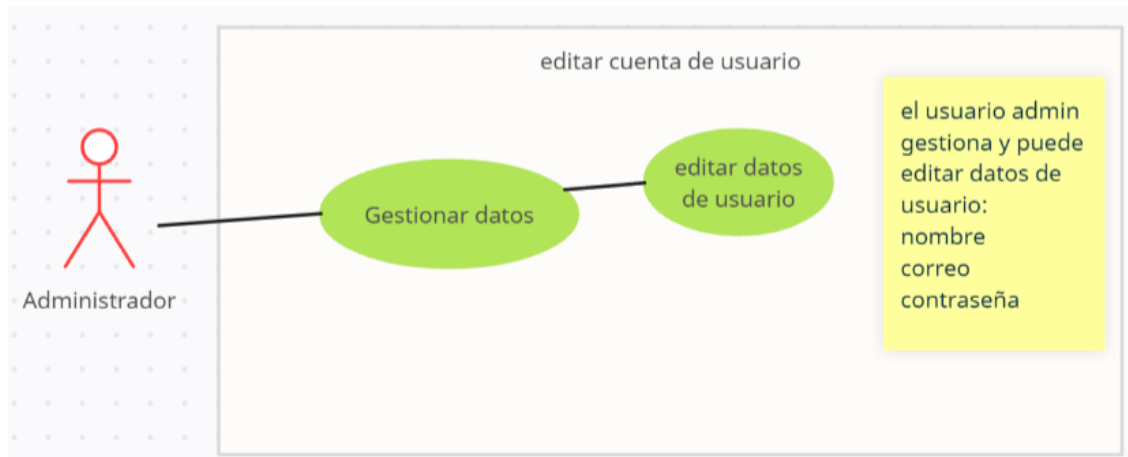


Figura. 17 Caso de uso editar datos. Fuente. Creately. Elaboración propia.

- **Escenario de requisito funcional 3**

La tabla 6 muestra el escenario de editar usuario, la cual describe precondiciones, flujo normal y alterno, así como las post condiciones.

Tabla 6 Escenario de requisito funcional 3. Elaboración propia

Número del requisito:	3
Nombre del requisito:	Editar usuario
Prioridad:	Media
Descripción:	El usuario administrador tendrá la posibilidad de editar datos de otros usuarios en caso de que alguno cambie, ya sea el caso de su correo electrónico o completar su nombre
Actores:	Administrador
Precondiciones:	Es necesario que para realizar esta acción el usuario inicie sesión como administrador.

Flujo Normal:	1: En la primera interfaz que podrá ver es la pantalla de inicio de sesión mostrando el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión. 3: Al ingresar al sistema navegar al módulo de usuarios. 4: Dar clic en la lista desplegable y dar clic en el tipo de usuario que desee editar 5: Seleccionar el usuario que desee editar 6: Modificar datos séase en los campos nombres, apellidos, correo electrónico o contraseña.
Flujo Alterno:	1: El usuario debe ingresar datos de acuerdo con el formato de lo contrario el sistema, mostrara error 2: Deberá corregir datos para intentar guardar cambios nuevamente.
Post condiciones:	Los datos deberán actualizarse correctamente en la base de datos, así como actualizar las listas de la plataforma

Pantallas de flujo normal Requisito funcional 3

Para comenzar el proceso de editar dará clic en la selección de tipo de usuario, el usuario dará clic en Lista como se señala en la figura 18.

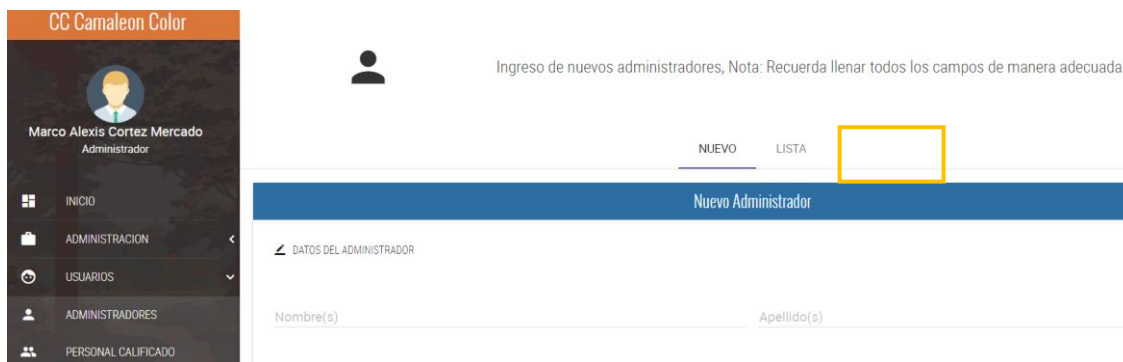


Figura. 18 Entrar a la sección de editar en lista. Fuente. Elaboración propia.

Al dar clic en “Lista” se muestran los usuarios como se muestra en la figura 19

Lista de Administradores						
10	entradas por página		Buscar...			
Id	Nombre(s)	Apellido(s)	Usuario	Correo	Contraseña	Acción
1	Marco Alexis	Cortez Mercado	Administrador	prueba@gmail.com	Bolillo34	 
2	Diana Ivonne	Hernández de la Cruz	DianaAdmin	Diana@gmail.com	123456789	 

Figura. 19 Lista de usuarios. Fuente. Elaboración propia.

En la figura 20 se muestra el icono en forma de lápiz al que deberá dar clic para editar.

Correo	Contraseña	Acción
prueba@gmail.com	Bolillo34	 
Diana@gmail.com	123456789	 

Figura. 20 Icono de editar usuario. Fuente. Elaboración propia.

Lo siguiente con lo que el usuario tendrá que interactuar es el cambio de datos que convenga ejemplo en figura 21.

Correo Electronico
diana@gmail.com

DETALLES DE LA CUENTA

Usuario
DianaAdmin

Contraseña

Figura. 21 Edición de administrador. Fuente. Elaboración propia.

Caso de editar datos de usuario en la sección de correo electrónico se muestra en la figura 22

Edición de Administrador

DETALLES DEL ADMINISTRADOR

Nombre(s)
Diana Ivonne

Apellido(s)
Hernández de la Cruz

Correo Electronico
ivonne.diana@gmail.com

DETALLES DE LA CUENTA

Figura. 22 Cambio de correo. Fuente. Elaboración propia.

Dar clic en el icono de  y así los datos serán actualizados en la lista y base de datos

- **Requisito funcional 4: Eliminar usuario**

El usuario administrador podrá eliminar a los usuarios, esto para mejor seguridad en caso de alguna baja de personal, no tenga acceso al sistema.

- **Caso de uso específico de requisito funcional 4**

La figura 23 muestra el caso de uso muestra la funcionalidad para eliminar un usuario existente.

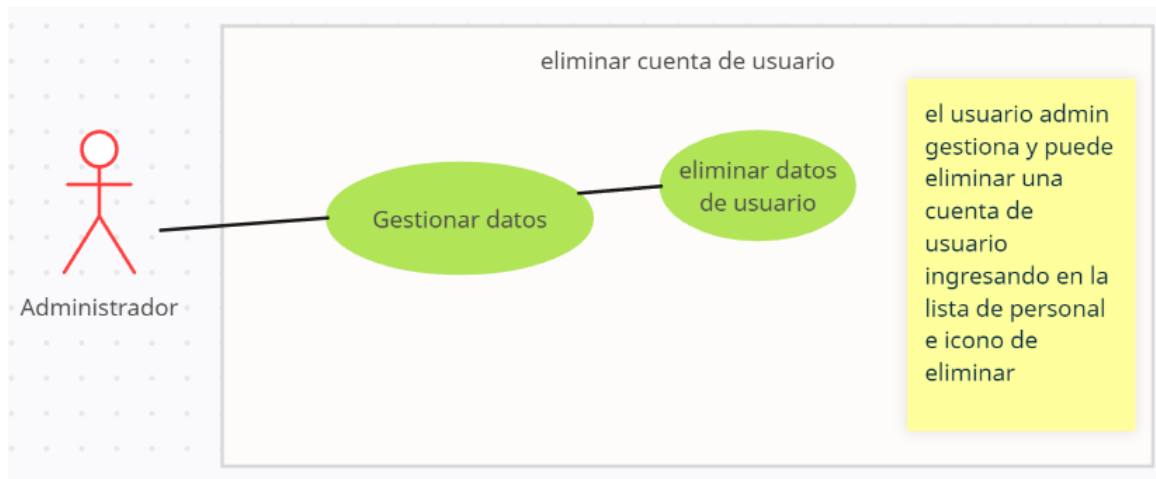


Figura. 23 Caso de uso específico eliminar cuenta de usuario. Fuente. Creately, elaboración propia.

- **Escenario de requisito funcional 4**

La tabla 7 muestra el escenario de la función eliminar usuario, la cual describe precondiciones, flujo normal y alterno, así como las post condiciones

Tabla 7 Escenario de requisito funcional 4. Elaboración propia.

Número del requisito:	4
Nombre del requisito:	Eliminar usuario
Prioridad:	Media
Descripción:	El usuario administrador.
Actores:	Administrador

Precondiciones:	Ingresar como administrador, y deberá existir al menos una cuenta de usuario en el sistema para poder ser eliminado.
Flujo Normal:	1: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando el login con el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión. 4: Al ingresar al sistema navegar al módulo de usuarios. 5: Dar clic en la lista desplegable y dar clic en el tipo de usuario que desee editar 6: El sistema mostrara una lista con todos los datos de los usuarios previamente creados. 7: El administrador busca el usuario que desee eliminar 8: Una vez localizado el usuario deberá seleccionar el botón con el icono de basura para eliminar.
Flujo Alterno:	1: El usuario no tiene privilegios de administrador. 2: Se deberá proporcionar confirmación adicional del director para poder eliminar un usuario 3: Si el usuario cancela el proceso de eliminación, el sistema regresara a la lista de usuarios sin haber realizado ningún cambio. 4: En caso de notar algún problema
Post condiciones:	1: El usuario eliminado se borra también de la base de datos. 2: La base de datos y la lista se deberá actualizar

Pantallas de flujo normal Requisito funcional 4

En la pantalla de usuarios dar clic en las listas al igual como si fuese a editar un usuario.

Figura 24.



Figura. 24 Ingreso a lista de usuarios. Fuente. Elaboración propia.

Al ingresar la lista de usuarios se muestra la siguiente interfaz con la lista de usuarios como se ve en la figura 25.

Lista de Personal						
ID	Nombre(s)	Apellido(s)	Usuario	Correo	Contraseña	Acción
1	Diana	Hernández de la Cruz	pintor1	personal@gmail.com	12345678	 

Mostrando 1 de 1 de 1 entradas

Figura. 25 Lista de usuarios. Fuente. Elaboración propia.

Al identificar el usuario a eliminar se dará clic al siguiente icono mostrado en figura 26:



Figura. 26 Icono de eliminar. Fuente. Elaboración propia.

- Requisito funcional 5: Registrar nuevo producto

El usuario con acceso al sistema como administrador podrá agregar productos, deberá llenar un formulario proporcionando la información requerida por el sistema, sin dejar ningún campo vacío, llevará campos como el nombre de producto, categoría, proyecto para el cual se utiliza el producto fecha de entrada de registro proveedor y la ubicación en donde se encontrará asignado el producto.

- **Caso de uso específico de requisito funcional 5**

La figura 27 muestra el caso de uso de la funcionalidad de agregar nuevo producto.

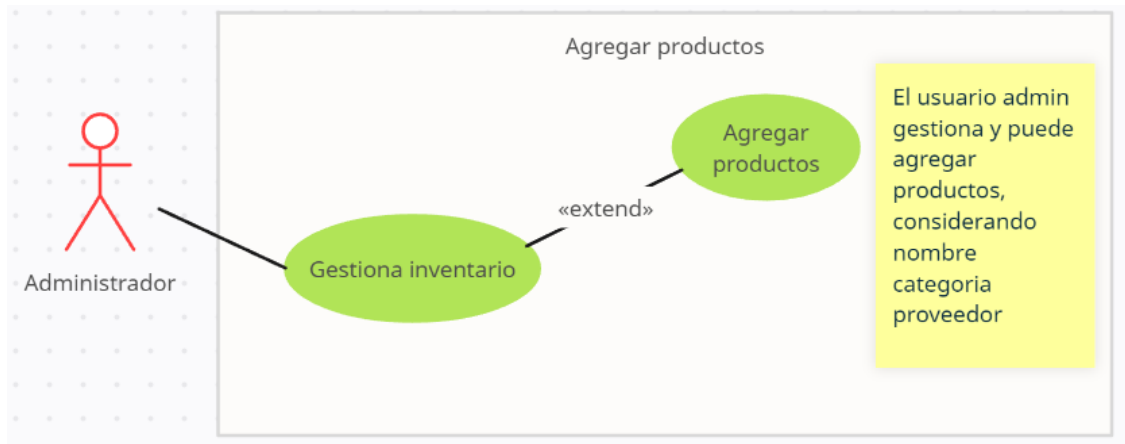


Figura. 27 Caso de uso Agregar producto. Fuente. Elaboración propia.

- **Escenario de requisito funcional 5**

La tabla 8 muestra el escenario de registrar un nuevo producto, la cual describe precondiciones, flujo normal y alterno, así como las post condiciones

Tabla 8 Escenario de requisito funcional 5. Elaboración propia.

Número del requisito:	5
Nombre del requisito:	Registrar nuevo producto
Prioridad:	Alta
Descripción:	El usuario administrador o usuario operativo registra nuevo producto para uso del inventario de la empresa.
Actores:	Administrador
Precondiciones:	El usuario que realizará esta función deberá iniciar sesión, como administrador o usuario operativo.
Flujo Normal:	1: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión. 4: Al ingresar al sistema navegar al módulo con el nombre de entradas. 5: Dar clic nuevo. 6: El sistema mostrará un formulario en donde deberá llenar los siguientes datos: Nombre del producto, día de ingreso, categoría, proyecto, proveedor, y ubicación.

	7: dar clic en el icono del + para agregar.
Flujo Alternativo:	1: Si el usuario excede de tiempo se cerrará sesión u sus cambios o serán guardados. 2: Si algún campo del formulario no se completa correctamente el sistema no dejara hacer ningún cambio, regresando a la pantalla de registro nuevamente hasta que se llene correctamente el formulario.
Post condiciones:	1: Las bases de datos del inventario se deberán actualizar con la información del nuevo producto

- **Requisito funcional 6: Editar producto**

El usuario administrador tiene los privilegios de editar algún producto que hayan metido en entradas ya existentes se podrá modificar en sus campos existentes excepto en el campo de ID.

- **Caso de uso específico de requisito funcional 6**

En la figura 28 se muestra el caso de uso específico para la función editar productos del inventario.

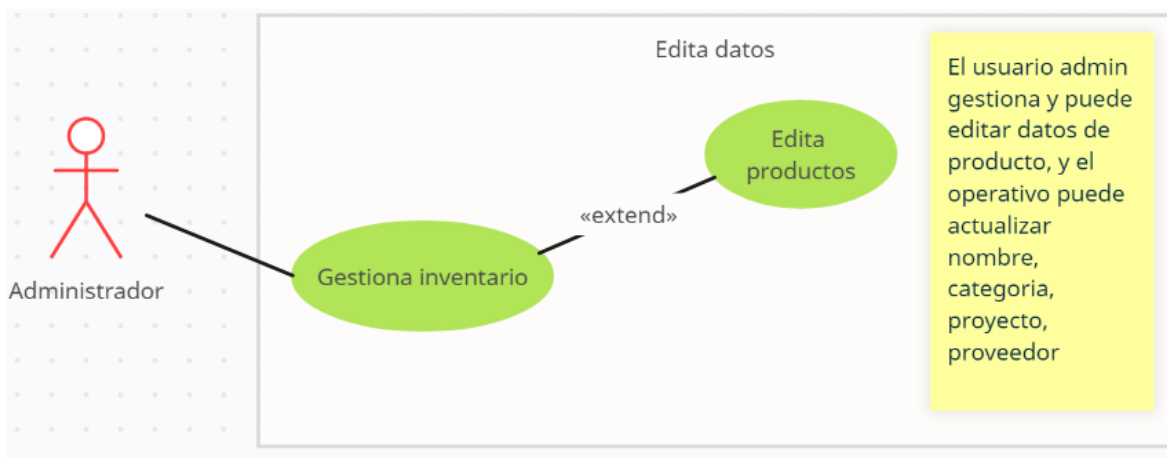


Figura. 28 Caso de uso editar usuario. Fuente. Creately, elaboración propia.

- **Escenario de requisito funcional 6**

La tabla 9 muestra el escenario de editar un producto, la cual describe precondiciones, flujo normal y alterno, así como las post condiciones.

Tabla 9 Escenario de requisito funcional 6 Elaboración propia.

Número del requisito:	6
Nombre del requisito:	Editar productos
Prioridad:	Media
Descripción:	El usuario administrador encargado del inventario podrá editar datos de los productos registrados en caso de actualización.
Actores:	Administrador
Precondiciones:	El usuario deberá iniciar sesión como administrador Deberá existir un producto previamente registrado para poder ser editado.
Flujo Normal:	1: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión. 4: Al ingresar al sistema navegar al módulo de entradas. 5: mostrara la pantalla de agregar nuevo producto. 6: En lugar de agregar un nuevo producto el usuario deberá dar clic en “Lista” 7: Se mostrará la lista de los productos registrados previamente. 8: El usuario selecciona el producto que desee editar, y deberá dar clic en el icono con símbolo de lápiz 9: Edita según le sea conveniente, se habilitará para edición el nombre de producto, categoría proyecto, proveedor y ubicación.
Flujo Alterno:	1: Deberá existir el producto que desee editar 2: Si se excede el tiempo de inicio de sesión aparecerá el mensaje de tiempo excedido y sus cambios no se guardarán. 3: Si no se llana algún cambio, no se guardaran los cambios y no permitirá continuar el proceso de edición de datos de producto.
Post condiciones:	1: El producto editado se refleja automáticamente en la también de la base de datos como en las listas. 2: La base de datos y la lista se deberá actualizar

Pantallas de flujo normal Requisito funcional 4

Seleccionar el módulo de entradas en el menú principal como se ve señalado en la figura 29.

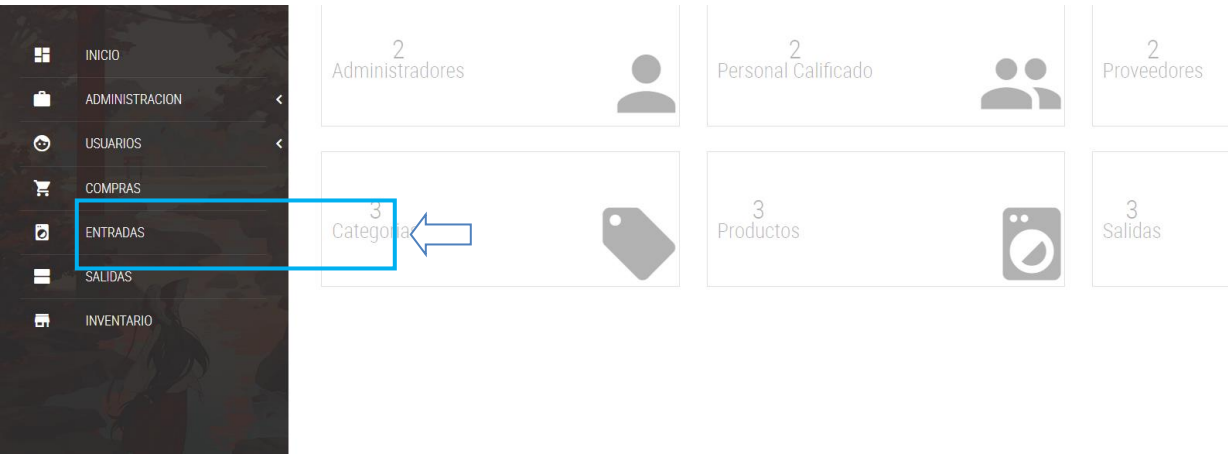


Figura. 29 Menú principal. Fuente. Elaboración propia.

Al ingresar al módulo de entradas encontrara la lista de los productos ya registrados, como se ve en la figura 30, por lo que deberá buscar para identificar el producto que necesite editar.



Figura. 30 Lista para editar. Fuente. Elaboración propia.

Seleccionar el producto y dar clic en el icono de lápiz



El cual permite llevara la siguiente pantalla, como se muestra en figura 31.

Figura. 31 formulario para editar producto. Fuente. Elaboración propia.

En la figura 32 se muestra el ejemplo del proceso de cambio de nombre de producto.

Figura. 32 Cambio de nombre de producto. Elaboración propia.

Se editan los datos y para guarda los cambios seleccionar el símbolo de  y volver a la lista visualizando el nombre del producto actualizado. Figura 33 muestra la lista actual.

ID	Nombre_del_producto	Fecha	Categoría	Proyecto	Prc
1	Pintura roja	2024-01-21	Puntura	No Especificado	Grup

Figura. 33 Lista actualizada de productos. Fuente. Elaboración propia.

- Requisito funcional 7: Eliminar producto

Si el usuario desea eliminar un producto en caso de ya no manejar ese producto solo el administrador tendrá los permisos de hacerlo mismo que se verá actualizado en el módulo de inventario.

- Caso de uso específico de requisito funcional 7

En la figura 34 se muestra el caso de uso específico de la funcionalidad de eliminar producto.

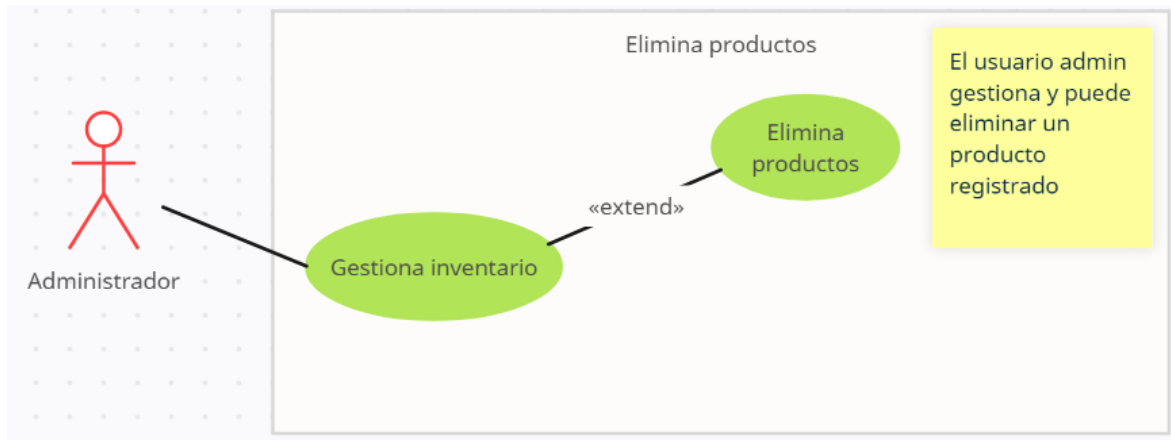


Figura. 34 Caso de uso específico de eliminar producto. Fuente. Creately, elaboración propia.

- Escenario de requisito funcional 7

La tabla 10 muestra el escenario de la función eliminar productos la cual describe precondiciones, flujo normal y alterno, así como las post condiciones.

Tabla 10 Escenario de requisito funcional 7. Elaboración propia.

Número del requisito:	7
Nombre del requisito:	Eliminar productos
Prioridad:	Media
Descripción:	El usuario administrador encargado del inventario podrá eliminar productos de su lista registrada en caso de que sea obsoleto en la empresa.
Actores:	Administrador
Precondiciones:	El usuario deberá iniciar sesión como administrador Deberá existir un producto previamente registrado para poder ser eliminado.
Flujo Normal:	1: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión. 4: Al ingresar al sistema navegar al módulo de entradas.

	5: Dar clic en “Lista”. 6: Se mostrará la lista de los productos registrados previamente. 8: El usuario selecciona el producto que desee eliminar, y deberá dar clic en el icono con símbolo de bote de basura.
Flujo Alterno:	1: Deberá existir el producto que desee eliminar 2: Si se excede el tiempo de inicio de sesión aparecerá el mensaje de tiempo excedido y sus cambios no se guardarán.
Post condiciones:	1: El producto eliminado se elimina igualmente en la base de datos, así como en las listas.

Pantallas de flujo normal Requisito funcional 7

Ir al módulo de lista de productos como se muestra en la figura 35.

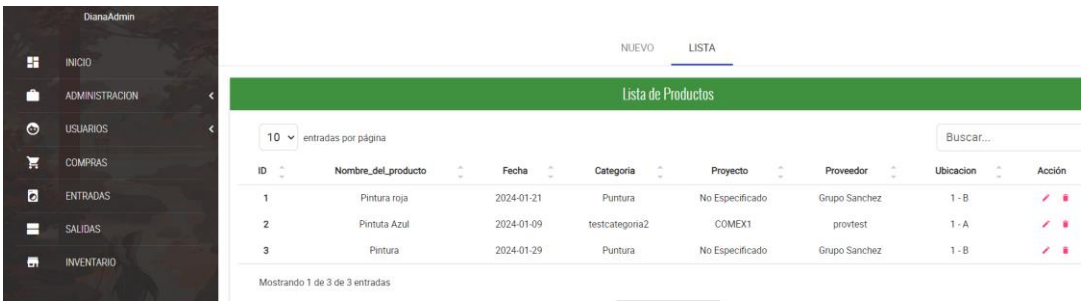


Figura. 35 Módulo de lista de productos. Fuente. Elaboración propia.

El usuario podrá elegir el producto que desee eliminar y seleccionar el icono siguiente:  Con el siguiente mensaje de confirmación como se muestra en figura 36 a la eliminación del producto seleccionado.

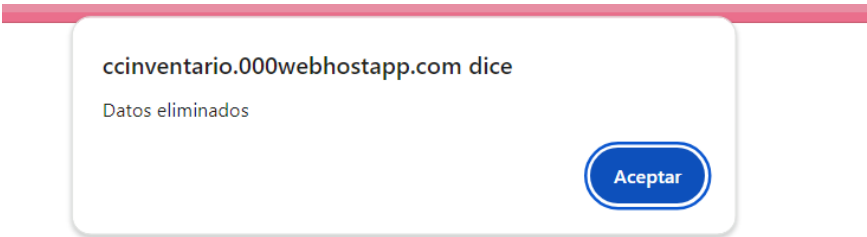


Figura. 36 Confirmación de eliminación de producto. Fuente. Elaboración propia.

10	entradas por página	Buscar...					
ID	Nombre_del_producto	Fecha	Categoría	Proyecto	Proveedor	Ubicacion	Acción
1	Pintura roja	2024-01-21	Puntura	No Especificado	Grupo Sanchez	1 - B	 
2	Pintura	2024-01-29	Puntura	No Especificado	Grupo Sanchez	1 - B	 

Mostrando 1 de 2 entradas

Figura. 37 Lista de productos actualizada. Fuente. Elaboración propia.

- Requisito funcional 8: Agregar nueva compra

En sistema tiene el módulo de agregar nueva compra en el cual usuario operativo y usuario administrador podrá registrar las compras que se hacen con numero de factura, fecha de compra, descripción.

- Caso de uso específico de requisito funcional 7

En la figura 38 se muestra el caso de uso específico de la función de agregar nueva compra.

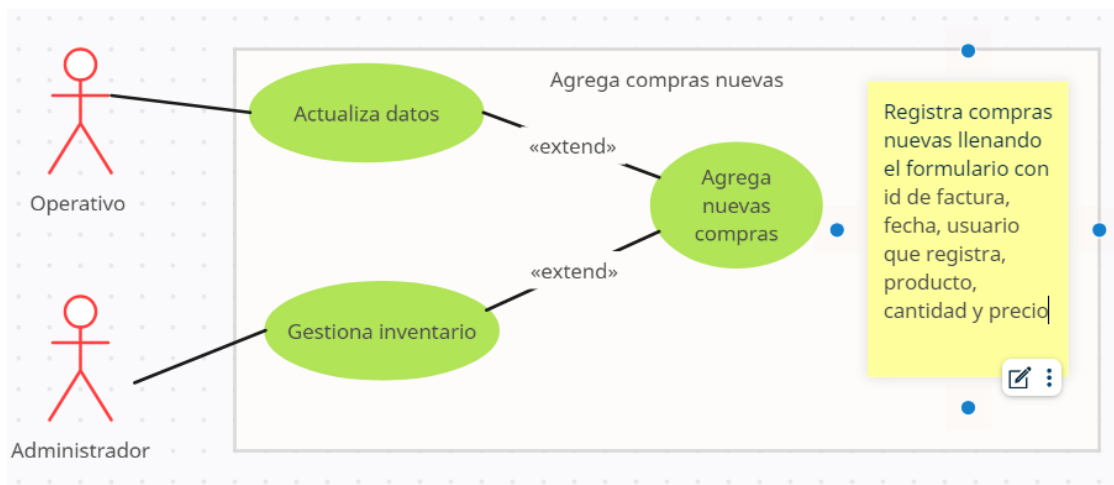


Figura. 38 Caso específico de agregar nuevas compras. Fuente Creately Elaboración propia

- Escenario de requisito funcional 8

La tabla 11 muestra el escenario de agregar nueva compra la cual describe precondiciones, flujo normal y alterno, así como las post condiciones.

Tabla 11 Escenario de requisito funcional 8. Elaboración propia.

Número del requisito:	8
Nombre del requisito:	Agregar nueva compra.
Prioridad:	Media
Descripción:	El usuario administrador encargado del inventario o usuario operativo podrá ingresar el registro de compras nuevas.
Actores:	Administrador / Operativo
Precondiciones:	El usuario deberá iniciar sesión como administrador o usuario operativo. Deberá existir un producto y proveedor previamente registrado para registrar la compra.
Flujo Normal:	1: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión en el caso de administrador, y en caso de usuario operativo con el nombre de usuario proporcionado por el administrador y contraseña igualmente proporcionado por el supervisor administrador. 3: Al ingresar al sistema navegar al módulo de compras. 4: registrar la compra llenando todos los campos correspondientes como lo son el ID de compra de la factura, fecha de la compra, nombre del producto usuario que ingresa la compra, cantidad y precio total de la compra 5: El usuario el icono con el símbolo de + para agregar esa cuenta. 6: verifica en la lista de compras si se guardó correctamente.
Flujo Alterno:	1: Deberá existir el producto y proveedor que para que pueda agregar esa compra, ya debe existir el producto en inventario, de ser nuevo se registra primero el producto y proveedor. 2: Si se excede el tiempo de inicio de sesión aparecerá el mensaje de tiempo excedido y sus cambios no se guardarán. 3: Si el usuario no llena todos los campos no se podrán guardar los cambios y regresara nuevamente al formulario para que complete los datos faltantes
Post condiciones:	1: La compra registrada se refleja en las listas y se guarda en la base de datos.

Pantallas de flujo normal Requisito funcional 8

Pantalla principal, ingresar al módulo de compras en Figura 39.

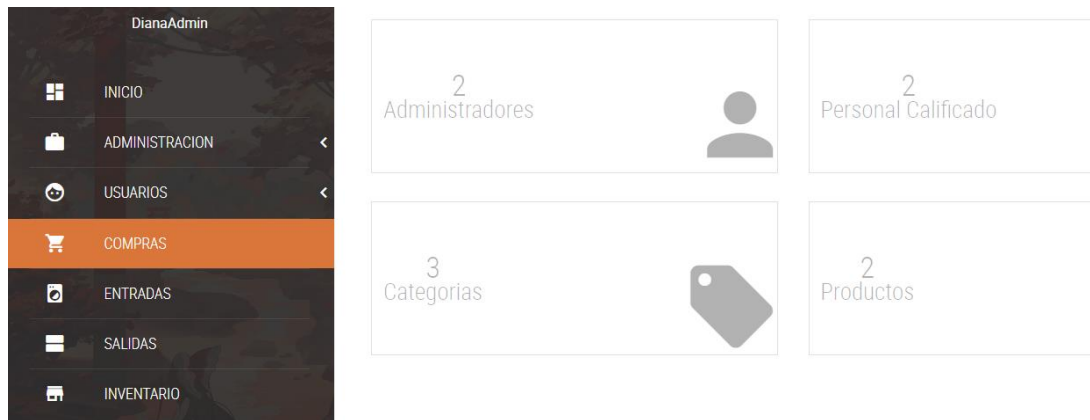


Figura. 39 Página principal y módulo de compras. Fuente. Elaboración propia.

Al dar clic en compras se muestra la siguiente pantalla reflejada en la figura 40.

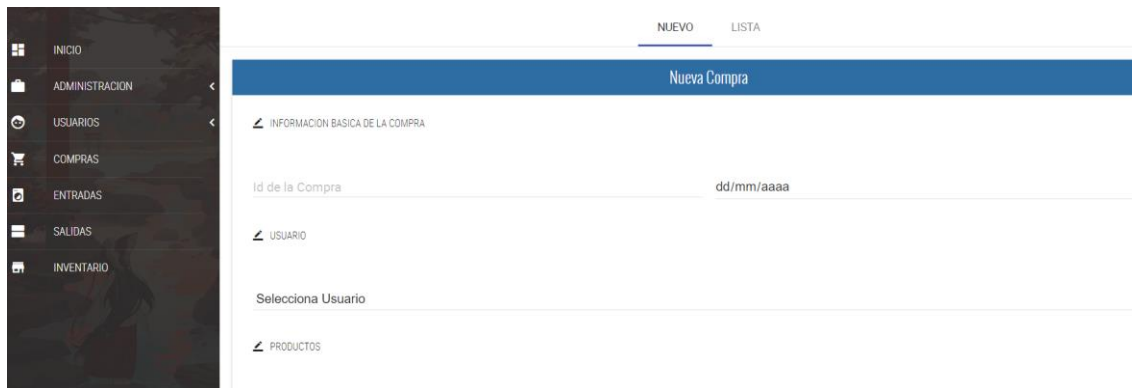


Figura. 40 Módulo de nueva compra. Fuente. Elaboración propia.

En esta sección se podrá agregar un nuevo producto llenando todos los campos. Como se muestra a continuación en las siguientes pantallas como ejemplo en la figura 41.

INGRESE CANTIDAD EN: MINILITRO,GRAMOS O PIEZAS

Ingrese la Cantidad
1000

FINALIZACION DE COMPRA

Ingrese la cantidad 1	Precio Unitario 400	464.00
--------------------------	------------------------	--------

+

Figura. 41 Agregar nueva compra. Fuente. Elaboración propia.

Al quedar registrada la compra se muestra en el historial de compras (Figura 42).

ID	Nombre del producto	Categoría	Unidad de Medida	Stock	Precio Unitario	Fecha de Compra	Proveedor	Precio Total
3	Diana Ivonne	Pintura roja	MiniLitros	1000	400	2024-03-01	Grupo Sanchez	464.00

Figura. 42 Compra agregada a la lista de compras. Fuente. Elaboración propia,

Al seguir este proceso se verá reflejado en el módulo de inventario actualizado en figura 43.

ID	Nombre del producto	Stock	Unidad de Medida	Proyecto	Ubicacion	Categoria	Proveedor
1	Pintura roja	10843	MiniLitros	No Especificado	1 - B	Puntura	Grupo Sanchez

Mostrando 1 de 1 de 1 entradas

Figura. 43 Actualización en inventario de la nueva compra. Fuente. Elaboración propia.

En caso de que falte un campo por llenar se mostrara un mensaje de advertencia mencionando que dato falto registrar para que el registro de la compra se puede efectuar correctamente en figura 44 se muestra que hay datos faltantes.

Grupo Sanchez

⌵ MEDIDA

Mini Litros

⌵ INGRESE CANTIDAD EN: MINILITRO,GRAMOS O PIEZAS

Ingrese la Cantidad

⌵ FINALIZACION DE COMPRA

Ingrese la cantidad

Precio Unitario

Figura. 44 Ejemplo de datos faltantes. Fuentes. Elaboración propia.

Si el usuario no registra un dato saldrá el siguiente mensaje en figura 45, en este ejemplo falto agregar la cantidad a registrar

ccinventario.000webhostapp.com dice

Cantidad erronea.

Aceptar

Figura. 45 Ejemplo de advertencia de campo faltante. Fuente. Elaboración propia.

- **Requisito funcional 9: Consulta de productos en el módulo de inventario**

El sistema cuenta con una barra de búsqueda en la sección de inventario en la cual se podrá buscar por proyecto, por nombre o por ID según prefiera el usuario administrador o usuario operativo.

- **Caso de uso específico de requisito funcional 9**

En la figura 46 se muestra el caso de uso específico de la funcionalidad de consultar productos en el módulo de inventario.

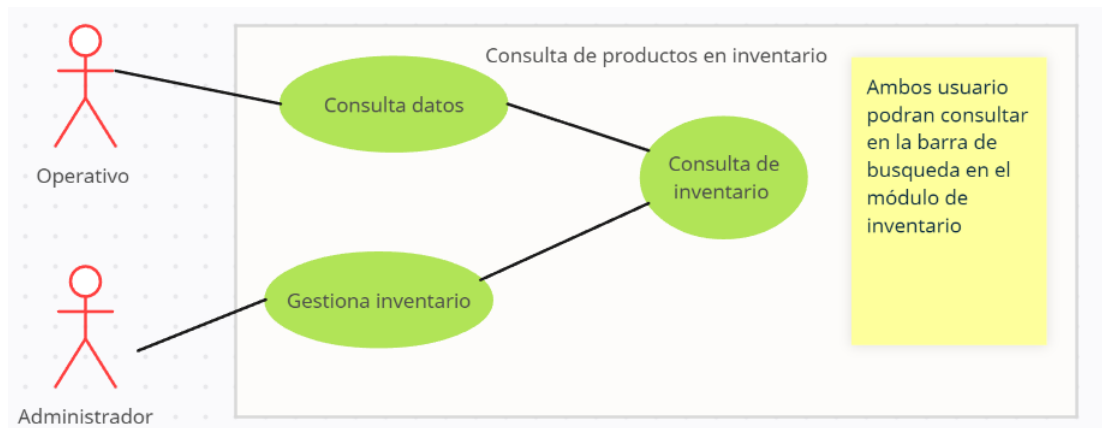


Figura. 46 Caso de uso específico de consultar productos. Fuente. Elaboración propia.

- **Escenario de requisito funcional 9**

La tabla 12 muestra el escenario de consulta de productos en inventario la cual describe precondiciones, flujo normal y alterno, así como las post condiciones.

Tabla 12 Escenario de requisito funcional 9. Elaboración propia.

Número del requisito:	9
Nombre del requisito:	Consulta de productos en inventario.
Prioridad:	Media
Descripción:	El usuario administrador encargado del inventario o usuario operativo podrá ingresar el registro de compras nuevas.
Actores:	Administrador / Operativo
Precondiciones:	El usuario deberá iniciar sesión como administrador o usuario operativo. Deberá existir un producto y proveedor previamente registrado para su búsqueda.
Flujo Normal:	1: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión en el caso de administrador, y en caso de usuario operativo con el nombre de usuario proporcionado por el administrador y contraseña igualmente proporcionado por el supervisor administrador. 4: Al ingresar al sistema navegar al módulo de inventario. 5: ir a la barra de búsqueda en la interfaz de inventario y buscar con ID, nombre de producto o proyecto.
Flujo Alterno:	1: No existe el producto que quisiera consultar 2: Si se excede el tiempo de inicio de sesión aparecerá el mensaje de tiempo excedido y deberá ingresar al sistema nuevamente para consultar el producto. 3: Si el usuario consulta con datos inexistentes el sistema no arrojará ningún resultado.
Post condiciones:	1: El sistema no realiza cambios en el inventario como resultado de la consulta.

Pantallas de flujo normal Requisito funcional 9

Ingresa a la pantalla principal posicionando el cursor en INVENTARIO, como se muestra en la figura 47.



Figura. 47 Pantalla principal para ingreso a inventario. Fuente. Elaboración Propia.

La siguiente pantalla es causa de haber ingresado al módulo de inventario para consulta ejemplo en figura 48.

ID	Nombre del producto	Stock	Unidad de Medida	Proyecto	Ubicacion	Categoria	Proveedor	Acción
1	Pintura roja	10843	MiniLitros	No Especificado	1 - B	Pintura	Grupo Sanchez	
2	Pintura Azul	5000	MiniLitros	COMEX1	1 - B	Pintura	Grupo Sanchez	

Mostrando 1 de 2 de 2 entradas

Exportar datos a Excel

Figura. 48 Lista de inventario. Fuente. Elaboración propia.

Ejemplo de búsqueda en figura 49.

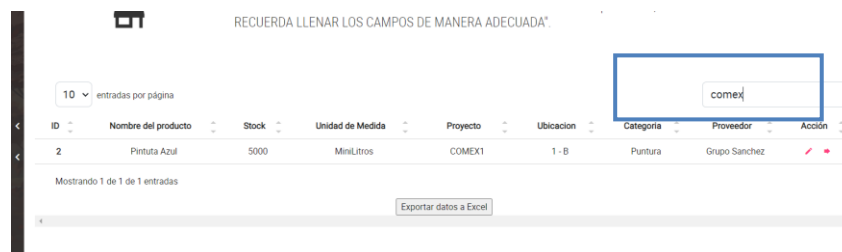


Figura. 49 Ejemplo de búsqueda por proyecto. Fuente. Elaboración propia.

- **Requisito funcional 10: Imprime y/o genera reporte de Excel del inventario**

Se podrá generar y descargar un reporte de las listas de productos registrados generando un archivo de Excel incluyendo todos los datos del inventario.

- **Caso de uso específico de requisito funcional 10**

En la figura 50 muestra el caso de uso de la funcionalidad de imprimir / generar reporte en el módulo de inventario.

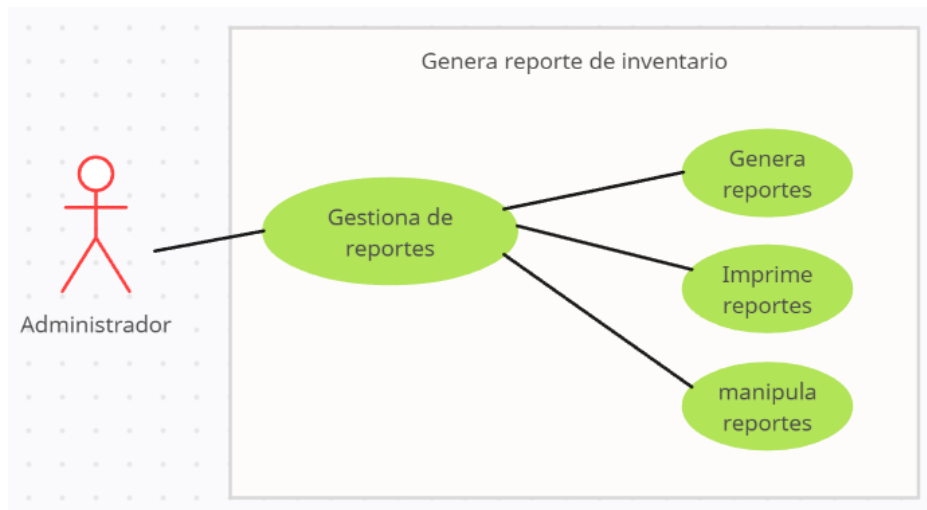


Figura. 50 Caso de uso específico Imprime/ genera reporte. Fuente. Elaboración propia.

- **Escenario de requisito funcional 10**

La tabla 13 muestra el escenario de imprimir y generar un reporte del inventario la cual describe precondiciones, flujo normal y alterno, así como las post condiciones.

Tabla 13 Escenario de requisito funcional 10. Fuente. Elaboración propia

Número del requisito:	10
Nombre del requisito:	Imprime y/o genera reporte de Excel del inventario.
Prioridad:	Media
Descripción:	El usuario administrador encargado del inventario podrá generar un reporte y generarlo en Excel para imprimir.
Actores:	Administrador.

Precondiciones:	El usuario deberá iniciar sesión como administrador. Deberá existir un inventario de productos para poder generar el reporte actualizado.
Flujo Normal:	1: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión como administrador. 4: Al ingresar al sistema navegar al módulo de inventario. 5: En la parte final de la lista de inventarios dará clic en el botón que dice “Exportar datos a Excel.”
Flujo Alterno:	1: Si excede del tiempo de sesión activa aparecerá el mensaje de tiempo excedido y deberá ingresar al sistema nuevamente para poder generar el reporte en Excel.
Post condiciones:	1: El sistema no realiza cambios en el inventario como resultado de la consulta.

Pantallas de flujo normal Requisito funcional 10

Se ingresa al módulo de inventario como se muestra en la pantalla de la figura 51.

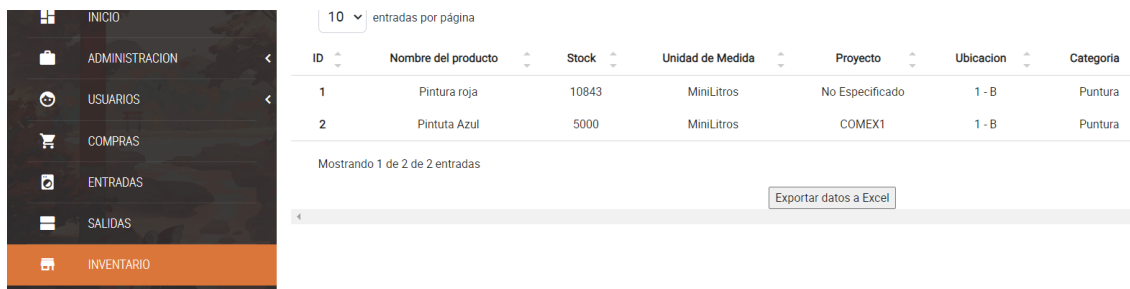


Figura. 51 Módulo de inventario. Fuente. Elaboración propia.

Al querer generar un reporte se deberá dar clic en el botón “Exportar datos a Excel” , botón que se puede visualizar en la imagen 52.

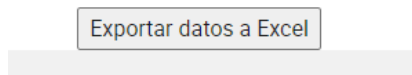


Figura. 52 Botón de exportar datos. Fuente. Elaboración propia.

Ejemplo de descarga de reporte de un archivo con extensión .xlsx en figura 53.

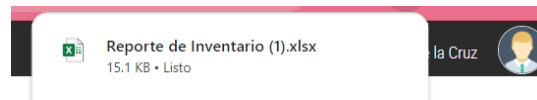


Figura. 53 Descarga de archivo .xlsx. Fuente. Elaboración propia.

Reporte generado en Excel para poder imprimir como se muestra en la figura 54

ID	Nombre de	Stock	Unidad de	Proyecto	Ubicación	Categoría	Proveedor	Acción
1	Pintura roja	10843	MinilLitros	No Especifica 1 - B	Puntura	Grupo Sanchez		
2	Pintura Azul	5000	MinilLitros	COMEX1	1 - B	Grupo Sanchez		

Figura. 54 Archivo en Excel. Fuente. Elaboración propia.

- Requisito funcional 11: Registrar proveedores

Usuario con privilegios de administrador puede agregar una lista de proveedores, en este módulo se requiere llenar datos obligatoriamente como lo son el nombre del proveedor, datos de contacto como dirección, teléfono, correo electrónico, sin dejar ningún campo vacío de lo contrario no permitirá el registro.

- Caso de uso específico de requisito funcional 11

En la figura 55 se muestra el caso de uso específico de la función registrar proveedores.



Figura. 55 Caso de uso específico de registrar proveedores. Fuente. Elaboración propia.

- **Escenario de requisito funcional 11**

La tabla 14 muestra el escenario de registrar a un nuevo proveedor y generar un reporte del inventario la cual describe precondiciones, flujo normal y alterno, así como las post condiciones.

Tabla 14 Escenario de requisito funcional 11. Elaboración propia.

Número del requisito:	11
Nombre del requisito:	Registrar nuevo proveedor
Prioridad:	Alta
Descripción:	El usuario administrador puede registrar un nuevo proveedor en administración en el módulo de proveedores.
Actores:	Administrador
Precondiciones:	El usuario que realizará esta función deberá iniciar sesión, como administrador.
Flujo Normal:	1: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión. 4: Al ingresar al sistema navegar a la sección de administración y hacer clic. 5: Se desplegará una lista donde se deberá dar clic en “Proveedores” 6: El sistema mostrará un formulario en donde deberá llenar los siguientes datos: Nombre del proveedor, dirección lo cual consta de seleccionar país, calle C.P estado, colonia, ciudad, teléfono y correo electrónico. 7: Dar clic en el icono de + para poder registrar un nuevo producto.
Flujo Alterno:	1: Si el usuario excede de tiempo se cerrará sesión u sus cambios o serán guardados. 2: Si algún campo del formulario no se completa correctamente el sistema no dejara hacer ningún cambio, regresando a la pantalla de registro nuevamente hasta que se llene correctamente el formulario. 3: el usuario no tiene privilegios de agregar información.
Post condiciones:	1: Las bases de datos del inventario se deberán actualizar con la información del nuevo proveedor.

Pantallas de flujo normal Requisito funcional 11

El usuario despliega la lista de administración para poder visualizar proveedores, posicionando el cursor como se ve en la figura 56.



Figura. 56 Selección de administración para agregar proveedores. Fuente. Elaboración propia.

Al entrar se mostrará la siguiente pantalla para agregar un nuevo proveedor (figura 57, 58).

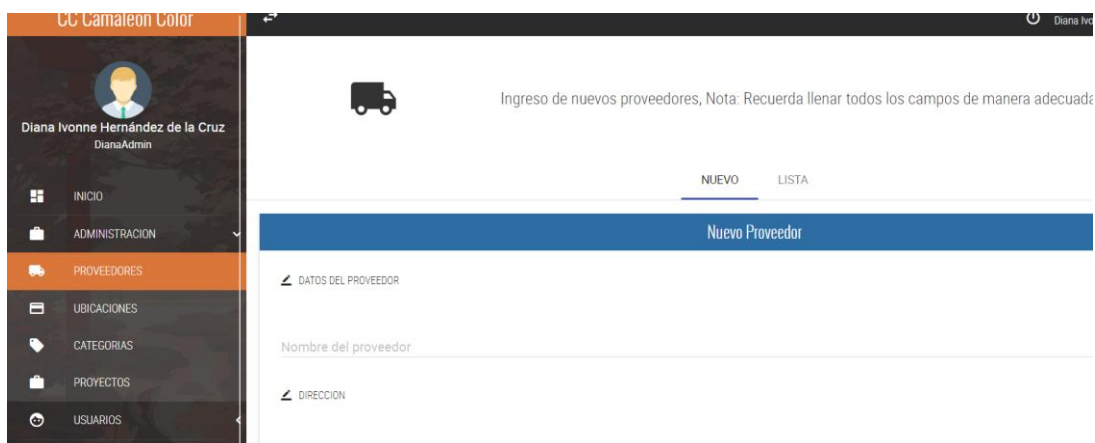


Figura. 57 Módulo y formulario de agregar proveedor. Fuente. Elaboración propia.

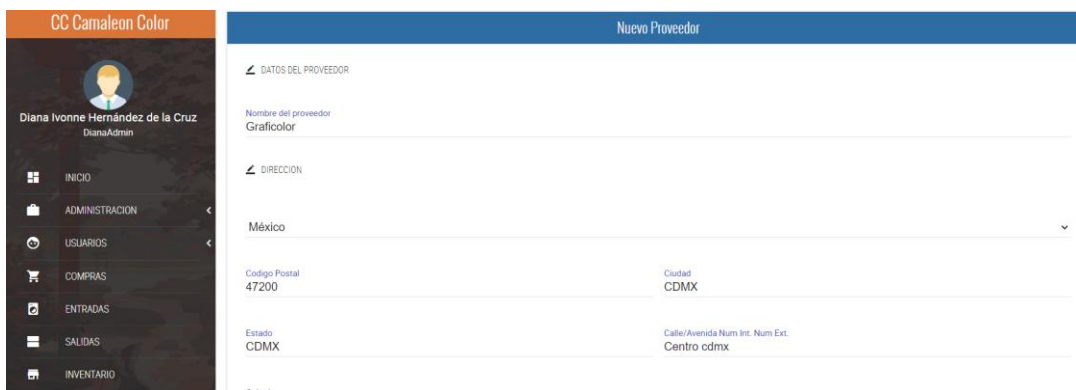


Figura. 58 Agregar datos de proveedor nuevo. Fuente. Elaboración propia.

Se deberán llenar todos los datos del formulario para poder ser registrado y dar clic en el símbolo de +

Para verificar el registro dar clic en “Lista” pantalla que se muestra en la figura 59.

ID	Nombre	Pais	CP	Ciudad	Estado	Calle	Colonia	Correo	Telefono	Acción
1	provtest	Unión Europea	55400	mexico	Indica la provincia	Cerrada de Fresno Mz5 L5	Ejidos de santa maria Tulpetlac	aracelimercado01@gmail.com	5621503696	
2	Grupo Sanchez	México	06880	CDMX	CMDX	Isabela La Catolica No 310	Algarin	2907SDIS01@sanchez.com.mx	5555383800	
3	graficolor	México	47200	CDMX	CDMX	Centro cdmx	Centro	graficolor@outlook.com	5547586547	

Mostrando 1 de 3 de 3 entradas

Exportar datos a Excel

Figura. 59 actualización de lista de proveedores. Fuente Elaboración propia.

- Requisito funcional 12: Eliminar y edita datos de proveedores

Se agrupan estas dos funciones ya que tiene privilegio el usuario administrador de actualizar y editar los datos del proveedor según sea conveniente, ya sea por cambio de dirección, razón social, cambio de teléfono etc., así como la eliminación de un proveedor en caso de ya o trabajar con ellos o asignarlo como inactivo.

- Caso de uso específico de requisito funcional 12

En la figura 60 se muestra el caso de uso del requisito funcional de editar y eliminar un proveedor.

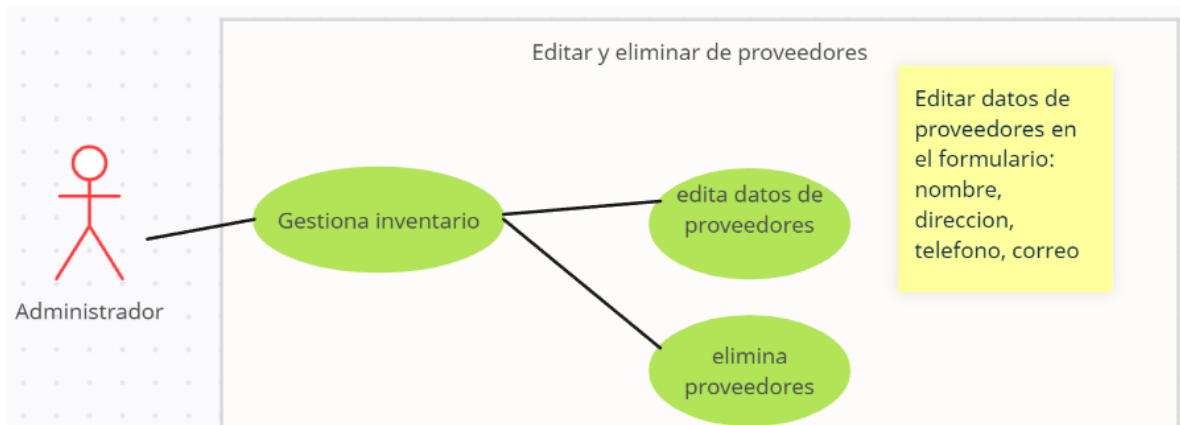


Figura. 60 Caso de uso específico de Editar y eliminar proveedor. Fuente. Elaboración propia.

- **Escenario de requisito funcional 12**

La tabla 15 muestra el escenario de editar y eliminar proveedores la cual describe precondiciones, flujo normal y alterno, así como las post condiciones.

Tabla 15 Escenario de requisito funcional 12. Elaboración propia.

Número del requisito:	12
Nombre del requisito:	Editar y eliminar proveedores.
Prioridad:	Media
Descripción:	El usuario administrador encargado del inventario podrá editar y eliminar los datos de los proveedores registrados previamente.
Actores:	Administrador
Precondiciones:	El usuario deberá iniciar sesión como administrador Deberá existir al menos un proveedor previamente registrado para poder ser editado o eliminado.
Flujo Normal:	1: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión. 4: Al ingresar al sistema navegar al módulo de administración. 5: Desplegar la lista y seleccionar proveedores 6: En lugar de agregar un nuevo proveedor el usuario deberá dar clic en “Lista” 7: Se mostrará la lista de los proveedores registrados previamente. 8: El usuario selecciona el proveedor que desee editar, o eliminar y deberá dar clic en el icono con símbolo de lápiz y para eliminarlo seleccionar el icono de bote de basura. 9: Edita según le sea conveniente, se habilitará para edición el nombre de proveedor, país C.P, calle, estado, teléfono y correo. 10: para eliminar proveedor solo tendrá que dar clic en “Lista” y seleccionar el producto que desee eliminar, posteriormente dar clic en el icono de bote de basura.
Flujo Alterno:	1: Deberá existir al menos un proveedor que desee editar o eliminar. 2: Si se excede el tiempo de inicio de sesión aparecerá el mensaje de tiempo excedido y sus cambios no se guardarán. 3: Si no se llena algún cambio, no se guardarán los cambios y no permitirá continuar el proceso de edición de datos de proveedor.
Post condiciones:	1: El proveedor editado o eliminado se refleja automáticamente en la base de datos y las listas.

Pantallas de flujo normal Requisito funcional 12

Para editar un proveedor navegara en el menú principal y seleccionar administración desplegar la lista y seleccionar proveedores, posteriormente dar clic en “LISTA” pantalla que se refleja en la figura 61.



Figura. 61 Lista de proveedores. Fuente. Elaboración propia.

ID	Nombre	Pais	CP	Ciudad	Estado	Calle	Colonia	Correo	Telefono	Acción
1	provtest	Unión Europea	55400	mexico	Indica la provincia	Cerrada de Fresno Mz5 L5	Ejidos de santa maria Tulpetic	aracelmercado01@gmail.com	5621503696	 
2	Grupo Sanchez	México	06880	CDMX	CDMX	Isabella La Católica No 516	Algarin	2907SDIS01@sanchez.com.mx	5555383800	 
3	Graficolor	México	47200	CDMX	CDMX	Centro cdmx	Centro	graficolor@outlook.com	5547386547	 

Figura. 62 Icono para editar proveedor.

Identificar el proveedor que desee editar, y dar clic en el icono de lápiz como se muestra en la figura 62.

Posteriormente el usuario va a poder visualizar la interfaz de edición de proveedor (Figura 63).



Figura. 63 Editar datos de proveedor. Fuente. Elaboración propia.

Dará clic en el icono  y se actualizarán los datos, realizada esta acción verificar los cambios en la lista de proveedores, como se muestra en la figura 64.

Lista de Proveedores										
10 ▼ entradas por página		Buscar...								
ID	Nombre	País	CP	Ciudad	Estado	Calle	Colonia	Correo	Telefono	Acción
1	Pintu Acuario	México	55400	mexico	Indica la provincia	Cerrada de Fresno Mz5 L5	Ejidos de santa maria Tulpetlac	aracelmercado01@gmail.com	5621503696	

Figura. 64 Actualización de datos de proveedores. Fuente. Elaboración propia.

Para eliminar a algún proveedor bastara con regresar a la lista de proveedores, identificar al proveedor que desee eliminar y seleccionar el icono de bote de basura, como se refleja en la figura 65.

Lista de Proveedores										
10 ▼ entradas por página		Buscar...								
ID	Nombre	País	CP	Ciudad	Estado	Calle	Colonia	Correo	Telefono	Acción
1	Pintu Acuario	México	55400	mexico	Indica la provincia	Cerrada de Fresno Mz5 L5	Ejidos de santa maria Tulpetlac		6	 
2	Grupo Sanchez	México	06880	CDMX	CMDX	Isabela La Catolica No 516	Algarin		0	 
3	Graficolor	México	47200	CDMX	CDMX	Centro cdmx	Centro		7	 

Mostrando 1 de 3 de 3 entradas

[Exportar datos a Excel](#)

Figura. 65 Selección de borrar registro de proveedor. Fuente. Elaboración propia.

Mensaje de confirmación de eliminación de proveedor (Figura 66).

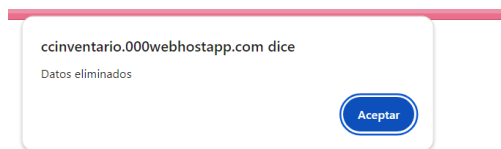


Figura. 66 Confirmación de eliminación. Fuente. Elaboración propia.

Dar clic en el botón de aceptar y regresar a la “LISTA” de proveedores (Figura 67).

Lista de Proveedores										
10 ▼ entradas por página		Buscar...								
ID	Nombre	País	CP	Ciudad	Estado	Calle	Colonia	Correo	Telefono	Acción
1	Pintu Acuario	México	55400	mexico	Indica la provincia	Cerrada de Fresno Mz5 L5	Ejidos de santa maria Tulpetlac	aracelmercado01@gmail.com	5621503696	
2	Grupo Sanchez	México	06880	CDMX	CMDX	Isabela La Catolica No 516	Algarin	2907SDIS01@sanchez.com.mx	5555383800	

Mostrando 1 de 2 de 2 entradas

[Exportar datos a Excel](#)

Figura. 67 Lista actualizada de datos eliminados de proveedores. Fuente. Elaboración propia.

- **Requisito funcional 13: Registro de salidas**

El inventario cuenta con una lista de los productos registrados y disponibles en el almacén, de los cuales cuando el usuario necesite hacer una salida al momento que necesiten un utilizar un producto, quedara registrado en el módulo de “salidas”.

- **Caso de uso específico de requisito funcional 13**

En la figura 68 se muestra el caso específico de la función registro de salidas.

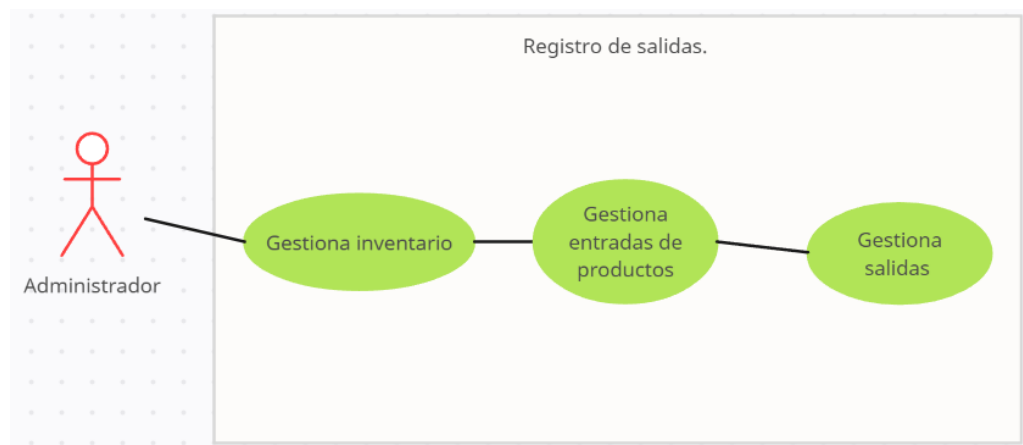


Figura. 68 Caso de uso de registro de salidas. Fuente. Creately, Elaboración propia.

- **Escenario de requisito funcional 13**

La tabla 16 muestra el escenario de registro de salidas la cual describe precondiciones, flujo normal y alterno, así como las post condiciones.

Tabla 16 Escenario de requisito funcional 13. Elaboración propia.

Número del requisito:	13
Nombre del requisito:	Registro de salidas.
Prioridad:	Alta
Descripción:	El usuario administrador encargado del inventario o usuario operativo podrá ingresar el registro de compras nuevas.
Actores:	Administrador / Operativo
Precondiciones:	El usuario deberá iniciar sesión como administrador o usuario operativo. Deberá existir la compra o el ingreso del inventario previamente registrado para poder realizar la salida del producto.

Flujo Normal:	1: En la primera interfaz que podrá ver será la pantalla de inicio de sesión mostrando el formulario que se deberá llenar. 2: Se ingresa correo y contraseña para el inicio de sesión en el caso de administrador, y en caso de usuario operativo con el nombre de usuario proporcionado por el administrador y contraseña igualmente proporcionado por el supervisor administrador. 4: Al ingresar al sistema navegar al módulo de inventario en el menú principal. 5: Se registra la salida cuando selecciona el icono de flecha dirigida a la derecha en el producto que desee retirar. 6: Lo único que se habilitara en este apartado es la cantidad de producto que va a retirar para su uso en mililitros o en gramos o piezas.
Flujo Alterno:	1: Deberá existir el producto para que se pueda realizar la salida del producto a utilizar y dará clic en el icono de flecha para realizar la salida. 2: Lo dirige a la interfaz de lista de historial de salidas. 3: Si el usuario no llena el único campo de cantidad a retirar con número no se efectuará el cambio.
Post condiciones:	La salida de producto queda registrada en el módulo de “salidas” refleja en las listas y registra en la base de datos.

Pantallas de flujo normal Requisito funcional 13

En la lista de productos del inventario, identificar cual es el producto que desea darle salida, posteriormente dar clic en el icono de la Flecha, como se muestra en la figura 69.

ID	Nombre del producto	Stock	Unidad de Medida	Proyecto	Ubicacion	Categoría	Proveedor	Acción
1	Pintura roja	10843	MiniLitros	No Especificado	1 - B	Pintura	Grupo Sanchez	

Figura. 69 Seleccionar icono para realizar salida de producto. Fuente. Elaboración propia.

El usuario podrá ver la siguiente interfaz después de dar clic en el icono mencionado, deberá ingresar con número la cantidad a retirar y dar clic en la flecha color azul (Figura 70).

SALIDA DE PRODUCTO

⌵

DATOS DEL PRODUCTO

Pintura roja

10/03/2024

📅 Grupo Sanchez

⌵

USUARIO

Selecciona Usuario

⌵

CANTIDAD A RETIRAR

10843

50

➔

Figura. 70 Dar salida al producto que selecciono. Fuente. Elaboración propia.

Se reflejará automáticamente en el historial de “Salidas” (Figura 71).

CC Camaleon Color

Diana Ivonne Hernández de la Cruz

DianaAdmin

INICIO

ADMINISTRACION

USUARIOS

COMPRAS

ENTRADAS

SALIDAS

INVENTARIO

🛒

NOTA: Recuerda hacer un buen uso del inventario y registrar las salidas correspondientes

10

entradas por página

Buscar...

ID	Usuario	Producto	Proveedor	Cantidad Retirada	Fecha de Salida
1	Marco Alexis	marco	provtest	2	2024-01-25
2	Diana Ivonne	Pintura	Grupo Sanchez	1000	2024-02-03
3	marco	marco	provtest	1	2024-03-10
4	pintar1	marco	provtest	1	2024-03-10
5		Pintura roja	Grupo Sanchez	50	2024-03-10

Mostrando 1 de 5 de 5 entradas

Exportar datos a Excel

Exportar datos a PDF

Figura. 71 Interfaz de registro de salidas de productos. Fuente. Elaboración propia.

3.9.2. Requisitos no funcionales específicos

- Requisito no funcional 1: Integridad de datos en el registro

En el sistema se implementan medidas para salvaguardar la integridad de datos mediante el registro y modificación de datos. Al realizar una validación en los campos numéricos, especialmente aquellos asociados con cantidades, para asegurar que cumplan con la restricción de una longitud máxima de 2 caracteres, y no se ingresen negativos en el caso de las compras, se pueda generar la operación por si sola al dar el total de la compra, se consideran en todos los campos de registro no hayan campos vacíos para evitar errores posibles al introducción la información de esta manera cuenta con la coherencia y consistencia durante la fase de registro y modificación de datos.

The screenshot shows a purchase form with the following elements and annotations:

- Medida:** A dropdown menu currently showing "Gramos".
- INGRESO CANTIDAD EN: MINILITRO, GRAMOS O PIEZAS:** A label above the quantity input field.
- Ingrese la Cantidad:** The text inside the quantity input field.
- FINALIZACION DE COMPRA:** A label above the final calculation row.
- Ingrese la cantidad:** The text in the first column of the final row, with the value "-87" entered.
- Precio Unitario:** The text in the second column of the final row, with the value "90" entered.
- Result:** The value "-9082.80" displayed in the third column of the final row.
- Annotations (red arrows):**
 - "Campo vacio" points to the "Medida" dropdown.
 - "Ingreso de cantidad negativa" points to the "-87" in the quantity field.
 - "Operación automática errónea por datos negativos" points to the "-9082.80" result.

Figura. 72 Ingreso de cantidades negativas, Fuente. Elaboración propia.

La prueba de integridad al ingresar datos para que el usuario no pueda ingresar cantidades negativas se muestra en la figura 73.

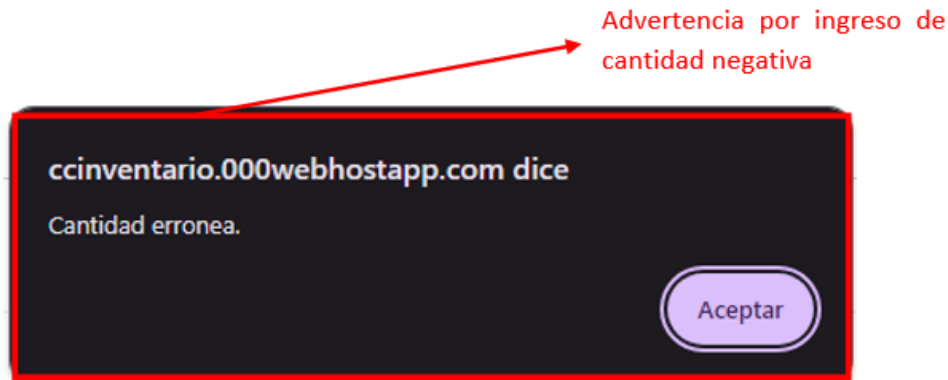


Figura. 73 Advertencia por el ingreso de cantidades negativas, Fuente. Elaboración propia.

Por lo que también muestra un mensaje de advertencia si algún campo está vacío

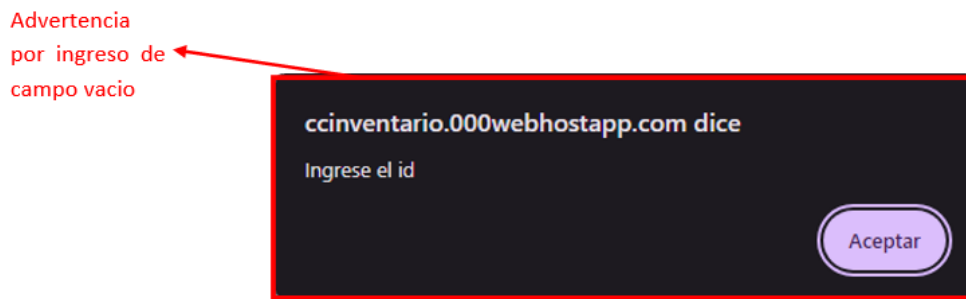


Figura. 74 Advertencia de campo vacío. Fuente. Elaboración propia

En la figura 75 se muestra el ejemplo de datos llenados correctamente

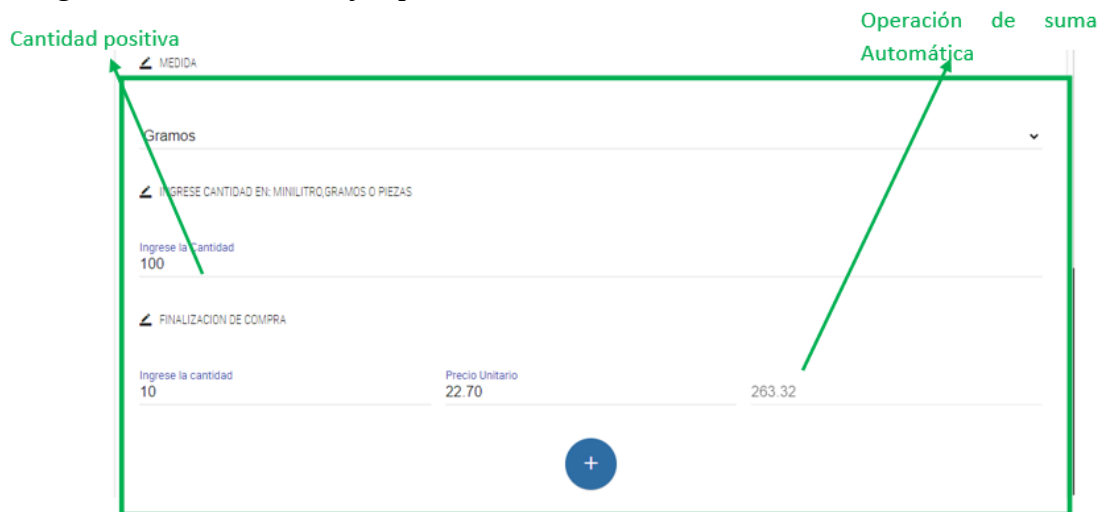


Figura. 75 Ingreso de una operación correcta, Fuente: Elaboración propia

- **Requisito no funcional 2: No repudio en el registro**

Con el propósito de conseguir el no repudio en los procesos de editar y eliminar el sistema procederá de manera automatizada a generar un registro. Dicho registro incorporara información ingresada de manera precisa de la modificación que se hizo, así como los datos específicos que han sido objeto de alteración.

Pantallas de requisito no funcional 2 No repudio en el registro

En la figura 76 un ejemplo de registros en la tabla de usuarios administradores.

Botones de acción:
Eliminar/Editar

Registro almacenado

Id	Nombre(s)	Apellido(s)	Usuario	Correo	Contraseña	Acción
1						[Eliminar] [Editar]
2						[Eliminar] [Editar]
3	prueba	documentacion	documento	noesreal@gmail.com	12345678	[Eliminar] [Editar]

Mostrando 1 de 3 de 3 entradas

Exportar datos a Excel

Figura. 76 Registros de tabla administradores, Fuente. Elaboración propia

Figura 77 se muestra la información modificada para un usuario administrador

Ingreso de datos modificados correctos.

Edición de Administrador

⏏ DATOS DEL ADMINISTRADOR

Nombre(s)
pruebaedicioncompleta

Apellido(s)
documentacion

Correo Electronico
noesreal@gmail.com

⏏ DETALLES DE LA CUENTA

Usuario
documento

Contraseña

+

Figura. 77 Datos modificados de tabla administradores, Fuente. Elaboración propia

Se muestra en la figura 78 que se resalta en color rojo las letras en forma de advertencia para revisar si se ingresó mal un dato.

The screenshot shows a web form titled 'Edición de Administrador' with tabs for 'EDITAR' and 'LISTA'. The form is divided into two sections: 'DATOS DEL ADMINISTRADOR' and 'DETALLES DE LA CUENTA'. In the first section, the 'Nombre(s)' field contains 'prueba' and the 'Apellido(s)' field contains 'documentacion.'. Both fields have red text 'Apellido Invalido' below them. The 'Correo Electronico' field contains 'noesrealgmail.com' and has a red text 'Correo Invalido' below it. In the second section, the 'Usuario' and 'Contraseña' fields are visible. Red arrows point from external labels to these error messages: 'Campos Editables' points to the form fields, 'Advertencia de violación del formato' points to the red error text, and 'Campo vacío' points to the empty 'Usuario' field.

Figura. 78 Advertencia por ingreso erróneo, Fuente. Elaboración propia

Figura 79 muestra un ejemplo que está mal el formato

This close-up shows the 'Apellido(s)' field with the text 'documentacion.'. Below the field, the text 'Apellido Invalido' is displayed. A red arrow points from the label 'Advertencia de formato incorrecto' to the error message. A tooltip box with an exclamation mark icon contains the text 'Haz coincidir el formato solicitado.'

Figura. 79 Ejemplo de formato inválido, Fuente. Elaboración propia.

- Requisito no funcional 3: Adaptabilidad en dispositivos

Con la finalidad de que el sistema coexista de manera correcta dentro de los diferentes dispositivos capaces de mostrar una página web, se opta por la implementación de la configuración para la visualización del contenido efectiva en una variedad de dispositivos, incluyendo computadoras de escritorio, laptops, tabletas y teléfonos móviles, proporcionando una experiencia de usuario consistente.

Pantallas de requisito no funcional 3 Adaptabilidad en dispositivos.

En la figura 80 se muestra la vista desde un ordenador



Figura. 80 Vista del sistema desde un ordenador, Fuente. Elaboración propia

Figura 81 vista desde un dispositivo móvil

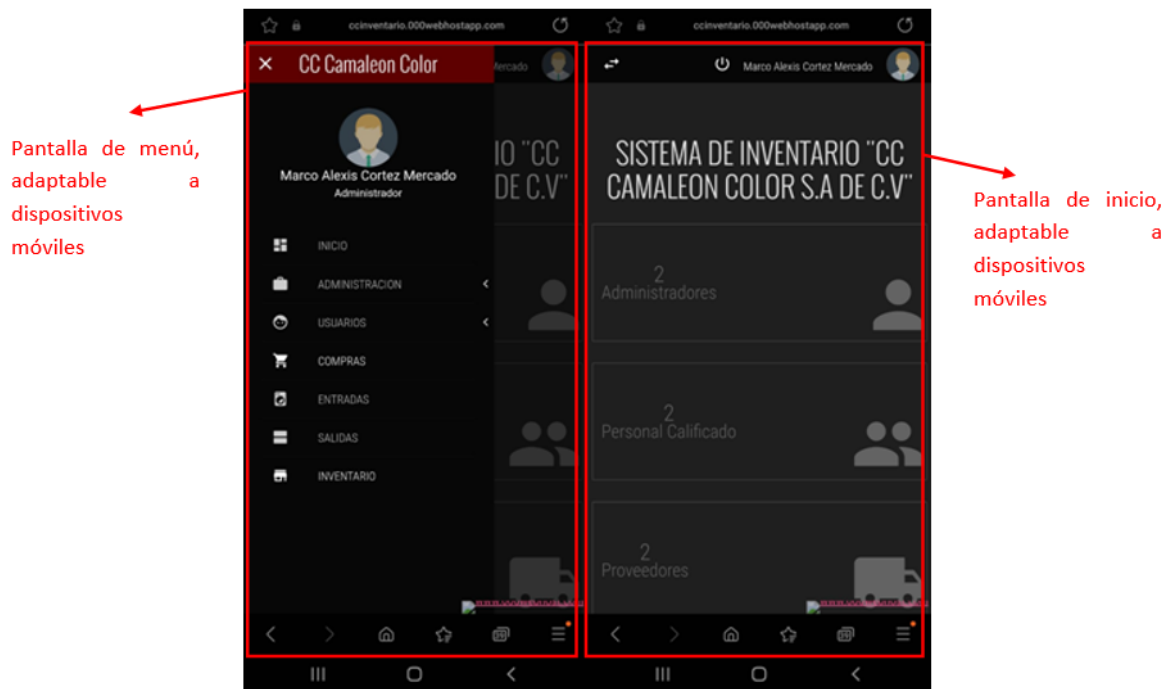


Figura. 81 Vista desde dispositivo móvil, Fuente. Elaboración propia

3.10. DESARROLLO DE CODIGO

3.10.1. Creación de Base de datos:

Para dar comienzo a la programación del sistema web, es fundamental dar creación a la base de datos que almacenará la información correspondiente del sistema, para esto se deberá de iniciar sesión con una cuenta dentro de la plataforma de Google Firebase.

Una vez dentro de la página se dará clic en el apartado de “comenzar” para continuar con el proceso del registro como se observa en la figura 82.

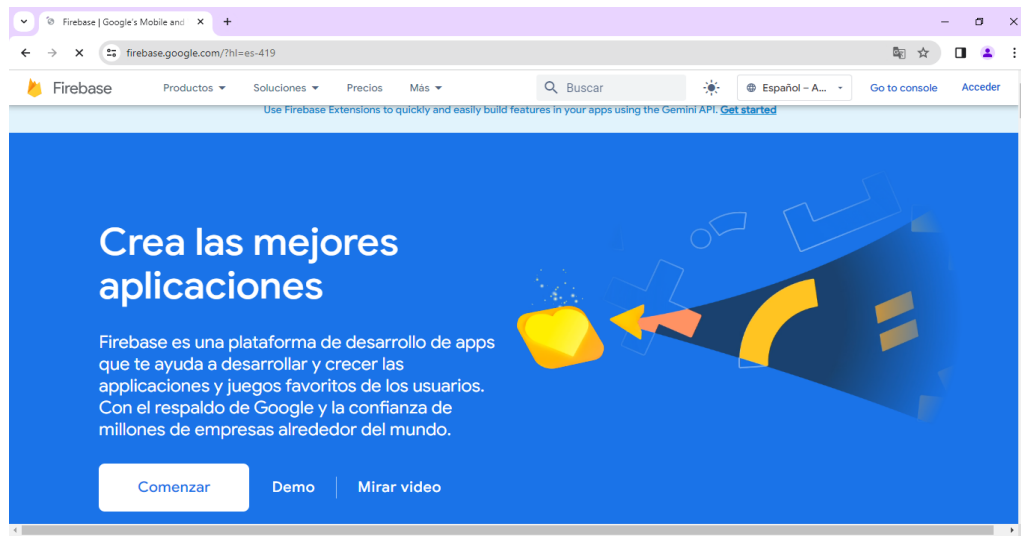


Figura. 82 Comienzo a Firebase. Fuente. Fuente. Elaboración propia.

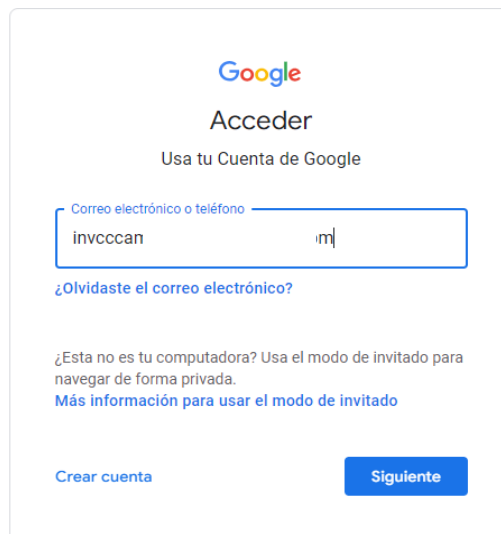


Figura. 83 Acceder a la cuenta de Google para crear proyecto en Firebase

Autenticarse con una cuenta de Gmail como se muestra en la figura 83, realizando esto se mostrará una nueva interfaz en la cual permitirá la creación de nuevos proyectos para el uso de Firebase, posteriormente se dará clic en agregar nuevo proyecto. (figura 84).

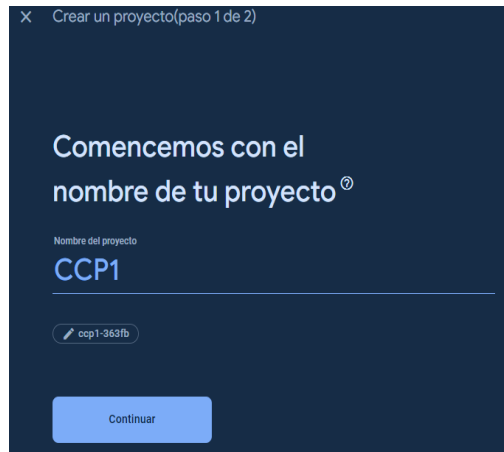


Figura. 84 Nombre del nuevo proyecto Fuente. Elaboración propia,

Al crear un nuevo proyecto como se muestra en de la figura 85, se llena el campo del nombre del proyecto y finaliza con la acción de botón “crear proyecto”.

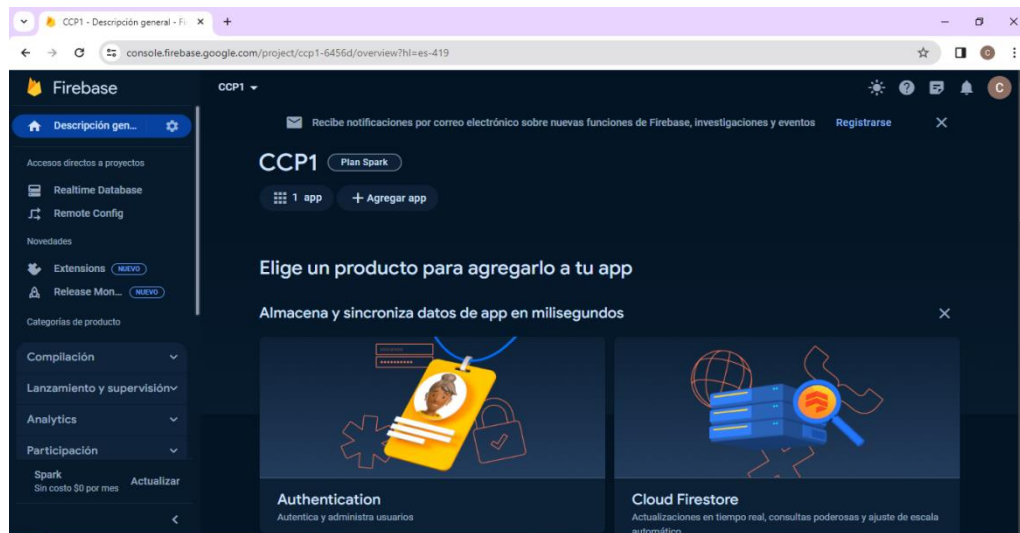
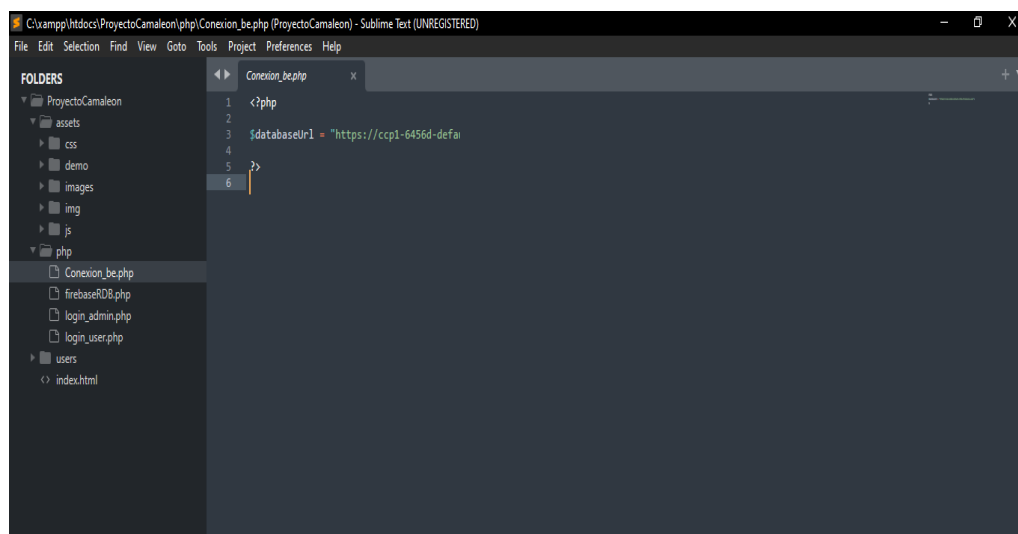


Figura. 85 Interfaz de configuración Fuente Elaboración propia.

Una vez finalizada la creación se vera la interfaz de Firebase del nuevo proyecto mostrada en la figura 85, se muestra la interfaz de configuración, finalizando la creación del proyecto y hacer uso de la base de datos en tiempo real.

En carpeta nombrada “PHP” se encuentra un archivo con el nombre de “Conexión_be.php” en el cual se encuentra declarado en lenguaje PHP y guardado en la variable “databaseurl” el enlace de vinculación a la base de datos de Firebase así como se muestra en la figura 87.



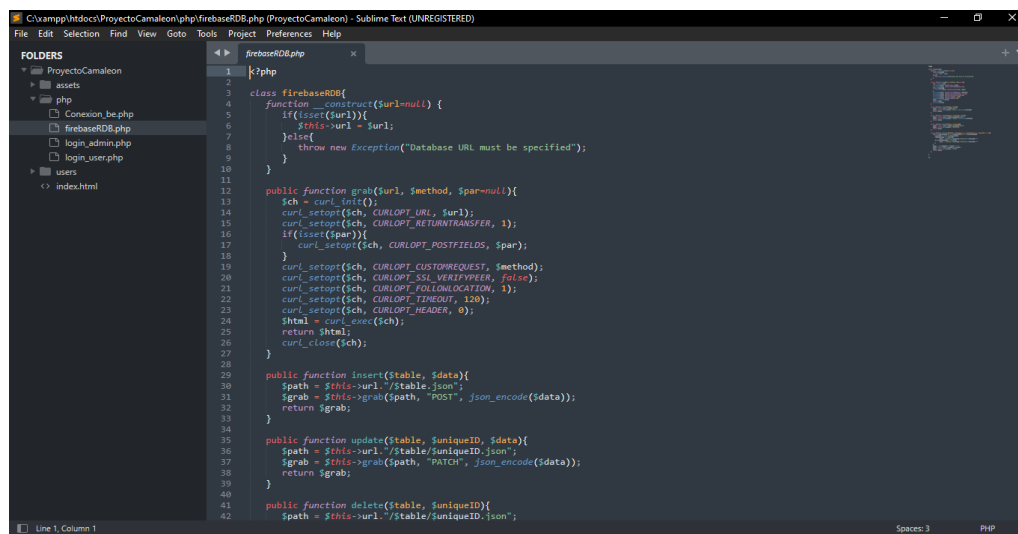
```

1 <?php
2
3 $databaseurl = "https://ccp1-6496d-defai
4
5 }>
6

```

Figura. 86 URL de conexión a Firebase. Fuente. Elaboración propia.

Dentro de la misma carpeta de PHP se encuentra el archivo “firebaseRDB.php” que contiene la estructura y constructor para los métodos de inserción, actualización, búsqueda y eliminación de información de la base de datos como se muestra en la figura 87.



```

1 <?php
2
3 class firebaseRDB{
4     function __construct($url=null) {
5         if(isset($url)){
6             $this->url = $url;
7         }else{
8             throw new Exception("Database URL must be specified");
9         }
10    }
11
12    public function grab($url, $method, $par=null){
13        $ch = curl_init();
14        curl_setopt($ch, CURLOPT_URL, $url);
15        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
16        if(isset($par)){
17            curl_setopt($ch, CURLOPT_POSTFIELDS, $par);
18        }
19        curl_setopt($ch, CURLOPT_CUSTOMREQUEST, $method);
20        curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
21        curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
22        curl_setopt($ch, CURLOPT_TIMEOUT, 120);
23        curl_setopt($ch, CURLOPT_HEADER, 0);
24        $html = curl_exec($ch);
25        return $html;
26        curl_close($ch);
27    }
28
29    public function insert($table, $data){
30        $path = $this->url."/{$table}.json";
31        $grab = $this->grab($path, "POST", json_encode($data));
32        return $grab;
33    }
34
35    public function update($table, $uniqueID, $data){
36        $path = $this->url."/{$table}/{$uniqueID}.json";
37        $grab = $this->grab($path, "PATCH", json_encode($data));
38        return $grab;
39    }
40
41    public function delete($table, $uniqueID){
42        $path = $this->url."/{$table}/{$uniqueID}.json";

```

Figura. 87 Estructura de conexión a la Base de Datos, Fuente. Elaboración propia.

3.10.2. Autenticación:

La implementación de un sistema de autenticación robusto y seguro es esencial en el desarrollo de sistemas web.

Se ingresa dentro del editor de código “Sublime Text” donde se visualiza dentro del apartado izquierdo las carpetas que almacenan los archivos, señalado en la figura 88.

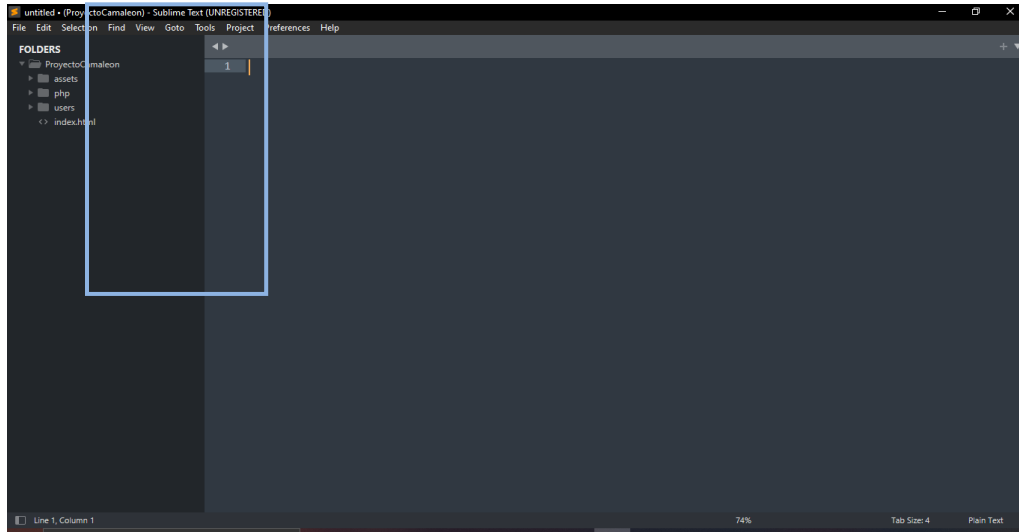


Figura. 88 Interfaz del editor de texto Sublime Text, Fuente. Elaboración propia.

En carpeta principal nombrada ProyectoCamaleon mostrada se muestra en la figura 89, se encuentra el archivo denominado “index.html” dicho archivo contiene la codificación del cuerpo de la función autenticarse en el sistema

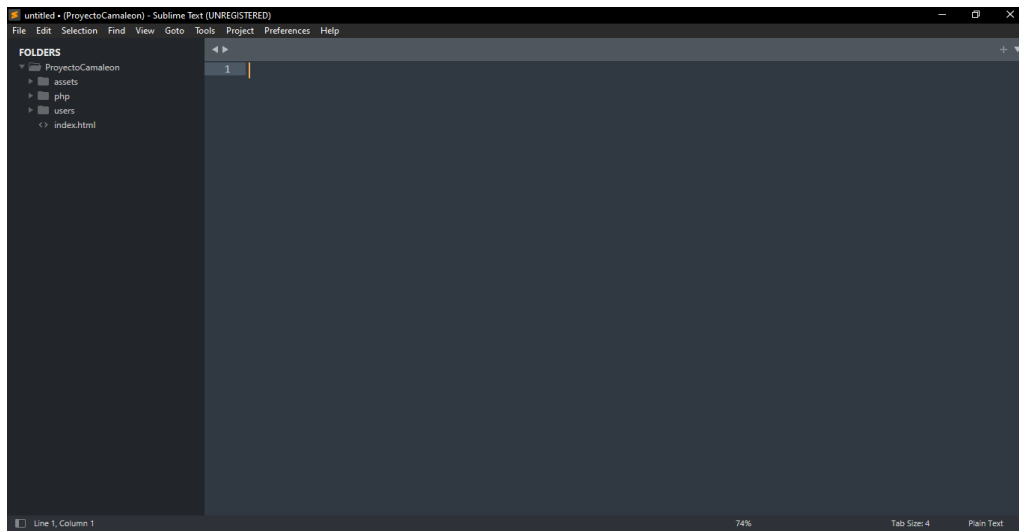
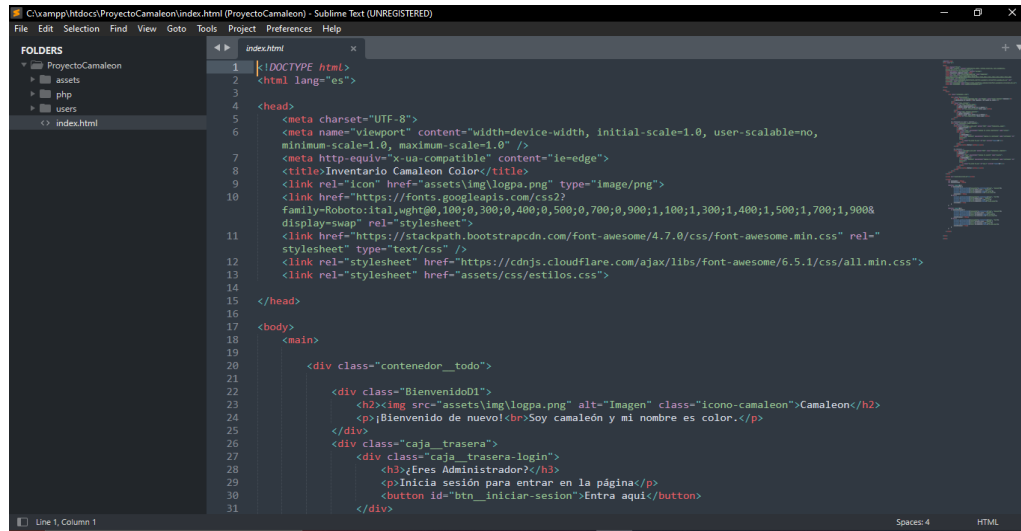


Figura. 89 Cuerpo de la carpeta Proyecto Camaleón. Fuente. Elaboración propia.

La siguiente pantalla (figura 90) muestra el archivo “index.html”, se encuentra la estructura de `!DOCTYPE html>` la cual informara al navegador el tipo de versión HTML que fue implementado en el archivo, a su vez se indica con una etiqueta de tipo lang el tipo de idioma con el que será escrita la página web en este caso se hace uso del idioma español.



```

1 <!DOCTYPE html>
2 <html lang="es">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no,
7     minimum-scale=1.0, maximum-scale=1.0" />
8   <meta http-equiv="x-ua-compatible" content="ie=edge">
9   <title>Inventario Camaleon Color</title>
10  <link rel="icon" href="assets/img/logpa.png" type="image/png">
11  <link href="https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,100;0,300;0,400;0,500;0,700;0,900;1,100;1,300;1,400;1,500;1,700;1,900&
12    display=swap" rel="stylesheet">
13  <link href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="
14    stylesheet" type="text/css" />
15  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css">
16  <link rel="stylesheet" href="assets/css/estilos.css">
17 </head>
18 <body>
19   <main>
20     <div class="contenedor_todo">
21       <div class="BienvenidoD1">
22         <h2>Camaleon</h2>
23         <p>¡Bienvenido de nuevo! Soy camaleón y mi nombre es color.</p>
24       </div>
25       <div class="caja_trasera">
26         <div class="caja_trasera_login">
27           <h3>Eres Administrador?</h3>
28           <p>Inicia sesión para entrar en la página.</p>
29           <button id="btn_iniciar-sesion">Entra aquí</button>
30         </div>
31       </div>

```

Figura. 90 Estructura de lenguaje HTML e Idioma del archivo index.html. Fuente. Elaboración propia.

Posteriormente se muestra la apertura de un head en el cual se proveerá la información como lo es el título y los enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assets” y de las subcarpetas de “img” y “css”, así como se observa en la figura 91.



```

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no,
  minimum-scale=1.0, maximum-scale=1.0" />
  <meta http-equiv="x-ua-compatible" content="ie=edge">
  <title>Inventario Camaleon Color</title>
  <link rel="icon" href="assets/img/logpa.png" type="image/png">
  <link href="https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,100;0,300;0,400;0,500;0,700;0,900;1,100;1,300;1,400;1,500;1,700;1,900&
  display=swap" rel="stylesheet">
  <link href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" rel="
  stylesheet" type="text/css" />
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css">
  <link rel="stylesheet" href="assets/css/estilos.css">
</head>

```

Figura. 91 Ingreso de título y enlaces de funciones del archivo index.html. Fuente. Elaboración propia.

Como se muestra en la figura 92, dentro de la carpeta de “assets” por la parte de las imágenes se encuentra la subcarpeta “img” la cual almacena las imágenes correspondientes para el uso de la función autenticarse.

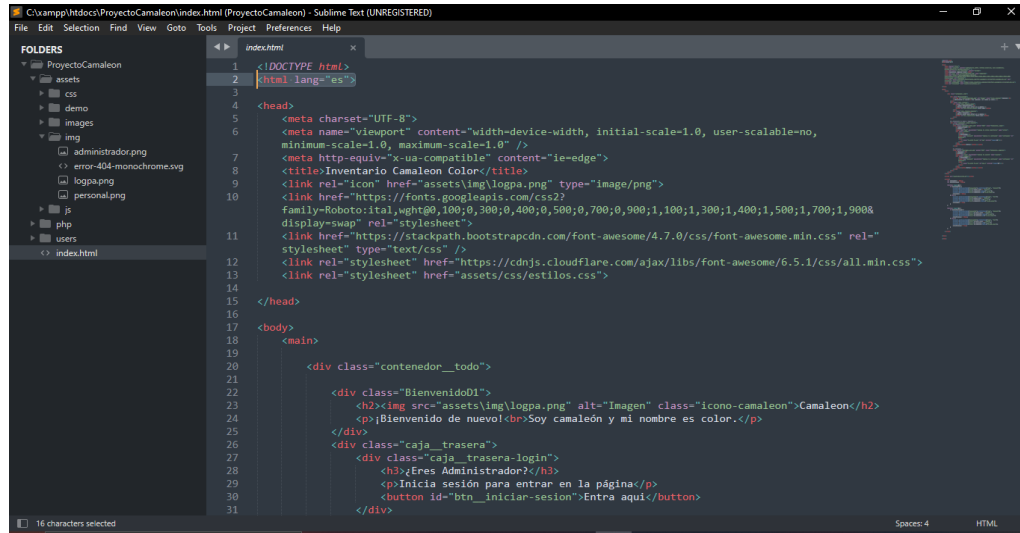


Figura. 92 Cuerpo de la carpeta img. Fuente. Elaboración propia.

Por consiguiente, dentro de la subcarpeta “css” se encuentra el archivo “estilos.css” el cual almacena la configuración de los estilos de los bodys, cajas, input, etc. del logeo. Dentro del archivo se definen colores, fuentes de letras, tamaño y posición de objetos, así como se muestra en a figura 93.

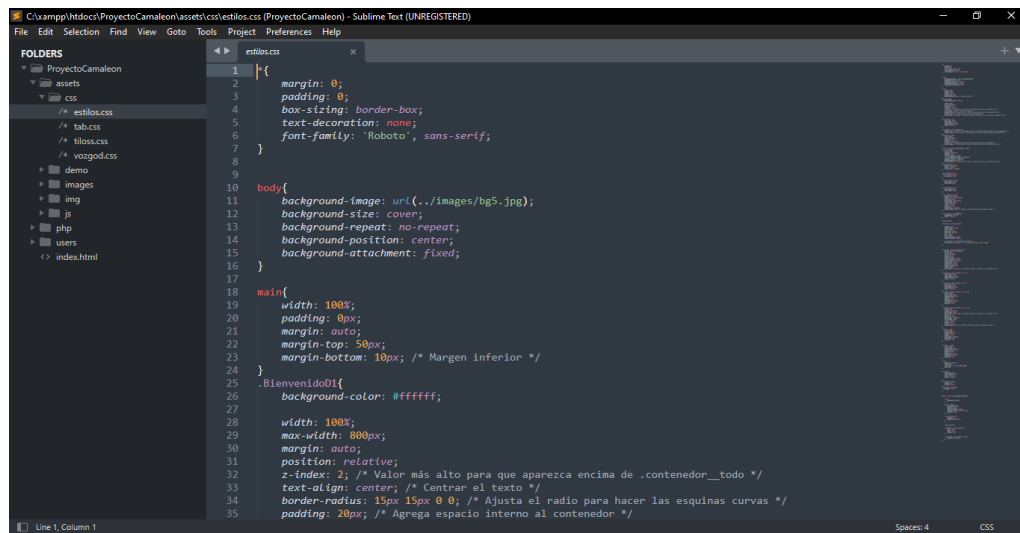


Figura. 93 Configuración del archivo estilos.css. Fuente. Elaboración propia.

Una vez definido el encabezado de la página mediante la etiqueta head, así como se muestra en la figura 94, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web, posteriormente se coloca un elemento que ayudará a representar el contenido principal de la página web que es el “body”.

Se abre un nuevo contenedor “div” con la clase de “contenedor_todo” utilizado para definirlo en el css y darle un formato deseado. Se abre otro contenedor dentro del anterior con las mismas especificaciones mencionadas, se coloca otra etiqueta que contendrá un archivo en imagen específico y el subtítulo de “Camaleón”.

Se cierra uno de los contenedores abiertos con anterioridad, permitiendo la apertura de dos nuevos los cuales contienen un encabezado y línea de texto específica de instrucciones para el usuario, a su vez se coloca un botón asignándole un id de reconocimiento “btn_iniciar_sesion”, se cierra el contenedor permitiendo la apertura de uno nuevo con los mismos elementos anteriores y asignado el id “btn_registrar” finalizando con el cierre de los contenedores anteriormente abiertos.

The image shows a code editor window with a file named 'index.html'. The code is written in HTML and defines the structure of a web page. It starts with a <body> tag, followed by a <main> tag. Inside the main tag, there is a <div class="contenedor_todo"> which contains two nested <div> elements. The first nested <div class="Bienvenido01"> contains an <h2> tag with an image source, a <h3> tag with the text 'Camaleon', and a <p> tag with a welcome message. The second nested <div class="caja_trasera"> contains two sub-sections: 'caja_trasera-login' and 'caja_trasera-register'. The login section has an <h3> tag asking if the user is an administrator, a <p> tag with login instructions, and a <button> tag with the id 'btn_iniciar-sesion'. The register section has an <h3> tag asking if the user is personal, a <p> tag with registration instructions, and a <button> tag with the id 'btn_registrarse'. The code is line-numbered from 17 to 38.

Figura. 94 Creación de contenedores de autenticación. Fuente. Elaboración propia.

Como se muestra en la figura 95, Se abre un nuevo contenedor el cual almacenara la información y objetos del primero formulario el cual dará acceso a los administradores y redirigirá al archivo “login_admin.php”, se colocan títulos y textos que indican el tipo de información a colocar en este caso: correo y contraseña con sus respectivos input de llenado, a cada uno de estos se le asigna un ID para poder ser identificados para los procesos de comparación, se crea una nueva etiqueta “span” donde se coloca un elemento imagen que ayudara a visualizar los datos del input, por lo que se le da un identificador y se le provee de una función onclick.

Finalizando con la inserción de un botón que dará proceso a la acción de redirección de archivo que se definió al inicio del formulario y cerrando la creación de la etiqueta formulario.



```

38
39
40 <!--Formulario de login y registro-->
41 <div class="contenedor_login-logout">
42 <!--Administrador-->
43 <form action="php/login_admin.php" method="POST" class="formulario_login">
44 <h2>Administrador</h2>
45 <p>CORREO:</p>
46 <input type="text" placeholder="ingresa tu correo electronico" name="correo">
47 <div class="wrapper">
48 <h1></h1>
49 <p>CONTRASEÑA:</p>
50 <input type="password" placeholder="ingresa tu contraseña" name="contrasena" id="
51 password">
52 <span>
53 <i class="fa-solid fa-eye" id="eye" onclick="toggleA()"></i>
54 </span>
55 </div>
56 <center><button>Entrar</button></center>
57 </form>
58
59

```

Figura. 95 Creación de formulario de autenticación (administradores). Fuente. Elaboración propia.

Ahora se presenta el apartado de un nuevo formulario para el usuario operativo que al igual que el anterior contendrá instrucciones e inputs específicos, aquí ya no se evalúa el campo correo si no que cambia por el campo “usuario” atributo característico del personal. La estructura del formulario es idéntica a la anterior contando con los elementos mencionados, variando solamente en el evento onclick definiendo un nombre diferente al anterior, se cierra el apartado de formulario y se da fin a la etiqueta de main, así como se muestra en figura 86.



```

57
58 <!--Personal-->
59 <form action="php/login_user.php" method="POST" class="formulario_register">
60 <h2>Personal</h2>
61 <p>USUARIO:</p>
62 <input type="text" placeholder="ingresa tu usuario" name="usuario">
63 <div class="wrapper-2">
64 <p>CONTRASEÑA:</p>
65 <input type="password" placeholder="ingresa tu contraseña" name="contrasena" id="
66 passwordp">
67 <span>
68 <i class="fa-solid fa-eye" id="eye-2" onclick="toggleP()"></i>
69 </span>
70 </div>
71 <center><button>Entrar</button></center>
72 </form>
73 </div>
74 </main>
75

```

Figura. 96 Creación de formulario de autenticación para personal. Fuente. Elaboración propia.

Posteriormente antes del cierre de las etiquetas del body y html, se colocan 2 scripts, el primero manda a llamar a un archivo JavaScript con el nombre de “script.js”. El cual permite hacer uso de las codificaciones que estén dentro del archivo, así como se muestra en la figura 97.

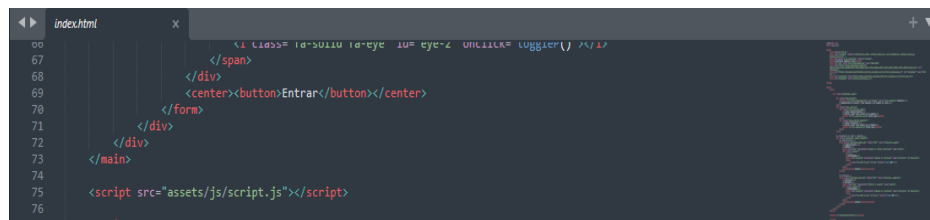


Figura. 97 Ingreso de animaciones mediante el archivo script.js. Fuente. Elaboración propia.

El script se ubica dentro de la carpeta de assets y la subcarpeta “js”, en el cual almacena las configuraciones de animación, así como se observa en la figura 98, para los cambios entre el formulario del administrador y del personal, mediante el llamado de identificadores asignados en el archivo “index.html” de cada formulario. Se definen variables para dar las funcionalidades de animación a cada caja de formulario.

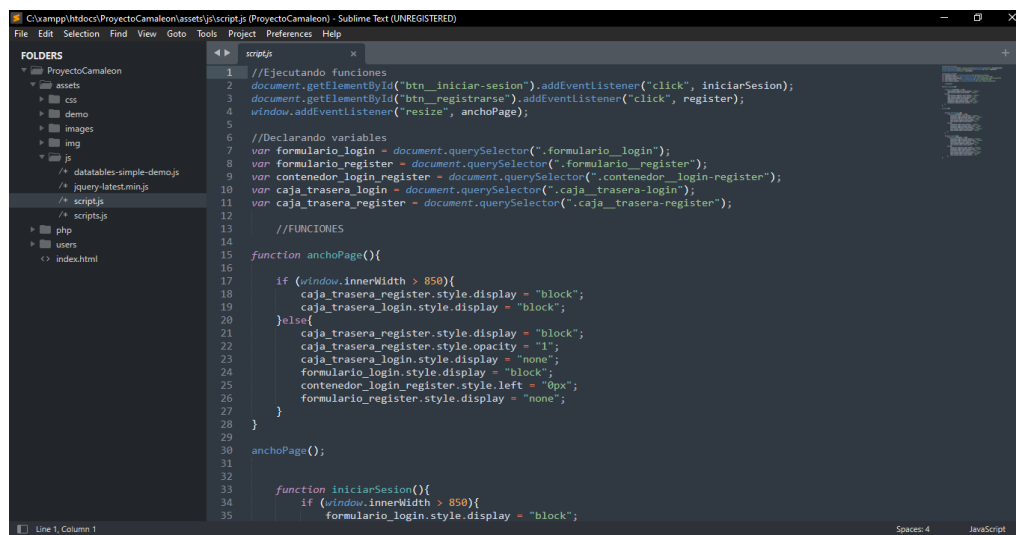
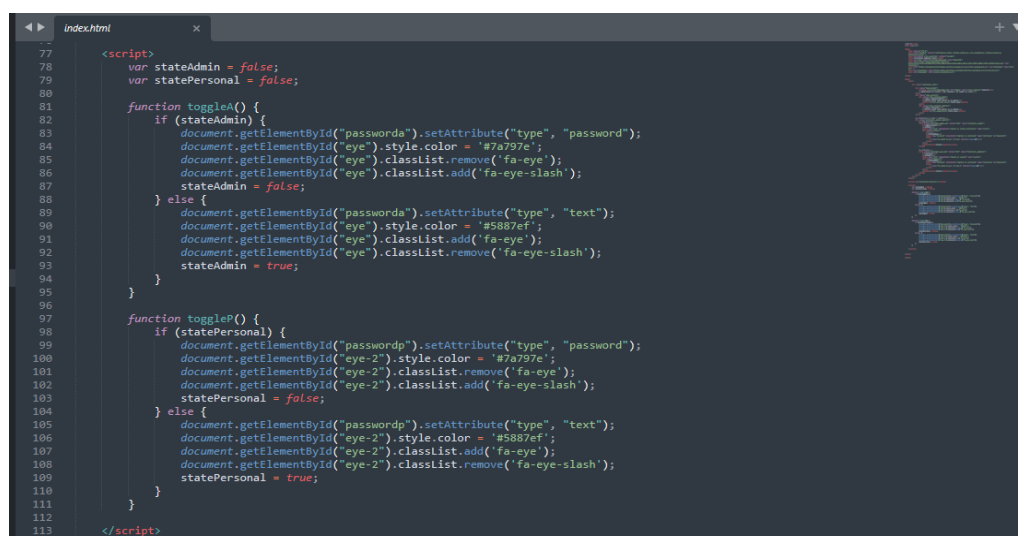


Figura. 98 Configuración de las animaciones para los formularios de administrador y personal. Fuente. Elaboración propia.

Continuado con el archivo `index.html`, como segundo script se encuentra una serie de instrucciones las cuales no hacen llamado a ningún archivo ya que las líneas de codificación están definidas en el mismo archivo, se declaran dos variables una para el formulario administrador y la otra para el usuario operativo, ambas variables son declaradas en falso. Posteriormente se coloca una función `toggle` para poder cambiar el estado de visibilidad del elemento de la contraseña, también se coloca el tipo de conversión de la contraseña y se define el color del estado de la imagen con el evento `toggle`, esto para el caso en que la variable este como falsa en caso de que la variable este en verdadero se coloca los parámetros de diseño de la imagen y se hace la conversión de password a texto. Lo mismo aplica para la función del personal, así como se muestra en la figura 99.



```
77 </script>
78 var stateAdmin = false;
79 var statePersonal = false;
80
81 function toggleA() {
82   if (stateAdmin) {
83     document.getElementById("passworda").setAttribute("type", "password");
84     document.getElementById("eye").style.color = '#7a797e';
85     document.getElementById("eye").classList.remove('fa-eye');
86     document.getElementById("eye").classList.add('fa-eye-slash');
87     stateAdmin = false;
88   } else {
89     document.getElementById("passworda").setAttribute("type", "text");
90     document.getElementById("eye").style.color = '#5887ef';
91     document.getElementById("eye").classList.add('fa-eye');
92     document.getElementById("eye").classList.remove('fa-eye-slash');
93     stateAdmin = true;
94   }
95 }
96
97 function toggleP() {
98   if (statePersonal) {
99     document.getElementById("passwordp").setAttribute("type", "password");
100    document.getElementById("eye-2").style.color = '#7a797e';
101    document.getElementById("eye-2").classList.remove('fa-eye');
102    document.getElementById("eye-2").classList.add('fa-eye-slash');
103    statePersonal = false;
104  } else {
105    document.getElementById("passwordp").setAttribute("type", "text");
106    document.getElementById("eye-2").style.color = '#5887ef';
107    document.getElementById("eye-2").classList.add('fa-eye');
108    document.getElementById("eye-2").classList.remove('fa-eye-slash');
109    statePersonal = true;
110  }
111 }
112
113 </script>
```

Figura. 99 Configuración para la visualización de la contraseña en formularios. Fuente. Elaboración propia.

Por otra parte, para que el login pueda comparar lo que se haya ingresado dentro de los inputs se encuentra el archivo `login_admin.php` donde se incluyen los archivos de conexión `“FirebaseRDB”` y `“Conexión_be”`, así como se muestra en la figura 100, para hacer uso y conexión a la base de datos. Se crean variables en las que se almacenaran los datos que se han ingresado en los inputs del archivo `“index.html”`

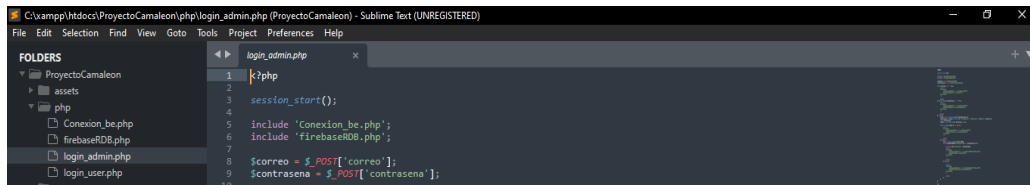


Figura. 100 Ingreso de archivos de conexión a Firebase del formulario administrador. Fuente. Elaboración propia.

Posteriormente se abren sentencias de condición las cuales evaluarán si los campos contienen información, así como se muestra en la figura 101, en caso de no contener se coloca un mensaje de alerta y se redirige al archivo de “index.html”.



Figura. 101 Evaluación de contenido de campos del formulario administrador. Fuente. Elaboración propia.

En caso de que todas las variables contengan información se colocan la declaración de 3 variables las cuales dispondrán el uso de la base de datos así como la comparación de información si es que lo que está dentro de la variable es idéntico a lo que contiene la tabla de la base de datos, así como se muestra en la figura 102, por lo que se coloca una sentencia de condición en caso de que la información coincida se coloca un mensaje de alerta exitoso y da direccionamiento a “home.php”, en caso de no serlo dará un mensaje de alerta erróneo y devolverá a “index.html”.

```

login_admin.php (ProyectoCamaleon) - Sublime Text (UNREGISTERED)
ols Project Preferences Help

login_admin.php
29 } else {
30     $fdb = new firebaseDB( $databaseurl );
31     $retrieve = $fdb->retrieve( "/usuarios", "Correo_a", "EQUAL", $correo );
32     echo $retrieve;
33     $data = json_decode( $retrieve, 1 );
34
35     if ( count( $data ) == 0 ) {
36         echo '
37         <script>
38             window.location = "../index.html";
39             alert("Correo no registrado");
40         </script>
41         ';
42         exit;
43     } else {
44         $id = array_keys( $data )[0];
45         if ( $data[$id]['Contraseña'] == $contrasena ) {
46             $_SESSION['usuario'] = $data[$id];
47             echo '
48             <script>
49                 window.location = "../users/admin/home.php";
50                 alert("Logueo exitoso");
51             </script>
52             ';
53             exit;
54         } else {
55             echo '
56             <script>
57                 window.location = "../users/admin/home.php";
58                 alert("Contraseña Incorrecta");
59             </script>
60             ';
61             exit;
62         }
63     }
64 }
65 }
66 }
67 }
68 }
69 }
70 }
    
```

Figura. 102 Configuración de ingreso de datos a la tabla usuarios. Fuente. Elaboración propia.

Para el caso de comparación del formulario del personal, las sentencias y condiciones son idénticas como lo muestran las figuras 103 y 104, exceptuando que al ser correctos los datos comparados se redirigirá al archivo “home.php” ubicado en la carpeta de “usuario”.

```

C:\xampp\htdocs\ProyectoCamaleon\php\login_user.php (ProyectoCamaleon) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

login_user.php
1 <?php
2 session_start();
3 include 'Conexion_be.php';
4 include 'firebaseDB.php';
5
6 $usuario = $_POST['usuario'];
7 $contrasena = $_POST['contrasena'];
8
9 if ( $usuario == "" ) {
10     echo '
11     <script>
12         window.location = "../index.html";
13         alert("Ingrese el usuario");
14     </script>
15     ';
16     exit;
17 } else if ( $contrasena == "" ) {
18     echo '
19     <script>
20         window.location = "../index.html";
21         alert("Ingrese la contraseña");
22     </script>
23     ';
24     exit;
25 }
26 }
27 }
28 }
    
```

Figura. 103 Ingreso de archivos de conexión a Firebase del formulario personal. Fuente. Elaboración propia.

```

login_user.php
1  <?php
2  $firebase = new Firebase($databaseUrl);
3  $retrieve = $firebase->retrieve("/personal", "Usuario_p", "EQUAL", $usuario);
4  echo $retrieve;
5  $data = json_decode($retrieve, 1);
6
7  if (count($data) == 0) {
8      echo '
9      <script>
10         window.location = "../index.html";
11         alert("Usuario no registrado");
12     </script>
13     '
14     ;
15     exit;
16 } else {
17     $id = array_keys($data)[0];
18     if ($data[$id]['Contraseña'] == $contrasena) {
19         $SESSION['usuario'] = $data[$id];
20         echo '
21         <script>
22             window.location = "../users/usuario/home.php";
23             alert("Logueo exitoso");
24         </script>
25         '
26         ;
27         exit;
28     } else {
29         echo '
30         <script>
31             alert("Intente loguearse nuevamente");
32         </script>
33         '
34         ;
35         exit;
36     }
37 }
38 }
39 }
40 }
41 }
42 }
43 }
44 }
45 }
46 }
47 }
48 }
49 }
50 }
51 }
52 }
53 }
54 }
55 }
56 }
57 }
58 }
59 }
60 }
61 }
62 }
63 }
64 }
65 }
66 }
67 }
68 }
69 }
70 }

```

Figura. 104 Configuración de ingreso de datos a la tabla personal. Fuente. Elaboración propia.

Carpeta Administrador Home:

En la carpeta de user/administrador se encuentra el archivo “home.php” el cual sería visualizado de manera inmediata una vez que la autenticación se haya concretado correctamente, dentro del archivo se especifica en lenguaje de php mediante sentencias de decisión donde se evalúa si una vez en la página de home.php se autentico de manera correcta, en caso de no ser así se coloca una alerta y retornara a index.html. A si mismo evalúa el tiempo de inactividad entre navegación de archivos teniendo un conteo de 300 segundos de inactividad. Si el conteo es completado se retornará a index.html de manera forzada, en caso de navegar entre archivos el conteo se reiniciará de manera automática, Culminando con el cierre del lenguaje de php, así como se muestra en la figura 105.

```

home.php
1  <?php
2  session_start();
3  if(isset($_SESSION['usuario'])){// Esta logeado?
4      echo '
5      <script>
6          alert("Inicio session");
7          window.location = "../index.html";
8      </script>
9      '
10     ;
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad = 300; //Expresion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if (($actual-$tiempo) >= $inactividad){
28     echo '
29     <script>
30         alert("Tiempo excedido");
31         window.location = "../index.html";
32     </script>
33     '
34     ;
35     session_destroy();
36     die();
37 }
38 // En caso que este sea mayor del tiempo seteado lo desloga.
39 $SESSION['time'] = $actual;
40 }
41 }
42 }
43 }
44 }
45 }
46 }
47 }
48 }
49 }
50 }
51 }
52 }
53 }
54 }
55 }
56 }
57 }
58 }
59 }
60 }
61 }
62 }
63 }
64 }
65 }
66 }
67 }
68 }
69 }
70 }

```

Figura. 105 Configuración de lenguaje y tiempo de inactividad del archivo home.php. Fuente. Elaboración propia.


```

home.php
42
43 <!DOCTYPE html>
44 <html lang="es">
45
46 <head>
47   <meta charset="UTF-8">
48   <meta name="viewport" content="width=device-width, initial-scale=1">
49   <title>Inicio</title>
50   <link rel="icon" href="assets/img/logopa.png" type="image/png">
51   <link rel="stylesheet" href="assets/css/normalize.css">
52   <link rel="stylesheet" href="assets/css/sweetalert2.css">
53   <link rel="stylesheet" href="assets/css/material.min.css">
54   <link rel="stylesheet" href="assets/css/material-design-iconic-font.min.css">
55   <link rel="stylesheet" href="assets/css/jquery.mCustomScrollbar.css">
56   <link rel="stylesheet" href="assets/css/main.css">
57   <script src="//ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
58   <script>
59     window.jQuery || document.write('<script src="assets/js/jquery-1.11.2.min.js"></script>')
60   </script>
61   <script src="assets/js/material.min.js"></script>
62   <script src="assets/js/sweetalert2.min.js"></script>
63   <script src="assets/js/jquery.mCustomScrollbar.concat.min.js"></script>
64   <script src="assets/js/marco.js"></script>
65
66 </head>
67

```

Figura. 106 Estructura de lenguaje HTML, enlaces e Idioma del archivo home.php. Fuente. Elaboración propia.

Posteriormente se coloca la estructura de `<!DOCTYPE html>` la cual informará al navegador el tipo de versión HTML que fue implementado en el archivo, a su vez se indica con una etiqueta de tipo `lang` el tipo de idioma con el que será escrita la página web en este caso se hace uso del idioma español. A su vez se muestra la apertura de un `head` en el cual se proveerá la información como lo es el título y los enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assets” y de las subcarpetas de “img”, “css” y “js”, así como se muestra en la figura 106.

Como se muestra en la figura 107, una vez definido el encabezado de la página mediante la etiqueta `head`, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web. Posteriormente se coloca un elemento que ayudara a representar el contenido principal de la página web que es el “body”.

```

home.php
67
68 <body>
69   <!-- barra de navegacion lateral -->
70   <section class="full-width navLateral">
71     <div class="full-width navLateral-bg btn-menu"></div>
72     <div class="full-width navLateral-body">
73       <div class="full-width navLateral-body-logo text-center tittles">
74         <i class="zmdi zmdi-close btn-menu"></i> CC Camaleon Color
75       </div>
76       <figure class="full-width navLateral-body-tittle-menu">
77         <div>
78           
79         </div>
80         <figcaption>
81           <span><?php echo $_SESSION['usuario']['NombreCompleto'];></span>&nbsp;<span><?php echo $_SESSION['
82             usuario']['Apellido'];></span></span>
83           <br>
84           <span><small><?php echo $_SESSION['usuario']['Usuario_a'];></small></span>
85         </figcaption>
86       </figure>
87

```

Figura. 107 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.

Una vez definido el encabezado de la página mediante la etiqueta head, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web. Posteriormente se coloca un elemento que ayudara a representar el contenido principal de la página web que es el “body”.

Posteriormente se abre una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, se colocan contenedores para aplicar estilos correspondientes a las sentencias, se agregan imágenes que harán la función del avatar del administrador, así como el nombre del usuario administrador que se autentico con anterioridad los datos son llamados desde la base de datos de firebase.

Posteriormente se coloca una etiqueta nav para proporcionar una navegación estructurada en la página, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta assets/css, así como se muestra en las figuras 108, 109, 110 y 111.

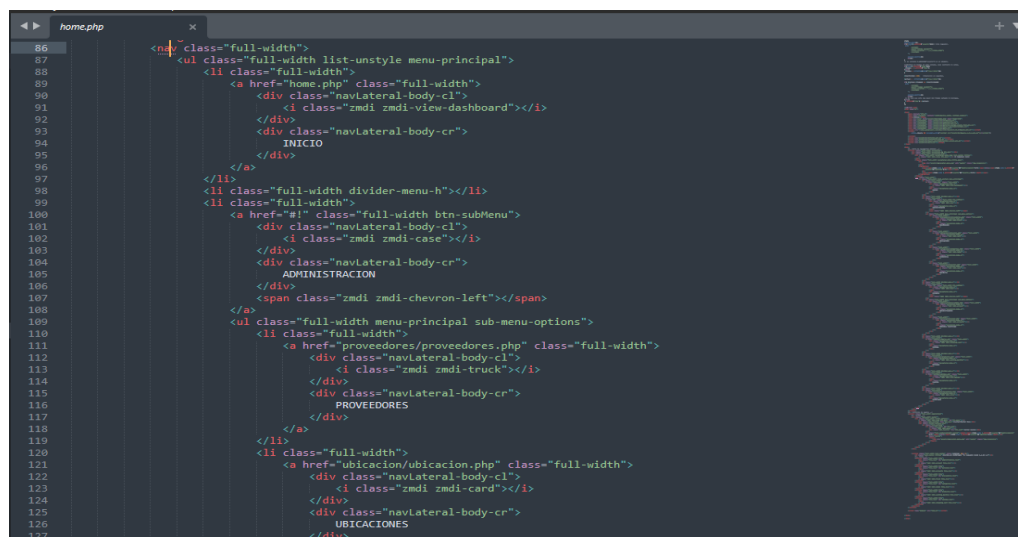
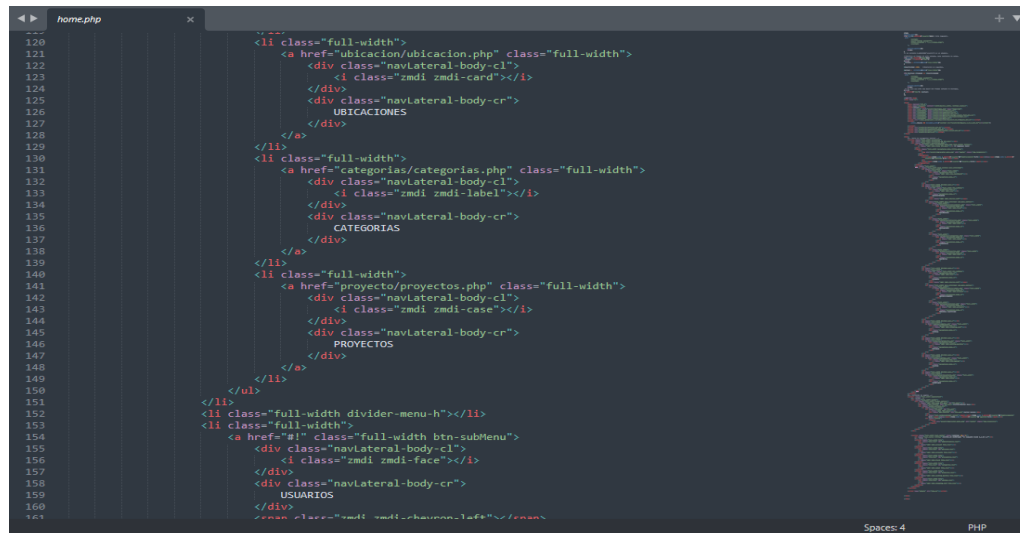
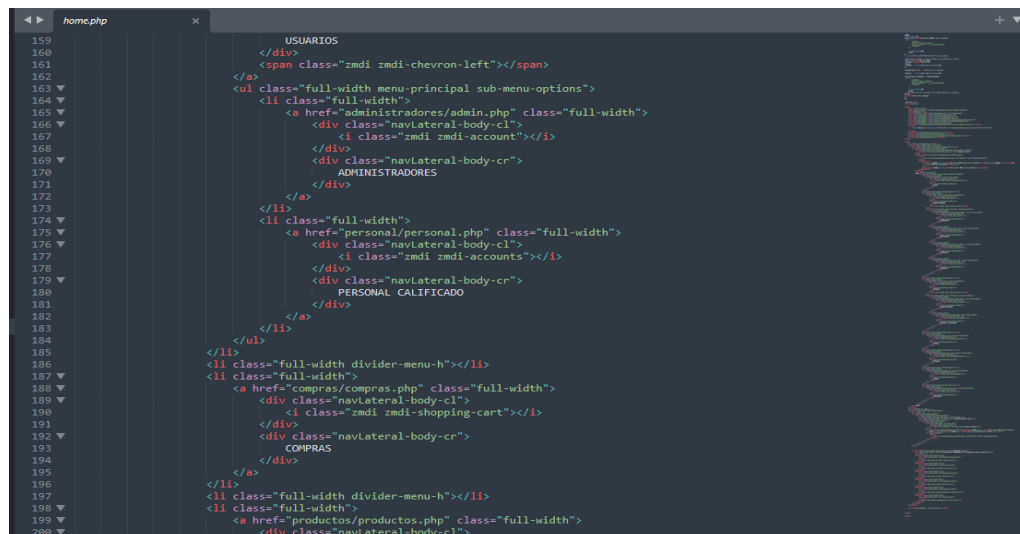


Figura. 108 Codificación de la barra de navegación lateral pt1. Fuente. Elaboración propia.



```
120 </li>
121 <li class="full-width">
122   <a href="ubicacion/ubicacion.php" class="full-width">
123     <div class="navLateral-body-cl">
124       <i class="zmdi zmdi-card"></i>
125     </div>
126     <div class="navLateral-body-cr">
127       UBICACIONES
128     </div>
129   </li>
130 <li class="full-width">
131   <a href="categorias/categorias.php" class="full-width">
132     <div class="navLateral-body-cl">
133       <i class="zmdi zmdi-label"></i>
134     </div>
135     <div class="navLateral-body-cr">
136       CATEGORIAS
137     </div>
138   </li>
139 <li class="full-width">
140   <a href="proyecto/proyectos.php" class="full-width">
141     <div class="navLateral-body-cl">
142       <i class="zmdi zmdi-cake"></i>
143     </div>
144     <div class="navLateral-body-cr">
145       PROYECTOS
146     </div>
147   </li>
148 </ul>
149
150 </li>
151 <li class="full-width divider-menu-h"></li>
152 <li class="full-width">
153   <a href="#" class="full-width btn-subMenu">
154     <div class="navLateral-body-cl">
155       <i class="zmdi zmdi-face"></i>
156     </div>
157     <div class="navLateral-body-cr">
158       USUARIOS
159     </div>
160   </li>
161 </li>
```

Figura. 109 Codificación de la barra de navegación lateral pt2. Fuente. Elaboración propia.



```
159 </div>
160 <span class="zmdi zmdi-chevron-left"></span>
161 </a>
162 </li>
163 <ul class="full-width menu-principal sub-menu-options">
164   <li class="full-width">
165     <a href="administradores/admin.php" class="full-width">
166       <div class="navLateral-body-cl">
167         <i class="zmdi zmdi-account"></i>
168       </div>
169       <div class="navLateral-body-cr">
170         ADMINISTRADORES
171       </div>
172     </li>
173   <li class="full-width">
174     <a href="personal/personal.php" class="full-width">
175       <div class="navLateral-body-cl">
176         <i class="zmdi zmdi-accounts"></i>
177       </div>
178       <div class="navLateral-body-cr">
179         PERSONAL CALIFICADO
180       </div>
181     </li>
182   </ul>
183 </li>
184 <li class="full-width divider-menu-h"></li>
185 <li class="full-width">
186   <a href="compras/compras.php" class="full-width">
187     <div class="navLateral-body-cl">
188       <i class="zmdi zmdi-shopping-cart"></i>
189     </div>
190     <div class="navLateral-body-cr">
191       COMPRAS
192     </div>
193   </li>
194 <li class="full-width divider-menu-h"></li>
195 <li class="full-width">
196   <a href="productos/productos.php" class="full-width">
197     <div class="navLateral-body-cl">
```

Figura. 110 Codificación de la barra de navegación lateral pt3. Fuente. Elaboración propia.

```

207 </li>
208 <li class="full-width divider-menu-h"></li>
209 <li class="full-width">
210 <a href="salidas/salidas.php" class="full-width">
211 <div class="navLateral-body-cl">
212 <i class="zmdi zmdi-view-agenda"></i>
213 </div>
214 <div class="navLateral-body-cr">
215 SALIDAS
216 </div>
217 </a>
218 </li>
219 <li class="full-width divider-menu-h"></li>
220 <li class="full-width">
221 <a href="inventario/inventario.php" class="full-width">
222 <div class="navLateral-body-cl">
223 <i class="zmdi zmdi-store"></i>
224 </div>
225 <div class="navLateral-body-cr">
226 INVENTARIO
227 </div>
228 </a>
229 </li>
230 </ul>
231 </nav>
232 </div>
233 </section>
    
```

Figura. 111 Codificación de la barra de navegación lateral pt4. Fuente. Elaboración propia.

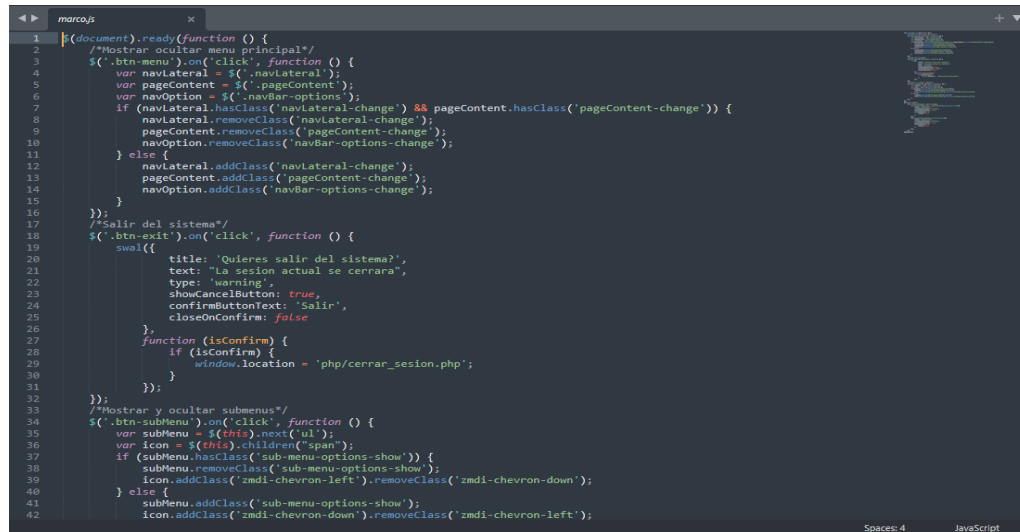
Por parte del contenido de la página web como se muestra en la figura 112, se coloca una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, la cual contendrá contenedores para aplicar estilos con css, además de continuar con las animaciones de la barra de navegación, se presenta la configuración para ocultar y mostrar la barra de navegación. Se programa la interfaz de cierre de sesión mediante la implementación de un botón con identificador. Posteriormente se coloca una imagen con el avatar de identificación del administrador y se manda a llamar los datos propios de este, desde la base de datos Firebase.

```

234 <!-- contenido de pagina -->
235 <section class="full-width pageContent">
236 <!-- navBar -->
237 <div class="full-width navBar">
238 <div class="full-width navBar-options">
239 <i class="zmdi zmdi-swap btn-menu" id="btn-menu"></i>
240 <div class="mdl-tooltip" for="btn-menu">Ocultar/Mostrar Menu</div>
241 <nav class="navBar-options-list">
242 <ul class="list-unstyled">
243 <li class="btn-exit" id="btn-exit">
244 <i class="zmdi zmdi-power"></i>
245 <div class="mdl-tooltip" for="btn-exit">Cerrar Sesión</div>
246 </li>
247 <li class="text-condensedLight noLink"><small><small><?php echo $_SESSION['usuario']['NombreCompleto']></small></small></li>
248 <li class="noLink">
249 <figure>
250 
251 </figure>
252 </li>
253 </ul>
254 </nav>
255 </div>
256 </div>
257
    
```

Figura. 112 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Las animaciones correspondientes al cierre de sesión y ocultar/mostrar la barra de navegación son estructuradas en el archivo “marco.js” ubicado en la carpeta assets/js, así como se muestra en la figura 113.



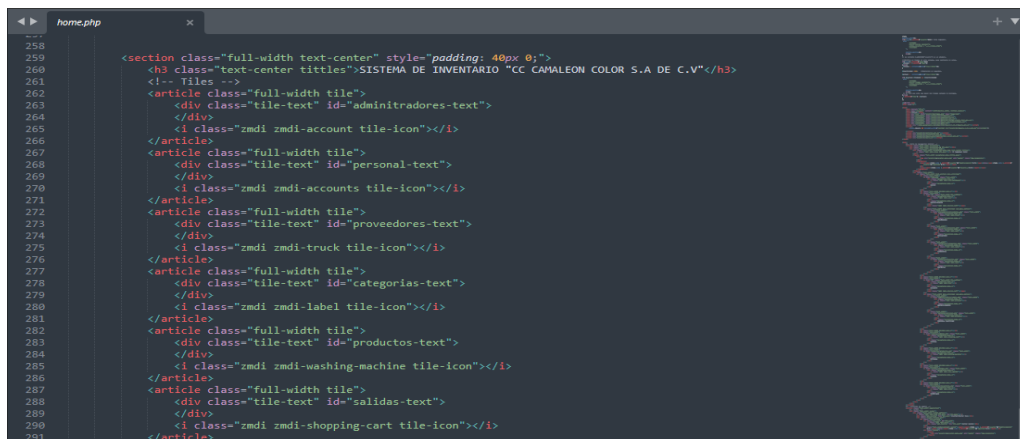
```

1  (document).ready(function () {
2      //Mostrar ocultar menu principal*/
3      $('.btn-menu').on('click', function () {
4          var navLateral = $('.navLateral');
5          var pageContent = $('.pageContent');
6          var navOption = $('.navBar-options');
7          if (navLateral.hasClass('navLateral-change') && pageContent.hasClass('pageContent-change')) {
8              navLateral.removeClass('navLateral-change');
9              pageContent.removeClass('pageContent-change');
10             navOption.removeClass('navBar-options-change');
11         } else {
12             navLateral.addClass('navLateral-change');
13             pageContent.addClass('pageContent-change');
14             navOption.addClass('navBar-options-change');
15         }
16     });
17     //Salir del sistema*/
18     $('.btn-exit').on('click', function () {
19         swal({
20             title: 'Quieres salir del sistema?',
21             text: 'La sesion actual se cerrara',
22             type: 'warning',
23             showCancelButton: true,
24             confirmButtonText: 'Salir',
25             closeOnConfirm: false
26         },
27         function (isConfirm) {
28             if (isConfirm) {
29                 window.location = 'php/cerrar_sesion.php';
30             }
31         });
32     });
33     //Mostrar y ocultar submenus*/
34     $('.btn-subMenu').on('click', function () {
35         var subMenu = $(this).next('ul');
36         var icon = $(this).children('span');
37         if (subMenu.hasClass('sub-menu-options-show')) {
38             subMenu.removeClass('sub-menu-options-show');
39             icon.addClass('zmdi-chevron-left').removeClass('zmdi-chevron-down');
40         } else {
41             subMenu.addClass('sub-menu-options-show');
42             icon.addClass('zmdi-chevron-down').removeClass('zmdi-chevron-left');
43         }
44     });
45 });

```

Figura. 113 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Dentro de una nueva etiqueta tipo section se da finalización a la configuración de la página home.php en la cual se colocan contenedores ubicándose dentro de estos el título del sistema web, así como azulejos mediante las etiquetas de artículos los cuales contienen imágenes de acuerdo a la especificación de la tabla de registro representada, a su vez mediante el archivo “home.js”, como se muestra en la figura 114, se manda a llamar desde la base de datos la cantidad de registros que contiene la tabla y son mostrados de forma numérica dentro de la estructura del artículo.



```

258 <section class="full-width text-center" style="padding: 40px 0;">
259 <h3 class="text-center titles">SISTEMA DE INVENTARIO "CC CAPALEON COLOR S.A DE C.V"</h3>
260 <div class="row">
261 <div class="full-width tile">
262 <div class="tile-text" id="administradores-text">
263 <div>
264 <i class="zmdi zmdi-account tile-icon"></i>
265 </div>
266 </div>
267 <div class="full-width tile">
268 <div class="tile-text" id="personal-text">
269 <div>
270 <i class="zmdi zmdi-accounts tile-icon"></i>
271 </div>
272 </div>
273 <div class="full-width tile">
274 <div class="tile-text" id="proveedores-text">
275 <div>
276 <i class="zmdi zmdi-truck tile-icon"></i>
277 </div>
278 </div>
279 <div class="full-width tile">
280 <div class="tile-text" id="categorias-text">
281 <div>
282 <i class="zmdi zmdi-label tile-icon"></i>
283 </div>
284 </div>
285 <div class="full-width tile">
286 <div class="tile-text" id="productos-text">
287 <div>
288 <i class="zmdi zmdi-washing-machine tile-icon"></i>
289 </div>
290 </div>
291 <div class="full-width tile">
292 <div class="tile-text" id="salidas-text">
293 <div>
294 <i class="zmdi zmdi-shopping-cart tile-icon"></i>
295 </div>
296 </div>
297 </div>
298 </div>
299 </div>
300 </div>
301 </div>

```

Figura. 114 Creación de azulejos de visualización de registros. Fuente. Elaboración propia.

En el archivo “home.js” se hacen las conexiones hacia la base de datos, manejando los registros con un conteo para cada tabla identificada por los azulejos correspondientes, en caso de ocurrir algún error de red o sintaxis de información se almacenará en un catch arrojando un mensaje erróneo de alerta, así como se muestra en la figura 115.

```

1  import {
2    initializeApp
3  } from "https://www.gstatic.com/firebasejs/10.6.0/firebase-app.js";
4  import {
5    getDatabase,
6    ref,
7    get
8  } from "https://www.gstatic.com/firebasejs/10.6.0/firebase-database.js";
9
10 const firebaseConfig = {
11   apiKey: "AIzaSyOPzZKE74dX0K_SuVnognTGMir4udm8Y0A",
12   authDomain: "ccp1-6456d.firebaseio.com",
13   databaseURL: "https://ccp1-6456d-default-rtdb.firebaseio.com",
14   projectId: "ccp1-6456d",
15   storageBucket: "ccp1-6456d.appspot.com",
16   messagingSenderId: "266674927659",
17   appId: "1:266674927659:web:3fbb25dcae1b50b138126c"
18 };
19
20 const app = initializeApp(firebaseConfig);
21 const database = getDatabase(app);
22
23 const adminCount = document.getElementById("administradores-text");
24 const personalCount = document.getElementById("personal-text");
25 const proveedorCount = document.getElementById("proveedores-text");
26 const categoriaCount = document.getElementById("categorias-text");
27 const productoCount = document.getElementById("productos-text");
28 const salidaCount = document.getElementById("salidas-text");
29
30 function CargarDatos() {
31
32   const usuariosRef = ref(database, "usuarios");
33   const personalRef = ref(database, "personal");
34   const proveedorRef = ref(database, "proveedores");
35   const categoriaRef = ref(database, "categorias");
36   const productoRef = ref(database, "productos");
37   const salidaRef = ref(database, "salidas");
38
39   get(usuariosRef).then(snapshot => {
40     const usuarios = snapshot.val();
41     let Count = 0;
42

```

Figura. 115 Codificación del conteo de registros para el uso de azulejos, Fuente. Elaboración propia.

3.10.3. Administradores:

Por parte del archivo “admin.php” se especifica en lenguaje de php mediante sentencias de decisión donde se evalúa si una vez en la página de home.php se autentico de manera correcta, en caso de no ser así se coloca una alerta y retornara a index.html. A si mismo evalúa el tiempo de inactividad entre navegación de archivos teniendo un conteo de 300 segundos de inactividad. Si el conteo es completado se retornará a index.html de manera forzada, en caso de navegar entre archivos el conteo se reiniciará de manera automática, Culminando con el cierre del lenguaje de php, así como se muestra en la figura 116.

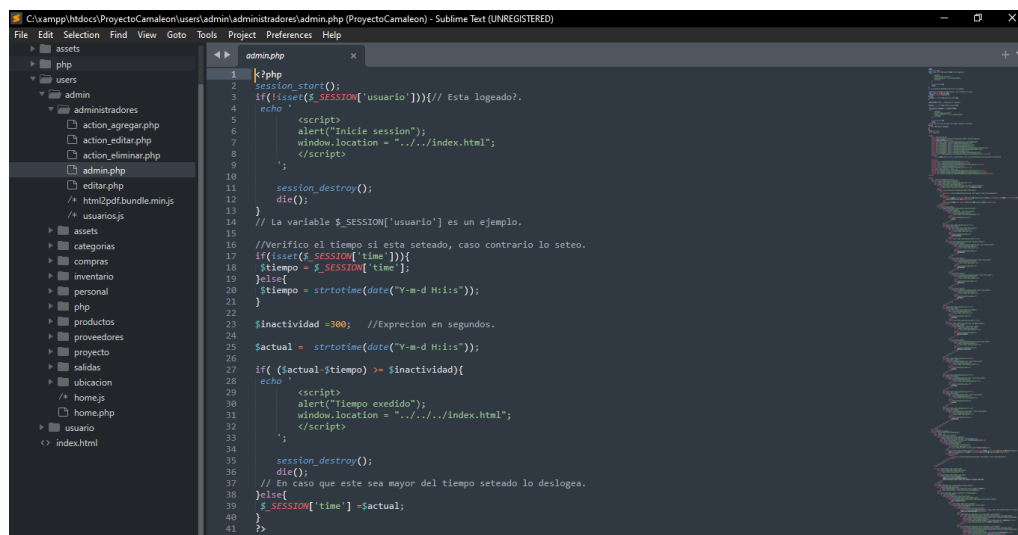


Figura. 116 Configuración de lenguaje y tiempo de inactividad del archivo admin.php. Fuente. Elaboración propia.

Posteriormente se coloca la estructura de! DOCTYPE html> la cual informara al navegador el tipo de versión HTML que fue implementado en el archivo, así como se muestra en la figura 117, a su vez se indica con una etiqueta de tipo lang el tipo de idioma con el que será escrita la página web en este caso se hace uso del idioma español. A su vez se muestra la apertura de un head en el cual se proveerá la información como lo es el título y lo enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assets” y de las subcarpetas de “img”, “css” y “js”.

```

41  ?>
42  <!DOCTYPE html>
43  <html lang="es">
44
45  <head>
46    <meta charset="UTF-8">
47    <meta name="viewport" content="width=device-width, initial-scale=1">
48    <title>Administradores</title>
49    <link rel="icon" href="../../assets/img/logopa.png" type="image/png">
50    <link rel="stylesheet" href="../../assets/css/normalize.css">
51    <link rel="stylesheet" href="../../assets/css/sweetalert2.css">
52    <link rel="stylesheet" href="../../assets/css/material.min.css">
53    <link rel="stylesheet" href="../../assets/css/material-design-iconic-font.min.css">
54    <link rel="stylesheet" href="../../assets/css/jquery.mCustomScrollbar.css">
55    <link rel="stylesheet" href="../../assets/css/main.css">
56    <link rel="stylesheet" href="../../assets/css/simple-datatables.min.css">
57    <script src="//ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
58    <script>
59      window.jQuery || document.write('<script src="../../assets/js/jquery-1.11.2.min.js"></script>')
60    </script>
61
62    <script src="../../assets/js/material.min.js"></script>
63    <script src="../../assets/js/sweetalert2.min.js"></script>
64    <script src="../../assets/js/jquery.mCustomScrollbar.concat.min.js"></script>
65    <script src="../../assets/js/marco2.js"></script>
66    <script src="../../assets/js/simple-datatables.js"></script>

```

Figura. 117 Ingreso de Titulo y enlaces de funciones del archivo admin.php, Fuente. Elaboración propia.

Como se muestra en la figura 118, una vez definido el encabezado de la página mediante la etiqueta head, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web. Posteriormente se coloca un elemento que ayudara a representar el contenido principal de la página web que es el “body”. Posteriormente se abre una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, se colocan contenedores para aplicar estilos correspondientes a las sentencias, se agregan imágenes que harán la función del avatar del administrador, así como el nombre del usuario administrador que se autentico con anterioridad los datos son llamados desde la base de datos de Firebase.

```

72
73  <body>
74    <!-- barra de navegacion lateral -->
75    <section class="full-width navLateral">
76      <div class="full-width navLateral-bg btn-menu"></div>
77      <div class="full-width navLateral-body">
78        <div class="full-width navLateral-body-logo text-center titles">
79          <i class="zmdi zmdi-close btn-menu"></i> CC Camaleon Color
80        </div>
81        <figure class="full-width navLateral-body-tittle-menu">
82          <div>
83            
84          </div>
85          <figcaption>
86            <span><span><?php echo $_SESSION['usuario'];></span>&nbsp;<span><?php echo $_SESSION['
87              usuario '][ 'Apellido '];></span></span>
88            <br>
89            <span><small><?php echo $_SESSION['usuario'];></small></span>
90          </figcaption>

```

Figura. 118 Codificación del nombre y avatar del usuario, Fuente. Elaboración propia.

Posteriormente se coloca una etiqueta nav para proporcionar una navegación estructurada en la página, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta assets/css, así como se muestra en las figuras 119, 120, 121 y 122.


```
admin.php
91 <nav class="full-width">
92 <ul class="full-width list-unstyled menu-principal">
93 <li class="full-width">
94 <a href=".." home.php" class="full-width">
95 <div class="navLateral-body-cl">
96 <i class="zmdi zmdi-view-dashboard"></i>
97 </div>
98 <div class="navLateral-body-cr">
99 INICIO
100 </div>
101 </a>
102 </li>
103 <li class="full-width divider-menu-h"></li>
104 <li class="full-width">
105 <a href="#1" class="full-width btn-subMenu">
106 <div class="navLateral-body-cl">
107 <i class="zmdi zmdi-case"></i>
108 </div>
109 <div class="navLateral-body-cr">
110 ADMINISTRACION
111 </div>
112 <span class="zmdi zmdi-chevron-left"></span>
113 </a>
114 <ul class="full-width menu-principal sub-menu-options">
115 <li class="full-width">
116 <a href=".." proveedores/proveedores.php" class="full-width">
117 <div class="navLateral-body-cl">
118 <i class="zmdi zmdi-truck"></i>
119 </div>
120 <div class="navLateral-body-cr">
121 PROVEEDORES
122 </div>
123 </a>
124 </li>
125 <li class="full-width">
126 <a href=".." ubicacion/ubicacion.php" class="full-width">
127 <div class="navLateral-body-cl">
128 <i class="zmdi zmdi-card"></i>
129 </div>
130 <div class="navLateral-body-cr">
131 UBICACIONES
132 </div>
133 </a>
134 </li>
135 <li class="full-width">
136 <a href=".." categorias/categorias.php" class="full-width">
137 <div class="navLateral-body-cl">
138 <i class="zmdi zmdi-label"></i>
139 </div>
140 <div class="navLateral-body-cr">
141 CATEGORIAS
142 </div>
143 </a>
144 </li>
145 <li class="full-width">
146 <a href=".." proyecto/proyectos.php" class="full-width">
147 <div class="navLateral-body-cl">
148 <i class="zmdi zmdi-case"></i>
149 </div>
150 <div class="navLateral-body-cr">
151 PROYECTOS
152 </div>
153 </a>
154 </li>
155 </ul>
156 </li>
157 <li class="full-width divider-menu-h"></li>
158 <li class="full-width">
159 <a href="#1" class="full-width btn-subMenu">
160 <div class="navLateral-body-cl">
161 <i class="zmdi zmdi-face"></i>
162 </div>
163 <div class="navLateral-body-cr">
164 USUARIOS
165 </div>
166 <span class="zmdi zmdi-chevron-left"></span>
167 </a>
168 <ul class="full-width menu-principal sub-menu-options">
169 <li class="full-width">
170 <a href="admin.php" class="full-width">
```

Figura. 119 Codificación de la barra de navegación lateral pt1, Fuente. Elaboración propia.

```
admin.php
129 </div>
130 <div class="navLateral-body-cr">
131 UBICACIONES
132 </div>
133 </a>
134 </li>
135 <li class="full-width">
136 <a href=".." categorias/categorias.php" class="full-width">
137 <div class="navLateral-body-cl">
138 <i class="zmdi zmdi-label"></i>
139 </div>
140 <div class="navLateral-body-cr">
141 CATEGORIAS
142 </div>
143 </a>
144 </li>
145 <li class="full-width">
146 <a href=".." proyecto/proyectos.php" class="full-width">
147 <div class="navLateral-body-cl">
148 <i class="zmdi zmdi-case"></i>
149 </div>
150 <div class="navLateral-body-cr">
151 PROYECTOS
152 </div>
153 </a>
154 </li>
155 </ul>
156 </li>
157 <li class="full-width divider-menu-h"></li>
158 <li class="full-width">
159 <a href="#1" class="full-width btn-subMenu">
160 <div class="navLateral-body-cl">
161 <i class="zmdi zmdi-face"></i>
162 </div>
163 <div class="navLateral-body-cr">
164 USUARIOS
165 </div>
166 <span class="zmdi zmdi-chevron-left"></span>
167 </a>
168 <ul class="full-width menu-principal sub-menu-options">
169 <li class="full-width">
170 <a href="admin.php" class="full-width">
```

Figura. 120 Codificación de la barra de navegación lateral pt2, Fuente. Elaboración propia.

```

178 </li>
179 <li class="full-width">
180 <a href="../personal/personal.php" class="full-width">
181 <div class="navlateral-body-cl">
182 <i class="zmdi zmdi-accounts"></i>
183 </div>
184 <div class="navlateral-body-cr">
185 PERSONAL CALIFICADO
186 </div>
187 </a>
188 </li>
189 </ul>
190 </li>
191 <li class="full-width divider-menu-h"></li>
192 <li class="full-width">
193 <a href="../compras/compras.php" class="full-width">
194 <div class="navlateral-body-cl">
195 <i class="zmdi zmdi-shopping-cart"></i>
196 </div>
197 <div class="navlateral-body-cr">
198 COMPRAS
199 </div>
200 </a>
201 </li>
202 <li class="full-width divider-menu-h"></li>
203 <li class="full-width">
204 <a href="../productos/productos.php" class="full-width">

```

Figura. 121 Codificación de la barra de navegación lateral pt3, Fuente. Elaboración propia

```

198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176

```

Figura. 122 Codificación de la barra de navegación lateral pt4, Fuente. Elaboración propia.

Por parte del contenido de la página web se coloca una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, así como se muestra en la figura 123, la cual contendrá contenedores para aplicar estilos con css, además de continuar con las animaciones de la barra de navegación, se presenta la configuración para ocultar y mostrar la barra de navegación. Se programa la interfaz de cierre de sesión mediante la implementación de un botón con identificador. Posteriormente se coloca una imagen con el avatar de identificación del administrador y se manda a llamar los datos propios de este, desde la base de datos Firebase.

```
admin.php
239 <!-- contenido de pagina -->
240 <section class="full-width pageContent">
241 <!-- navBar -->
242 <div class="full-width navBar">
243 <div class="full-width navBar-options">
244 <i class="zmdi zmdi-swap btn-menu" id="btn-menu"></i>
245 <div class="mdl-tooltip" for="btn-menu">Ocultar/Mostar Menu</div>
246 <nav class="navBar-options-list">
247 <ul class="list-unstyled">
248 <li class="btn-exit" id="btn-exit">
249 <i class="zmdi zmdi-power"></i>
250 <div class="mdl-tooltip" for="btn-exit">Cerrar Sesion</div>
251 </li>
252 <li class="text-condensedLight noLink"><small><small><?php echo $_SESSION['usuario']['NombreCompleto']></small></small>&nbsp;<small><?php echo $_SESSION['usuario']['Apellido']></small></small></li>
253 <li class="noLink">
254 <figure>
255 
256 </figure>
257 </li>
258 </ul>
259 </nav>
260 </div>
261 </div>
262
```

Figura. 123 Creación de los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.

Las animaciones correspondientes al cierre de sesión y ocultar/mostrar la barra de navegación son estructuradas en el archivo “usuarios.js” ubicado en la carpeta assets/js, como se muestra en la figura 124.

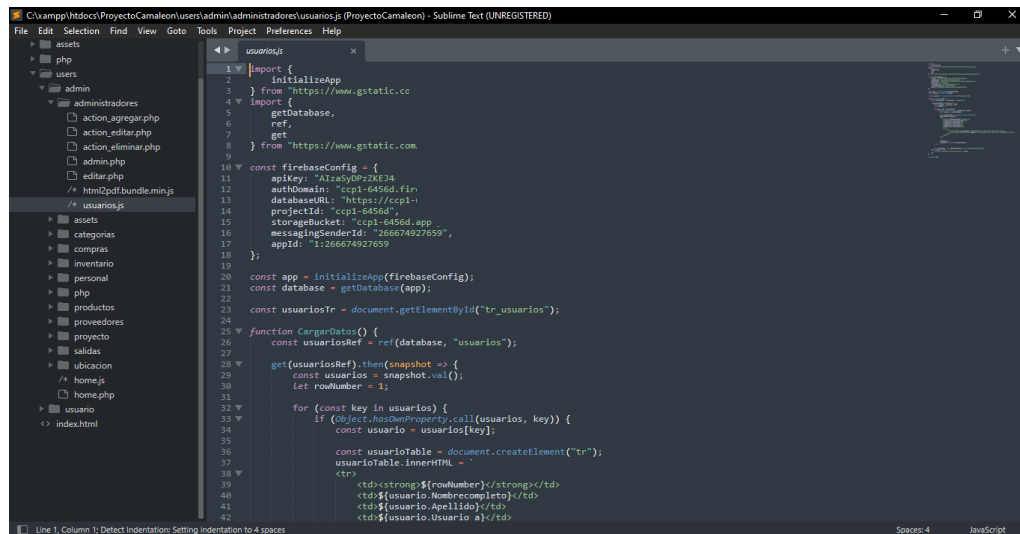


Figura. 124 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente, Elaboración propia.

A su vez se coloca un nuevo contenedor el cual contendrá dos enlaces que permitirán navegar dentro de la misma página, uno para visualizar el contenido de los registros del formulario en una tabla y el segundo que muestra el formulario a ser completado, así como se muestra en la figura 125.

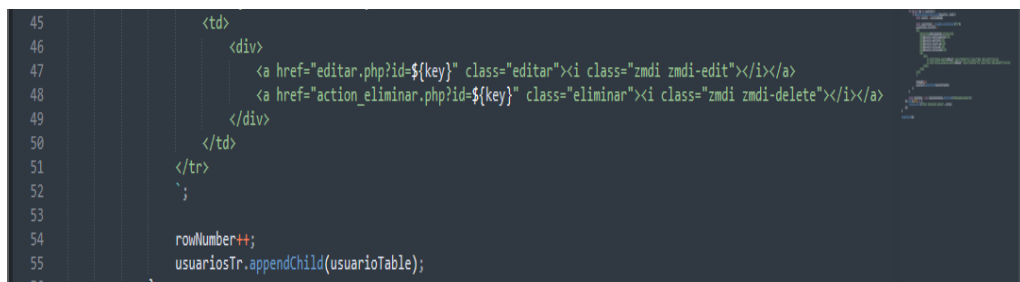


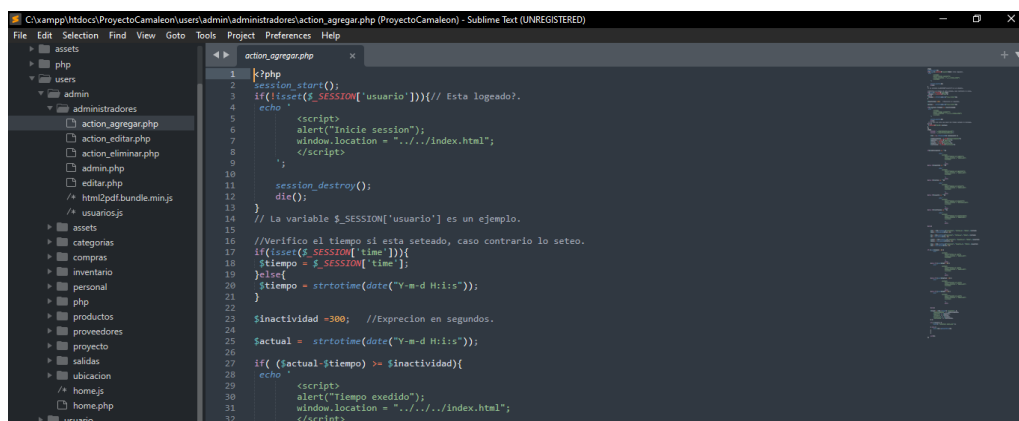
Figura. 125 Codificación de enlaces de edición y eliminación, Fuente, Elaboración propia.

En la figura 129 muestra que, al finalizar las inserciones de los input se crea un botón dentro de un contenedor que comenzara el procesamiento de información del formulario.

```
<div class="text-center">
  <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
    mdl-button--colored bg-primary" id="btn-addAdmin">
    <i class="zmdi zmdi-plus"></i>
  </button>
  <div class="mdl-tooltip" for="btn-addAdmin">Agregar Administrador</div>
</div>
```

Figura. 129 Creación del botón agregar administrador, Fuente, Elaboración propia.

Para el procesamiento de datos este completo el uso del archivo “action_agregar.php” cumple con la función de verificar el tiempo de inactividad dentro de la página y almacenar los datos obtenidos de los inputs en variables, las cuales serán evaluadas mediante sentencias condicionales identificando si la variable recibió información o esta es nula, si a información obtenida resulta ser nula se regresará al archivo “admin.php”, así como se muestra en la figura 130.



```
1 <?php
2 session_start();
3 if(!isset($_SESSION['usuario'])){// Esta logeado?.
4   echo '
5   <script>
6     alert("Inicie session");
7     window.location = "../index.html";
8   </script>
9   '
10  }
11  session_destroy();
12  die();
13  }
14  // La variable $_SESSION['usuario'] es un ejemplo.
15
16  //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17  if(!isset($_SESSION['time'])){
18    $tiempo = $_SESSION['time'];
19  }else{
20    $tiempo = strtotime(date("Y-m-d H:i:s"));
21  }
22
23  $inactividad =300; //Expresion en segundos.
24
25  $actual = strtotime(date("Y-m-d H:i:s"));
26
27  if( ($actual-$tiempo) >= $inactividad){
28    echo '
29    <script>
30      alert("Tiempo excedido");
31      window.location = "../index.html";
32    </script>
```

Figura. 130 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente, Elaboración propia.

Por lo tanto, si la información es correcta se mandará a llamar a las tablas de administradores y de personal, así como se muestra en la figura 131.

```
103
104
105     exit;
106
107 }else{
108
109
110     $ret = $rdb->retrieve("/usuarios", "Correo_a", "EQUAL", $correo);
111     $dato = json_decode($ret, 1);
112
113     $re = $rdb->retrieve("/personal", "Correo_p", "EQUAL", $correo);
114     $da = json_decode($re, 1);
115
116     $retri = $rdb->retrieve("/usuarios", "Usuario_a", "EQUAL", $usuario);
117     $datos = json_decode($retri, 1);
118
119     $res = $rdb->retrieve("/personal", "Usuario_p", "EQUAL", $usuario);
120     $das = json_decode($res, 1);
121
122
```

Figura. 131 Codificación para el llamado de tablas en Firebase, Fuente. Elaboración propia.

Por consiguiente, si la información es correcta se verificará por medio de sentencias de decisión si es que los datos proporcionados no están almacenados dentro de las tablas de la base de datos, así como se muestra en la figura 132, si están almacenados por medio de un script regresara una alerta y redirigía al archivo admin.php.

```
122
123 if (count($dato) > 0) {
124     echo
125     <script>
126         alert("Correo en uso");
127         window.location = "admin.php";
128     </script>
129     ';
130     exit;
131 }else if (count($da) > 0) {
132     echo
133     <script>
134         alert("Correo en uso");
135         window.location = "admin.php";
136     </script>
137     ';
138     exit;
139 }else if (count($datos) > 0) {
140     echo
141     <script>
142         alert("Usuario en uso");
143         window.location = "admin.php";
144     </script>
145     ';
146     exit;
147 }else if (count($das) > 0) {
148     echo
149     <script>
150         alert("Usuario en uso");
151         window.location = "admin.php";
152     </script>
153     ';
154     exit;
155
```

Figura. 132 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia.

Por otro lado, en caso de no estar almacenados, se insertará por medio de una variable que almacenará los datos correctos y devolverá al archivo admin.php, así como se muestra en la figura 133.

```

162 } else {
163
164     $insert = $rdb->insert( "usuarios", [
165         "Nombrecompleto" => $nombrecompleto,
166         "Apellido" => $apellido,
167         "Correo_a" => $correo,
168         "Usuario_a" => $usuario,
169         "Contraseña" => $contrasena,
170     ] );
171
172     if ( $insert ) {
173         header( "Location: admin.php" );
174     } else {
175         echo $rdb->getLastError();
176     }
177 }
178 }
179 }
180
181 exit();
182 }>
183

```

Figura. 133 Codificación de ingreso de datos a la tabla usuarios, Fuente. Elaboración propia.

Como se muestra en la figura 134, una vez realizado el ingreso de información se coloca un contenedor el cual almacenara el título de la lista de registros y se da formato a la tabla de registros con los campos acorde a los inputs y registros de la base de datos, se asignan identificadores para dar formato mediante css y hacer uso de Java Script.

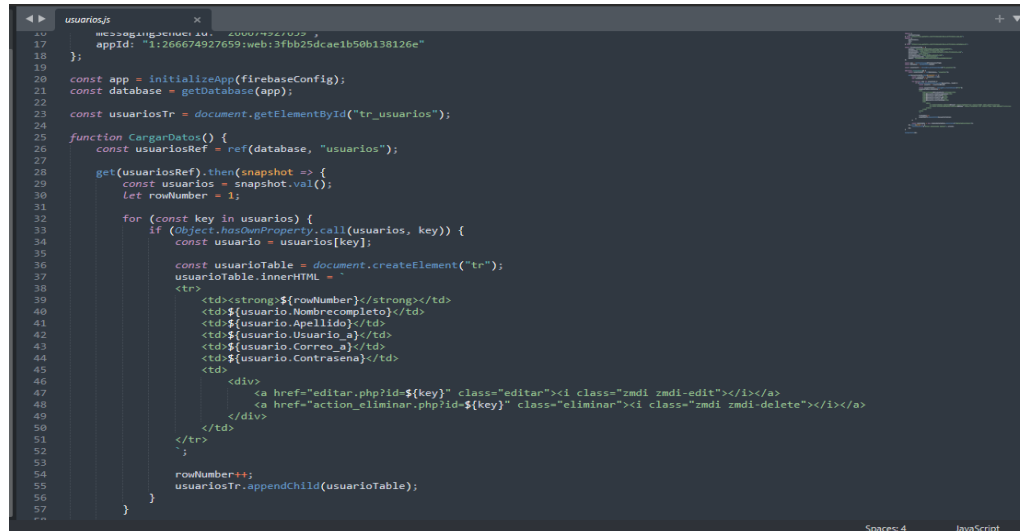
```

343
344 <div class="mdl-tabs_panel" id="tablistAdmin">
345 <div class="mdl-grid">
346 <div class="mdl-cell mdl-cell--12-col">
347 <div class="full-width panel mdl-shadow--2dp">
348 <div class="full-width panel-title bg-success text-center titles">
349     Lista de Administradores
350 </div>
351 <div class="full-width panel-content">
352 <div class="table-responsive text-center">
353 <table class="table table-bordered table-hover" id="datatablesSimple">
354 <thead>
355 <tr>
356 <th scope="col"><strong>Id</strong></th>
357 <th scope="col"><strong>Nombre(s)</strong></th>
358 <th scope="col"><strong>Apellido(s)</strong></th>
359 <th scope="col"><strong>Usuario</strong></th>
360 <th scope="col"><strong>Correo</strong></th>
361 <th scope="col"><strong>Contraseña</strong></th>
362 <th scope="col"><strong>Acción</strong></th>
363 </tr>
364 </thead>
365 <tbody id="tr_usuarios">
366 </tbody>
367 </table>
368 <button id="btnExportar" class="btn btn-success">
369 <i class="fas fa-file-excel"></i> Exportar datos a Excel
370 </button>
371 </div>
372 </div>
373 </div>
374 </div>
375 </div>
376 </div>
377 </div>
378 </section>
379

```

Figura. 134 Creación de la tabla de registros para administradores, Fuente. Elaboración propia.

Para llenar la tabla de los registros se ocupa el archivo “usuario.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros de cada insert para almacenarlos y mostrarlos en el orden indicado con anterioridad, así como se muestra en la figura 135.



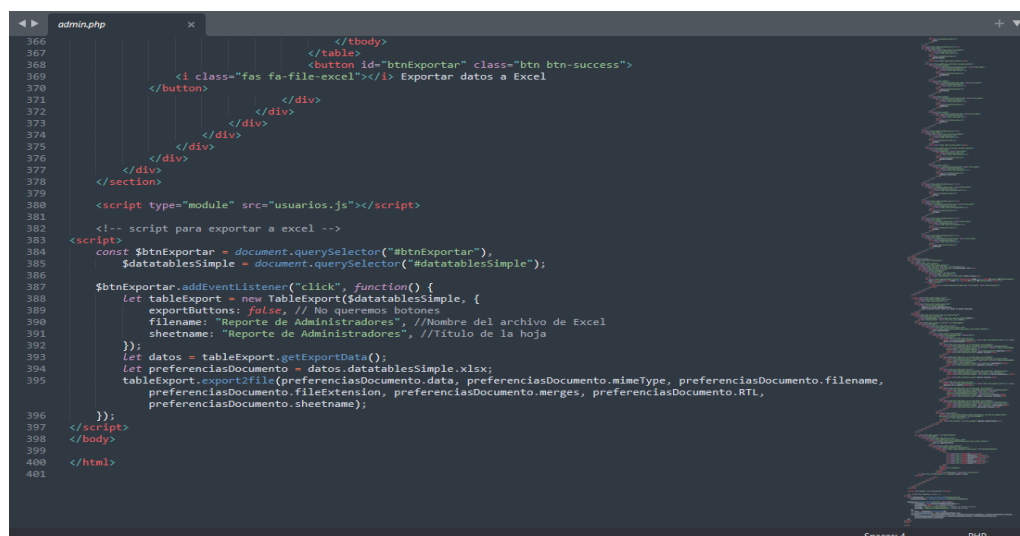
```

17  appId: "1:266674927659:web:3fbb25dcae1b50b138126e"
18  };
19
20  const app = initializeApp(firebaseConfig);
21  const database = getDatabase(app);
22
23  const usuariosTr = document.getElementById("tr_usuarios");
24
25  function CargarDatos() {
26    const usuariosRef = ref(database, "usuarios");
27
28    get(usuariosRef).then(snapshot => {
29      const usuarios = snapshot.val();
30      let rowNumber = 1;
31
32      for (const key in usuarios) {
33        if (Object.hasOwnProperty.call(usuarios, key)) {
34          const usuario = usuarios[key];
35
36          const usuarioTable = document.createElement("tr");
37          usuarioTable.innerHTML = `
38            <td><strong>${rowNumber}</strong></td>
39            <td>${usuario.NombreCompleto}</td>
40            <td>${usuario.Apellido}</td>
41            <td>${usuario.Usuario_a}</td>
42            <td>${usuario.Correo_a}</td>
43            <td>${usuario.Contrasena}</td>
44            <td>
45              <div>
46                <a href="editar.php?id=${key}" class="editar"><i class="zmdi zmdi-edit"></i></a>
47                <a href="action_eliminar.php?id=${key}" class="eliminar"><i class="zmdi zmdi-delete"></i></a>
48              </div>
49            </td>
50          `;
51          rowNumber++;
52          usuariosTr.appendChild(usuarioTable);
53        }
54      }
55    });
56  }
57

```

Figura. 135 Creación y codificación para la visualización de registros en la tabla de administradores, Fuente. Elaboración propia.

De ahí que dentro de la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, mediante el uso de un script colocado posteriormente del llamado “usuarios.js”, así como se muestra en la figura 136.



```

366  </tbody>
367  </table>
368  <button id="btnExportar" class="btn btn-success">
369    <i class="fas fa-file-excel"></i> Exportar datos a Excel
370  </button>
371  </div>
372  </div>
373  </div>
374  </div>
375  </div>
376  </div>
377  </section>
378
379  <script type="module" src="usuarios.js"></script>
380
381  <!-- script para exportar a excel -->
382  <script>
383    const $btnExportar = document.querySelector("#btnExportar");
384    $datatablesSimple = document.querySelector("#datatablesSimple");
385
386    $btnExportar.addEventListener("click", function() {
387      let tableExport = new TableExport($datatablesSimple, {
388        exportButtons: false, // No queremos botones
389        filename: "Reporte de Administradores", //Nombre del archivo de Excel
390        sheetname: "Reporte de Administradores", //Título de la hoja
391      });
392      let datos = tableExport.getExportData();
393      let preferenciasDocumento = datos.datatablesSimple.xlsx;
394      tableExport.exportToTable(tableExport.getExportData(), preferenciasDocumento.filename,
395        preferenciasDocumento.sheetname);
396    });
397  </script>
398  </body>
399  </html>
400
401

```

Figura. 136 Creación y codificación del botón de conversión a formato Excel, Fuente. Elaboración propia.

En relación con el archivo `admin.php` se encuentra “`editar.php`” perteneciente a la misma carpeta, se encuentran las configuraciones iniciales de la barra de navegación, no obstante, se hace un llamado a la conexión de la base de datos dentro de la interpretación de PHP, se especifica la tabla a la cual se tendrá acceso de información, así como se muestra en la figura 137.

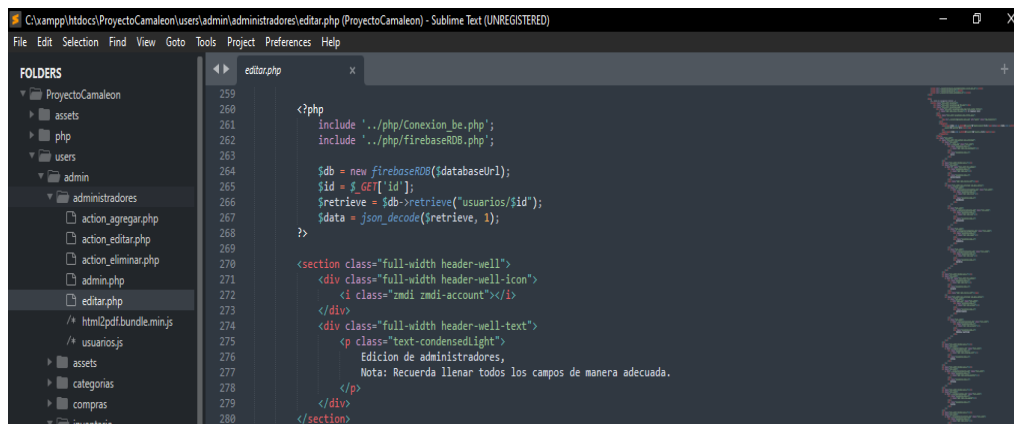


Figura. 137 Creación y configuración del archivo editar.php. Fuente. Elaboración propia.

Como se muestra en la figura 138, los inputs reciben una alteración de código, permitiendo que los datos almacenados con anterioridad en los campos designados sean llamados y visualizados dentro del mismo input, se conservan las reglas de patrones indicados por el archivo antecesor admin.php.

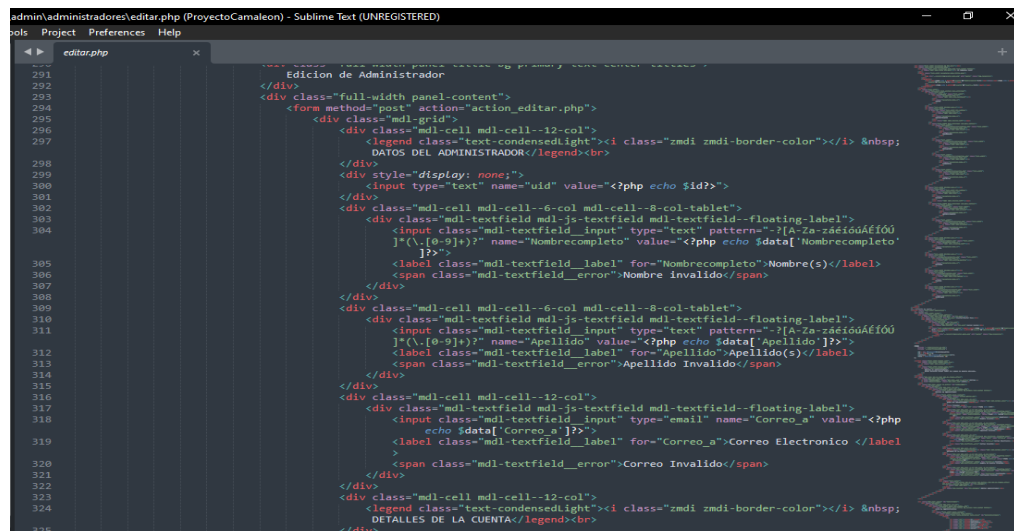


Figura. 138 Codificación y llamado de registros en los campos del formulario administrador, Fuente. Elaboración propia.

Se coloca un botón permitiendo la redirección al archivo “action_editar”, como se muestra en la figura 139, continuando con el proceso de actualización de datos del formulario.

```

337 <span class="mdl-textfield_error">Contraseña Invalida</span>
338 </div>
339 </div>
340 </div>
341 <div class="text-center">
342 <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
mdl-button--colored bg-primary" id="btn-addAdmin">
343 <i class="zmdi zmdi-plus"></i>
344 </button>
345 <div class="mdl-tooltip" for="btn-addAdmin">Editar Administrador</div>
346 </div>
347 </form>
348 </div>
349 </div>
350 </div>
351 </div>
352 </div>
353

```

Figura. 139 Botón editar administrador, Fuente. Elaboración propia.

Como se muestra en la figura 140, al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad.

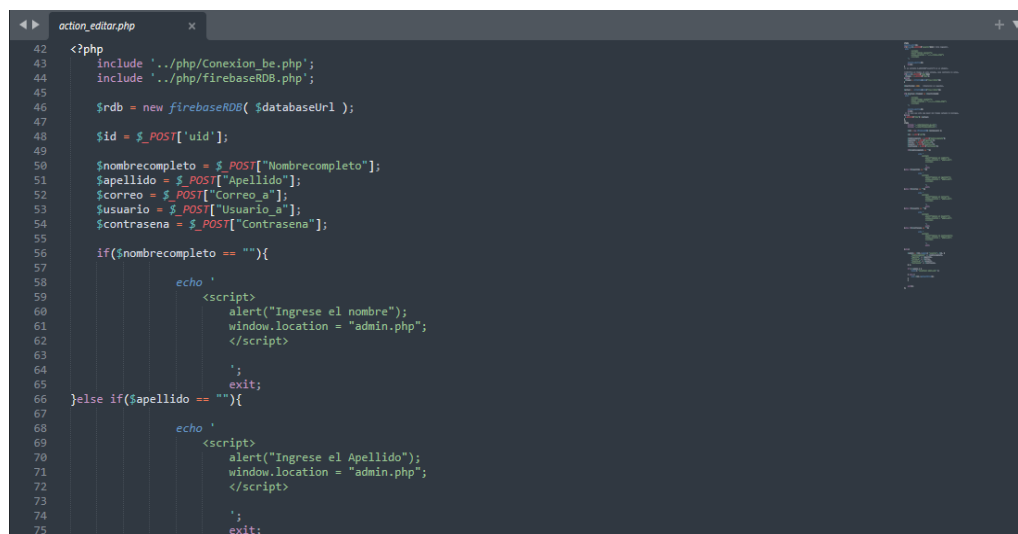
```

1 <?php
2 session_start();
3 if(isset($_SESSION['usuario'])){// Esta logeado?.
4     echo '
5         <script>
6             alert("Inicie session");
7             window.location = "../index.html";
8         </script>
9     '
10
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Exprecion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28     echo '
29         <script>
30             alert("Tiempo excedido");
31             window.location = "../index.html";
32         </script>
33     '
34
35     session_destroy();
36     die();
37 // En caso que este sea mayor del tiempo seteado lo desloga.
38 }else{
39     $_SESSION['time'] = $actual;
40 }
41 }
42 <?php

```

Figura. 140 Codificación de tiempo de inactividad del archivo action_editar.php. Fuente, Elaboración propia.

Se hace el llamado a la conexión de la base de datos y creación de variables que almacenaran los cambios indicados en los input, a su vez se verifica que dichos inputs contengan información ya que al no contenerla se coloca una alerta y re direccionamiento al archivo “admin.php”, así como se muestra en la figura 141.



```

42 <?php
43 include '../php/Conexion_be.php';
44 include '../php/firebaseRDB.php';
45
46 $rdb = new firebaseRDB( $databaseUrl );
47
48 $id = $_POST['uid'];
49
50 $nombrecompleto = $_POST['Nombrecompleto'];
51 $apellido = $_POST['Apellido'];
52 $correo = $_POST['Correo_a'];
53 $usuario = $_POST['Usuario_a'];
54 $contrasena = $_POST['Contrasena'];
55
56 if($nombrecompleto == ""){
57
58     echo '
59
60     <script>
61     alert("Ingrese el nombre");
62     window.location = "admin.php";
63     </script>
64
65     ';
66     exit;
67 }else if($apellido == ""){
68
69     echo '
70
71     <script>
72     alert("Ingrese el Apellido");
73     window.location = "admin.php";
74     </script>
75
76     ';
77     exit;

```

Figura. 141 Conexión a la base de datos y verificación para datos repetidos de los campos, Fuente. Elaboración propia.

En cambio, al proceso de ingreso de información, se anula la evaluación de datos repetidos y se comienza el proceso de actualización de datos, así como se muestra en la figura 142, el cual una vez finalizado dará redireccionamiento al archivo admin.php.



```

97
98 echo '
99
100 <script>
101 alert("Ingrese la Contraseña");
102 window.location = "admin.php";
103 </script>
104
105 ';
106 exit;
107 }else{
108
109 $update = $rdb->update( "usuarios", $id, [
110     "Nombrecompleto" => $nombrecompleto,
111     "Apellido" => $apellido,
112     "Correo_a" => $correo,
113     "Usuario_a" => $usuario,
114     "Contrasena" => $contrasena,
115 ] );
116
117 if ( $update ) {
118     header( "Location: admin.php" );
119
120 } else {
121     echo $rdb->getLastError();
122 }
123
124
125 exit();
126
127 ?>
128

```

Figura. 142 Codificación de ingreso de datos actualizados a la tabla usuarios, Fuente. Elaboración propia.

El llenado de la tabla de los registros se ocupa el archivo “usuario.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros que fueron actualizados y mostrarlos en el orden indicado con anterioridad, así como se muestra en la figura 143.

```

354 <div class="mdl-tabs_panel" id="tabListAdmin">
355 <div class="mdl-grid">
356 <div class="mdl-cell mdl-cell--12-col">
357 <div class="full-width panel mdl-shadow--2dp">
358 <div class="full-width panel-title bg-success text-center titles">
359 Lista de Administradores
360 </div>
361 <div class="full-width panel-content">
362 <div class="table-responsive text-center">
363 <table class="table table-bordered table-hover" id="datatablesSimple">
364 <thead>
365 <tr>
366 <th scope="col"><strong>Id</strong></th>
367 <th scope="col"><strong>Nombre(s)</strong></th>
368 <th scope="col"><strong>Apellido(s)</strong></th>
369 <th scope="col"><strong>Usuario</strong></th>
370 <th scope="col"><strong>Correo</strong></th>
371 <th scope="col"><strong>Contraseña</strong></th>
372 <th scope="col"><strong>Acción</strong></th>
373 </tr>
374 </thead>
375 <tbody id="tr_usuarios">
376 </tbody>
377 </table>
378 </div>
379 </div>
380 </div>
381 </div>
382 </div>
383 </div>
384 </div>
385 </section>
386 <script type="module" src="usuarios.js"></script>
387 </body>
388 </html>
389
390
391

```

Figura. 143 Creación de la tabla de registros actualizados para administradores, Fuente. Elaboración propia.

Por otra parte, así como se observa en la figura 144, se encuentra el archivo “action_eliminar” ubicado dentro de la misma carpeta de administración, Al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad.

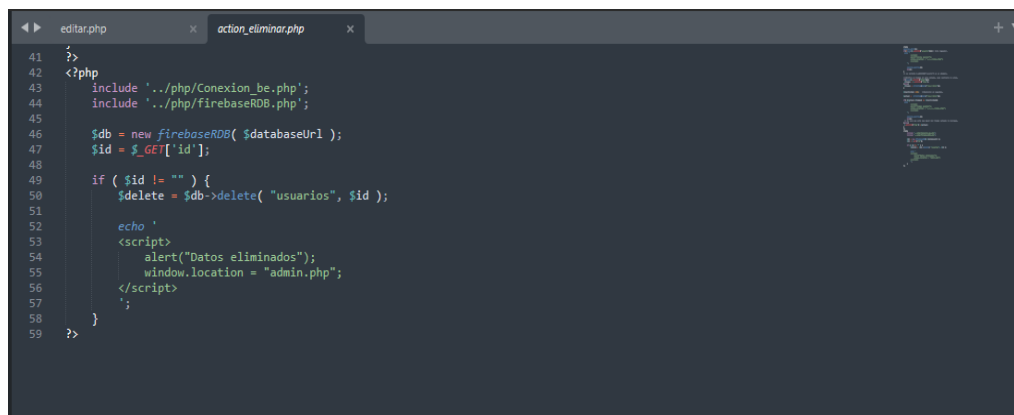
```

1 <?php
2 session_start();
3 if(isset($_SESSION['usuario'])){// Esta logeado?.
4 echo '
5 <script>
6 alert("Inicio session");
7 window.location = "../index.html";
8 </script>
9 '
10
11 session_destroy();
12 die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18 $tiempo = $_SESSION['time'];
19 }else{
20 $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Exprecion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28 echo '
29 <script>
30 alert("Tiempo excedido");
31 window.location = "../index.html";
32 </script>
33 '
34
35 session_destroy();
36 die();
37 // En caso que este sea mayor del tiempo seteado lo desloga.
38 }else{
39 $_SESSION['time'] = $actual;
40 }
41 }?>
42 <?php

```

Figura. 144 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente, Elaboración propia.

Se hace el llamado a la conexión de la base de datos y se comienza el borrado de registro mediante una sentencia de evaluación que hace uso del identificador del registro y la variable delete, así como se muestra en la figura 145, una vez finalizada la eliminación se coloca una alerta sobre la eliminación y se redirección al archivo “admin.php”.

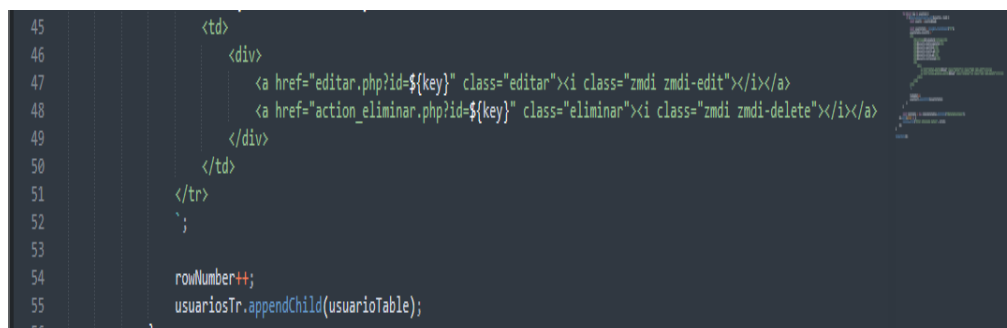


```

41 }
42 <?php
43 include '../php/Conexion_be.php';
44 include '../php/firebaseROB.php';
45
46 $db = new firebaseROB( $databaseUrl );
47 $id = $_GET['id'];
48
49 if ( $id != "" ) {
50     $delete = $db->delete( "usuarios", $id );
51
52     echo '
53     <script>
54         alert("Datos eliminados");
55         window.location = "admin.php";
56     </script>
57     ';
58 }
59 ?>
    
```

Figura. 145 Codificación y eliminación de registros de la tabla usuario, Fuente. Elaboración propia.

Para que las funcionalidades de edición y eliminación empiecen su proceso, dentro de la tabla de visualización de registros se visualizan 2 botones de acción para cada proceso correspondiente, dichos botones son ubicados en el archivo “usuarios.js”, así como se observa en la figura 146.



```

45 <td>
46     <div>
47         <a href="editar.php?id={$key}" class="editar"><i class="zmdi zmdi-edit"></i></a>
48         <a href="action_eliminar.php?id={$key}" class="eliminar"><i class="zmdi zmdi-delete"></i></a>
49     </div>
50 </td>
51 </tr>
52 `;
53
54 rowNum++;
55 usuariosTr.appendChild(usuarioTable);
56
    
```

Figura. 146 Codificación de enlaces de edición y eliminación, Fuente. Elaboración propia.

3.10.4. Carpeta Assets:

Por otra parte, los estilos, imágenes, fuentes, iconos y js se encuentran dentro de la carpeta assets, así como se muestra en la figura 147.

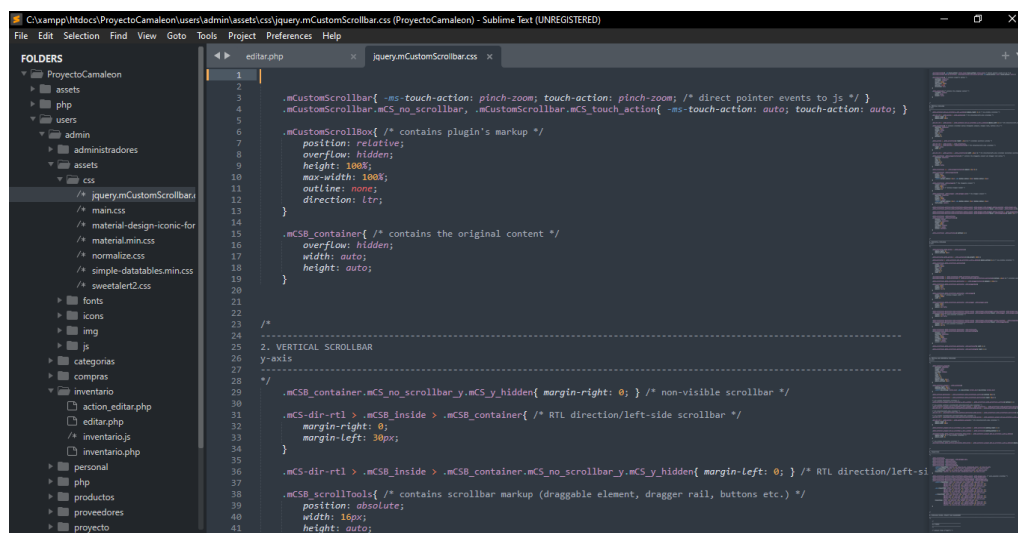


Figura. 147 Contenido de la carpeta assets, Fuente. Elaboración propia.

Por otra parte, los estilos, imágenes, fuentes, iconos y js se encuentran dentro de la carpeta assets.

Primeramente, como se muestra en la figura 148, dentro de esta carpeta se encuentra la subcarpeta de “css” la cual almacena todos los archivos pertenecientes a los estilos de las páginas web.

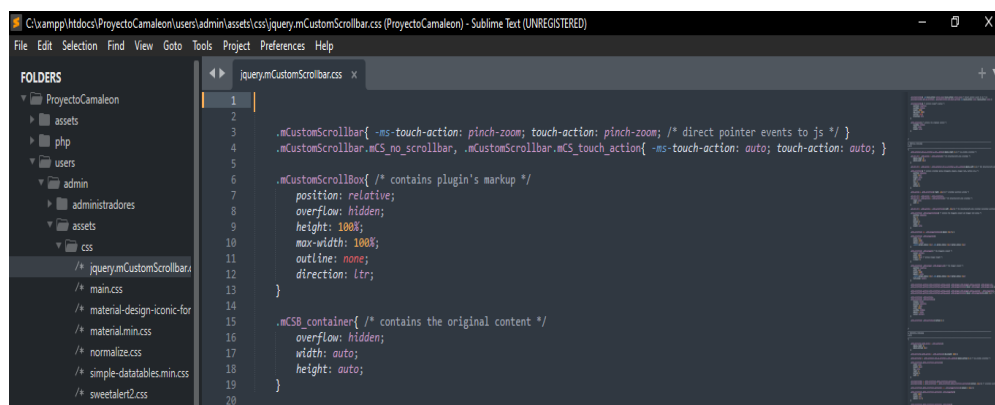


Figura. 148 Contenido de la carpeta CSS, Fuente. Elaboración propia.

Por consiguiente, como se muestra en la figura 149, se encuentra la carpeta “fonts” la cual almacena las fuentes de letra utilizadas en cada una de las páginas web.

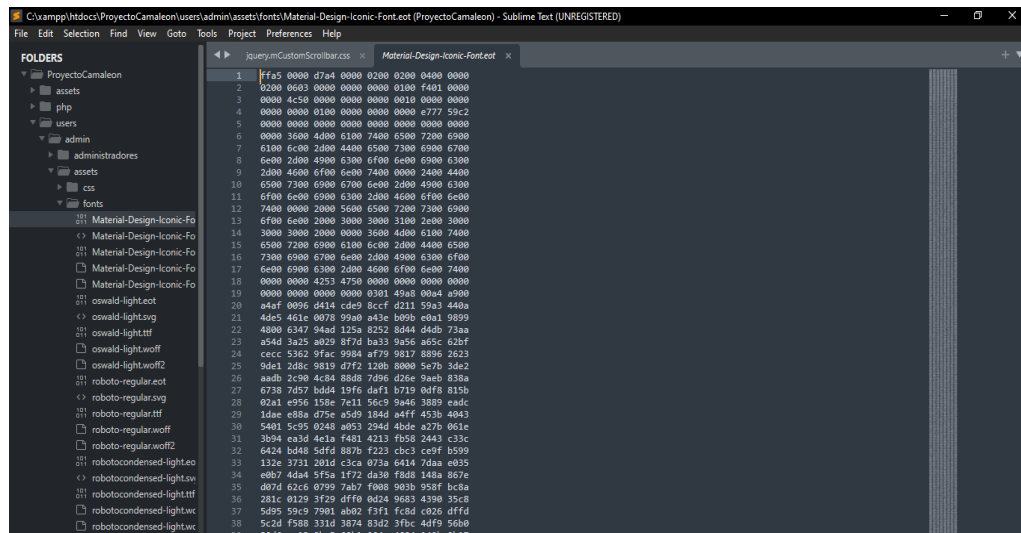


Figura. 149 Contenido de la carpeta fonts, Fuente. Elaboración propia.

Se encuentra la carpeta de “icons” que guarda los iconos de acción para las páginas web, así como se muestra en la figura 150.

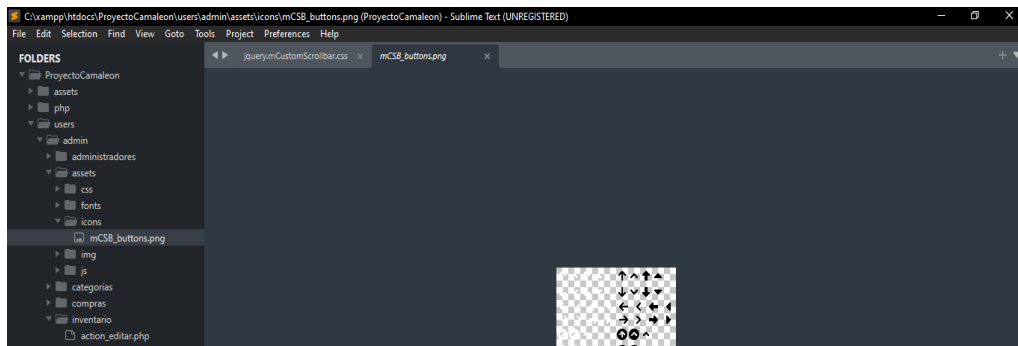


Figura. 150 Contenido de la carpeta icons, Fuente. Elaboración propia.

A su vez se encuentra la carpeta “img” la cual contiene las imágenes propias de la barra de navegación, fondos, etc., así como se muestra en la figura 151.

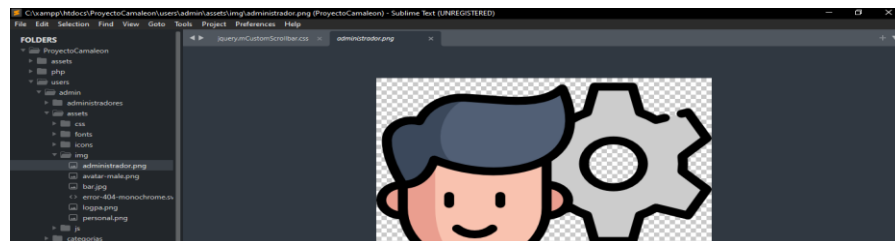


Figura. 151 Contenido de la carpeta img., Fuente. Elaboración propia.

Finalizando con la carpeta “js” que almacena los archivos JavaScript, así como se muestra en la figura 152, necesarios para las funcionalidades de animaciones y procesos de datos.

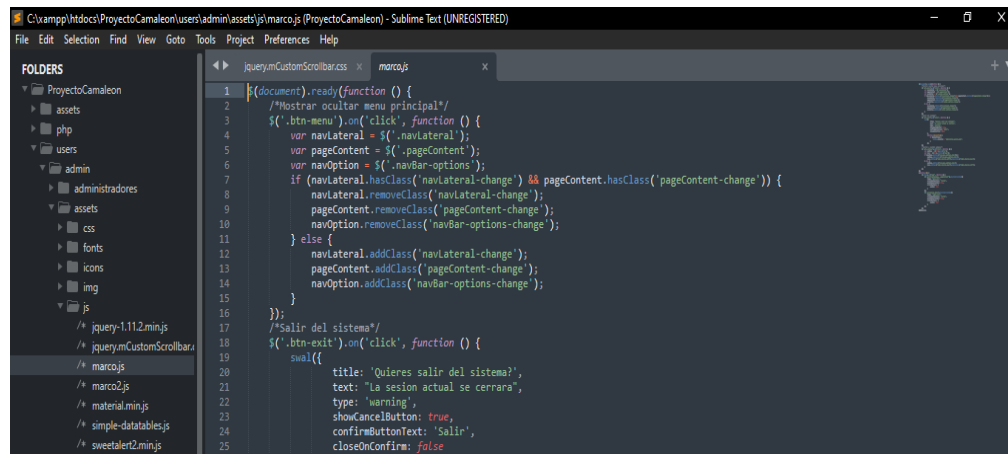


Figura. 152 Contenido de la carpeta js, Fuente. Elaboración propia.

3.10.5. Categorías:

Se muestra en la figura 153, por parte del archivo “categorias.php” se especifica en lenguaje de php mediante sentencias de decisión donde se evalúa si una vez en la página de home.php se autentico de manera correcta, en caso de no ser así se coloca una alerta y retornara a index.html. A si mismo evalúa el tiempo de inactividad entre navegación de archivos teniendo un conteo de 300 segundos de inactividad. Si el conteo es completado se retornará a index.html de manera forzada, en caso de navegar entre archivos el conteo se reiniciará de manera automática, Culminando con el cierre del lenguaje de php.

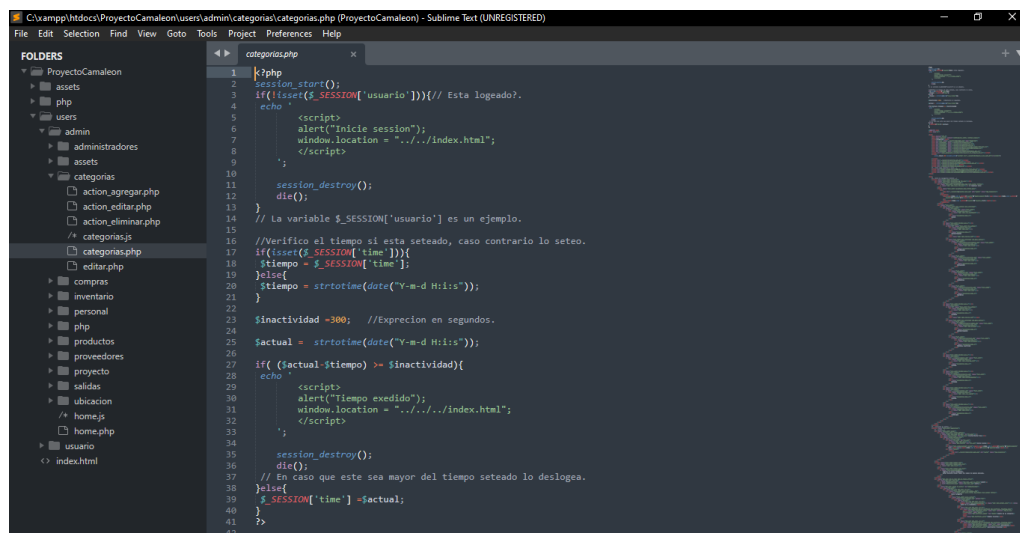
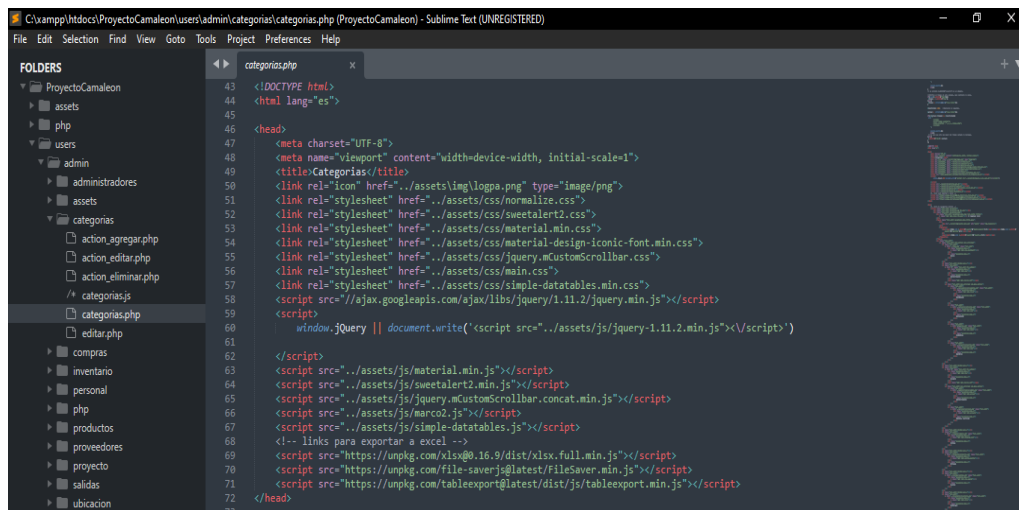


Figura. 153 Configuración de lenguaje y tiempo de inactividad del archivo categorias.php, Fuente. Elaboración propia.

Posteriormente se coloca la estructura de `<!DOCTYPE html>` la cual informara al navegador el tipo de versión HTML que fue implementado en el archivo, a su vez se indica con una etiqueta de tipo lang el tipo de idioma con el que será escrita la página web en este caso se hace uso del idioma español. A su vez se muestra la apertura de un head en el cual se proveerá la información como lo es el título y los enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assets” y de las subcarpetas de “img”, “css” y “js”, así como se muestra en la figura 154.



```

43 <!DOCTYPE html>
44 <html lang="es">
45
46 <head>
47   <meta charset="UTF-8">
48   <meta name="viewport" content="width=device-width, initial-scale=1">
49   <title>Categorías</title>
50   <link rel="icon" href="..assets/img/logo.png" type="image/png">
51   <link rel="stylesheet" href="..assets/css/normalize.css">
52   <link rel="stylesheet" href="..assets/css/sweetalert2.css">
53   <link rel="stylesheet" href="..assets/css/material.min.css">
54   <link rel="stylesheet" href="..assets/css/material-design-iconic-font.min.css">
55   <link rel="stylesheet" href="..assets/css/jquery.mCustomScrollbar.css">
56   <link rel="stylesheet" href="..assets/css/main.css">
57   <link rel="stylesheet" href="..assets/css/simple-datatables.min.css">
58   <script src="..assets/js/jquery-1.11.2/jquery.min.js"></script>
59   <script>
60     window.jQuery || document.write('<script src="..assets/js/jquery-1.11.2.min.js"></script>')
61   </script>
62   <script src="..assets/js/material.min.js"></script>
63   <script src="..assets/js/sweetalert2.min.js"></script>
64   <script src="..assets/js/jquery.mCustomScrollbar.concat.min.js"></script>
65   <script src="..assets/js/marco2.js"></script>
66   <script src="..assets/js/simple-datatables.js"></script>
67   <!-- links para exportar a excel -->
68   <script src="https://unpkg.com/xlsx@0.16.9/dist/xlsx.full.min.js"></script>
69   <script src="https://unpkg.com/file-saver@latest/FileSaver.min.js"></script>
70   <script src="https://unpkg.com/tableexport@latest/dist/js/tableexport.min.js"></script>
71
72 </head>
73

```

Figura. 154 Ingreso de Título y enlaces de funciones del archivo categorías.php, Fuente. Elaboración propia.

Una vez definido el encabezado de la página mediante la etiqueta head, como se muestra en la figura 155, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web. Por lo que se coloca un elemento que ayudara a representar el contenido principal de la página web que es el “body”, se abre una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, se colocan contenedores para aplicar estilos correspondientes a las sentencias, se agregan imágenes que harán la función del avatar del administrador, así como el nombre del usuario administrador que se autentico con anterioridad los datos son llamados desde la base de datos de Firebase.

```

74 <body>
75 <!-- barra de navegacion lateral -->
76 <section class="full-width navlateral">
77 <div class="full-width navlateral-bg btn-menu"></div>
78 <div class="full-width navlateral-body">
79 <div class="full-width navlateral-body-logo text-center titles">
80 <i class="zmdi zmdi-close btn-menu"></i> CC Camaleon Color
81 </div>
82 <figure class="full-width navlateral-body-tittle-menu">
83 <div>
84 
85 </div>
86 <figcaption>
87 <span><span><?php echo $_SESSION['usuario']['NombreCompleto'];></span>&nbsp;<span><?php echo $_SESSION['
88 usuario']['Apellido'];></span></span>
89 <br>
90 <span><small><?php echo $_SESSION['usuario']['Usuario_a'];></small></span>
91 </figcaption>
92 </figure>

```

Figura. 155 Codificación del nombre y avatar del usuario, Fuente. Elaboración propia.

Se coloca una etiqueta nav para proporcionar una navegación estructurada en la página, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta assets/css, así como se observa en las figuras 156, 157 y 158.

```

100  categorias.php
101
102  <nav class="full-width">
103    <ul class="full-width list-unstyled menu-principal">
104      <li class="full-width">
105        <a href=".."home.php" class="full-width">
106          <div class="navLateral-body-cl">
107            <i class="zmdi zmdi-view-dashboard"></i>
108          </div>
109          <div class="navLateral-body-cr">
110            INICIO
111          </div>
112        </li>
113      </ul>
114      <li class="full-width divider-menu-h"></li>
115      <li class="full-width">
116        <a href="#1" class="full-width btn-subMenu">
117          <div class="navLateral-body-cl">
118            <i class="zmdi zmdi-case"></i>
119          </div>
120          <div class="navLateral-body-cr">
121            ADMINISTRACION
122          </div>
123        </li>
124      </ul>
125      <span class="zmdi zmdi-chevron-left"></span>
126      <ul class="full-width menu-principal sub-menu-options">
127        <li class="full-width">
128          <a href=".."proveedores/proveedores.php" class="full-width">
129            <div class="navLateral-body-cl">
130              <i class="zmdi zmdi-truck"></i>
131            </div>
132            <div class="navLateral-body-cr">
133              PROVEEDORES
134            </div>
135          </li>
136          <li class="full-width">
137            <a href=".."ubicacion/ubicacion.php" class="full-width">
138              <div class="navLateral-body-cl">
139                <i class="zmdi zmdi-card"></i>
140              </div>
141              <div class="navLateral-body-cr">
142                UBICACIONES
143              </div>
144            </li>
145          </ul>
146        </nav>
147      </div>
148    </div>
149  </div>
150  </div>
151  </div>
152  </div>
153  </div>
154  </div>
155  </div>
156  </div>
157  </div>
158  </div>
159  </div>
160  </div>
161  </div>
162  </div>
163  </div>
164  </div>
165  </div>
166  </div>
167  </div>
168  </div>
169  </div>
170  </div>
171  </div>
172  </div>
173  </div>
174  </div>
175  </div>
176  </div>
177  </div>
178  </div>
179  </div>
180  </div>
181  </div>
182  </div>
183  </div>
184  </div>
185  </div>
186  </div>
187  </div>
188  </div>
189  </div>
190  </div>
191  </div>
192  </div>
193  </div>
194  </div>
195  </div>
196  </div>
197  </div>
198  </div>
199  </div>
200  </div>
201  </div>
202  </div>
203  </div>
204  </div>
205  </div>
206  </div>
207  </div>
208  </div>
209  </div>
210  </div>
211  </div>
212  </div>
213  </div>
214  </div>
215  </div>
216  </div>
217  </div>
218  </div>
219  </div>
220  </div>
221  </div>
222  </div>
223  </div>
224  </div>
225  </div>
226  </div>
227  </div>
228  </div>
229  </div>
230  </div>
231  </div>
232  </div>
233  </div>
234  </div>
235  </div>
236  </div>
237  </div>
238  </div>
239  </div>
240  </div>
241  </div>
242  </div>
243  </div>
244  </div>
245  </div>
246  </div>
247  </div>
248  </div>
249  </div>
250  </div>
251  </div>
252  </div>
253  </div>
254  </div>
255  </div>
256  </div>
257  </div>
258  </div>
259  </div>
260  </div>
261  </div>
262  </div>
263  </div>
264  </div>
265  </div>
266  </div>
267  </div>
268  </div>
269  </div>
270  </div>
271  </div>
272  </div>
273  </div>
274  </div>
275  </div>
276  </div>
277  </div>
278  </div>
279  </div>
280  </div>
281  </div>
282  </div>
283  </div>
284  </div>
285  </div>
286  </div>
287  </div>
288  </div>
289  </div>
290  </div>
291  </div>
292  </div>
293  </div>
294  </div>
295  </div>
296  </div>
297  </div>
298  </div>
299  </div>
300  </div>
301  </div>
302  </div>
303  </div>
304  </div>
305  </div>
306  </div>
307  </div>
308  </div>
309  </div>
310  </div>
311  </div>
312  </div>
313  </div>
314  </div>
315  </div>
316  </div>
317  </div>
318  </div>
319  </div>
320  </div>
321  </div>
322  </div>
323  </div>
324  </div>
325  </div>
326  </div>
327  </div>
328  </div>
329  </div>
330  </div>
331  </div>
332  </div>
333  </div>
334  </div>
335  </div>
336  </div>
337  </div>
338  </div>
339  </div>
340  </div>
341  </div>
342  </div>
343  </div>
344  </div>
345  </div>
346  </div>
347  </div>
348  </div>
349  </div>
350  </div>
351  </div>
352  </div>
353  </div>
354  </div>
355  </div>
356  </div>
357  </div>
358  </div>
359  </div>
360  </div>
361  </div>
362  </div>
363  </div>
364  </div>
365  </div>
366  </div>
367  </div>
368  </div>
369  </div>
370  </div>
371  </div>
372  </div>
373  </div>
374  </div>
375  </div>
376  </div>
377  </div>
378  </div>
379  </div>
380  </div>
381  </div>
382  </div>
383  </div>
384  </div>
385  </div>
386  </div>
387  </div>
388  </div>
389  </div>
390  </div>
391  </div>
392  </div>
393  </div>
394  </div>
395  </div>
396  </div>
397  </div>
398  </div>
399  </div>
400  </div>
401  </div>
402  </div>
403  </div>
404  </div>
405  </div>
406  </div>
407  </div>
408  </div>
409  </div>
410  </div>
411  </div>
412  </div>
413  </div>
414  </div>
415  </div>
416  </div>
417  </div>
418  </div>
419  </div>
420  </div>
421  </div>
422  </div>
423  </div>
424  </div>
425  </div>
426  </div>
427  </div>
428  </div>
429  </div>
430  </div>
431  </div>
432  </div>
433  </div>
434  </div>
435  </div>
436  </div>
437  </div>
438  </div>
439  </div>
440  </div>
441  </div>
442  </div>
443  </div>
444  </div>
445  </div>
446  </div>
447  </div>
448  </div>
449  </div>
45
```

Figura. 156 Codificación de la barra de navegación lateral pt1, Fuente. Elaboración propia.

```

130
131
132         </div>
133         </div>
134         </a>
135     </li>
136     <li class="full-width">
137         <a href="categorias.php" class="full-width">
138             <div class="navlateral-body-cl">
139                 <i class="zmdi zmdi-label"></i>
140             </div>
141             <div class="navlateral-body-cr">
142                 CATEGORIAS
143             </div>
144         </a>
145     </li>
146     <li class="full-width">
147         <a href="../../proyecto/proyectos.php" class="full-width">
148             <div class="navlateral-body-cl">
149                 <i class="zmdi zmdi-case"></i>
150             </div>
151             <div class="navlateral-body-cr">
152                 PROYECTOS
153             </div>
154         </a>
155     </li>
156 </ul>
157
158 <li class="full-width divider-menu-h"></li>
159 <li class="full-width">
160     <a href="#" class="full-width btn-subMenu">
161         <div class="navlateral-body-cl">
162             <i class="zmdi zmdi-face"></i>
163         </div>
164         <div class="navlateral-body-cr">
165             USUARIOS
166         </div>
167     </a>
168     <span class="zmdi zmdi-chevron-left"></span>
169     <ul class="full-width menu-principal sub-menu-options">
170         <li class="full-width">
171             <a href="../../administradores/admin.php" class="full-width">

```

Figura. 157 Codificación de la barra de navegación lateral pt2, Fuente. Elaboración propia.

```

200
201
202     </div>
203 </li>
204 <li class="full-width divider-menu-h"></li>
205 <li class="full-width">
206     <a href="../../productos/productos.php" class="full-width">
207         <div class="navlateral-body-cl">
208             <i class="zmdi zmdi-washing-machine"></i>
209         </div>
210         <div class="navlateral-body-cr">
211             ENTRADAS
212         </div>
213     </a>
214 </li>
215 <li class="full-width divider-menu-h"></li>
216 <li class="full-width">
217     <a href="../../salidas/salidas.php" class="full-width">
218         <div class="navlateral-body-cl">
219             <i class="zmdi zmdi-view-agenda"></i>
220         </div>
221         <div class="navlateral-body-cr">
222             SALIDAS
223         </div>
224     </a>
225 </li>
226 <li class="full-width divider-menu-h"></li>
227 <li class="full-width">
228     <a href="../../inventario/inventario.php" class="full-width">
229         <div class="navlateral-body-cl">
230             <i class="zmdi zmdi-store"></i>
231         </div>
232         <div class="navlateral-body-cr">
233             INVENTARIO
234         </div>
235     </a>
236 </li>
237 </ul>
238 </div>
239 </section>
240 <!-- contenido de pagina -->
241 <section class="full-width mainContent">

```

Figura. 158 Codificación de la barra de navegación lateral pt3, Fuente. Elaboración propia.

Por parte del contenido de la página web se coloca una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, así como se muestra en la figura 159, la cual contendrá contenedores para aplicar estilos con css, además de continuar con las animaciones de la barra de navegación, se presenta la configuración para ocultar y mostrar la barra de navegación. Se programa la interfaz de cierre de sesión mediante la implementación de un botón con identificador. Posteriormente se coloca una imagen con el avatar de identificación del administrador y se manda a llamar los datos propios de este, desde la base de datos Firebase.

```

240 <!-- contenido de pagina -->
241 <section class="full-width pageContent">
242 <!-- navbar -->
243 <div class="full-width navbar">
244 <div class="full-width navbar-options">
245 <i class="zmdi zmdi-swap btn-menu" id="btn-menu"></i>
246 <div class="mdl-tooltip" for="btn-menu">Ocultar/Mostar Menu</div>
247 <nav class="navbar-options-list">
248 <ul class="list-unstyled">
249 <li class="btn-exit" id="btn-exit">
250 <i class="zmdi zmdi-power"></i>
251 <div class="mdl-tooltip" for="btn-exit">Cerrar Sesión</div>
252 </li>
253 <li class="text-condensedLight nolink"><small><small><?php echo $_SESSION['usuario']></small></small>
254 <small>&nbsp;<?php echo $_SESSION['usuario']></small></small></li>
255 <li class="nolink">
256 
257 </li>
258 </ul>
259 </nav>
260 </div>
261 </div>
262

```

Figura. 159 Botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.

Las animaciones correspondientes al cierre de sesión y ocultar/mostrar la barra de navegación son estructuradas en el archivo “marco.js”, así como se muestra en la figura 160, ubicado en la carpeta assets/js.

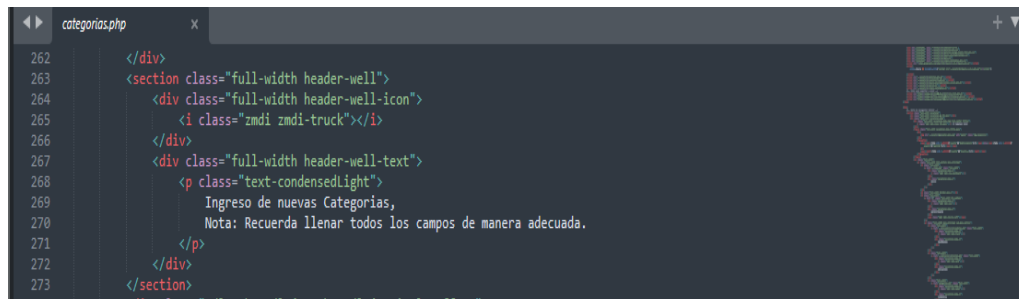
```

1 $(document).ready(function () {
2     //Mostrar ocultar menu principal*/
3     $('#btn-menu').on('click', function () {
4         var navLateral = $('#navLateral');
5         var pageContent = $('#pageContent');
6         var navOption = $('#navBar-options');
7         if (navLateral.hasClass('navLateral-change') && pageContent.hasClass('pageContent-change')) {
8             navLateral.removeClass('navLateral-change');
9             pageContent.removeClass('pageContent-change');
10            navOption.removeClass('navBar-options-change');
11        } else {
12            navLateral.addClass('navLateral-change');
13            pageContent.addClass('pageContent-change');
14            navOption.addClass('navBar-options-change');
15        }
16    });
17    //Salir del sistema*/
18    $('#btn-exit').on('click', function () {
19        swal({
20            title: 'Quieres salir del sistema?',
21            text: 'La sesión actual se cerrará',
22            type: 'warning',
23            showCancelButton: true,
24            confirmButtonText: 'Salir',
25            closeOnConfirm: false
26        },
27        function (isConfirm) {
28            if (isConfirm) {
29                window.location = 'php/cerrar_sesion.php';
30            }
31        });
32    });
33    //Mostrar y ocultar submenus*/
34    $('#btn-subMenu').on('click', function () {
35        var subMenu = $(this).next('ul');
36        var icon = $(this).children('span');
37        if (subMenu.hasClass('sub-menu-options-show')) {
38            subMenu.removeClass('sub-menu-options-show');
39            icon.addClass('zmdi-chevron-left').removeClass('zmdi-chevron-down');
40        } else {
41            subMenu.addClass('sub-menu-options-show');
42            icon.addClass('zmdi-chevron-down').removeClass('zmdi-chevron-left');
43        }
44    });
45

```

Figura. 160 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.

Posteriormente se coloca una etiqueta de section con sus respectivos contenedores para dar formato a la descripción del formulario, así como se muestra en la figura 161.

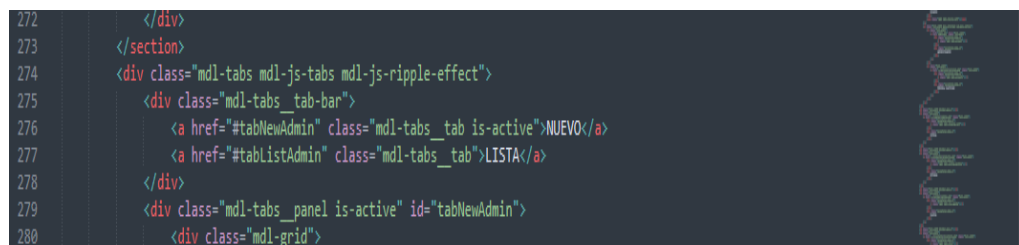


```

262     </div>
263     <section class="full-width header-well">
264       <div class="full-width header-well-icon">
265         <i class="zmdi zmdi-truck"></i>
266       </div>
267       <div class="full-width header-well-text">
268         <p class="text-condensedLight">
269           Ingreso de nuevas Categorías,
270           Nota: Recuerda llenar todos los campos de manera adecuada.
271         </p>
272       </div>
273     </section>
  
```

Figura. 161 Creación de contenedores del formulario categorías, Fuente. Elaboración propia.

A su vez se coloca un nuevo contenedor el cual contendrá dos enlaces que permitirán navegar dentro de la misma página, uno para visualizar el contenido de los registros del formulario en una tabla y el segundo que muestra el formulario a ser completado, así como se muestra en la figura 162.



```

272     </div>
273     </section>
274     <div class="mdl-tabs mdl-js-tabs mdl-js-ripple-effect">
275       <div class="mdl-tabs__tab-bar">
276         <a href="#tabNewAdmin" class="mdl-tabs__tab is-active">NUEVO</a>
277         <a href="#tabListAdmin" class="mdl-tabs__tab">LISTA</a>
278       </div>
279       <div class="mdl-tabs__panel is-active" id="tabNewAdmin">
280         <div class="mdl-grid">
  
```

Figura. 162 Creación de título del formulario e indicaciones, Fuente. Elaboración propia.


```
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
  <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
    <input class="mdl-textfield__input" type="text" pattern="-?[A-Za-záéíóúÁÉÍÓÚ]*(\.[0-9]+)?" name="Nombrecompleto">
    <label class="mdl-textfield__label" for="Nombrecompleto">Nombre(s)</label>
    <span class="mdl-textfield__error">Nombre invalido</span>
  </div>
</div>
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
```

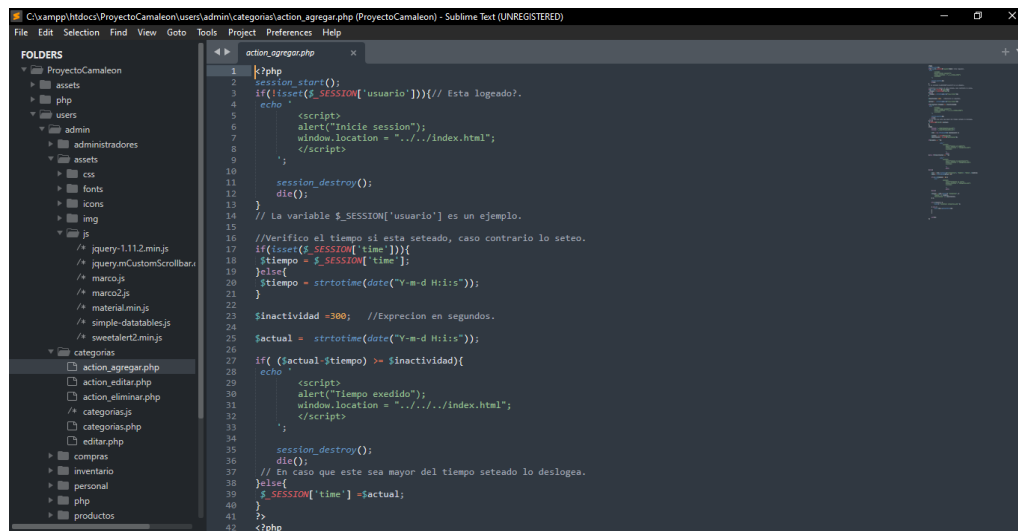
Figura. 165 Codificación de la estructura de los campos para el formulario, Fuente. Elaboración propia.

Al finalizar las inserciones del input se crea un botón dentro de un contenedor que comenzara el procesamiento de información del formulario, así como se muestra en la figura 166.

```
306
307
308
309
310
311
312
313
    <div>
    <div class="text-center">
    <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
    mdl-button--colored bg-primary" id="btn-addProvider">
    <i class="zmdi zmdi-plus"></i>
    </button>
    <div class="mdl-tooltip" for="btn-addProvider">Agregar Categoría</div>
    </div>
    </form>
```

Figura. 166 Creación del botón agregar categoría, Fuente. Elaboración propia.

Como se observa en la figura 167, para que el procesamiento de datos este completo el uso del archivo “action_agregar.php” cumple con la función de verificar el tiempo de inactividad dentro de la página y almacenar los datos obtenidos de los inputs en variables, las cuales serán evaluadas mediante sentencias condicionales identificando si la variable recibió información o esta es nula, si a información obtenida resulta ser nula se regresara al archivo “categorias.php”.



```
1 <?php
2 session_start();
3 if(!isset($_SESSION['usuario'])){// Esta logeado?.
4 echo '
5 <script>
6 alert("Inicie session");
7 window.location = "../index.html";
8 </script>
9 '
10
11 session_destroy();
12 die();
13 }
14 // La variable $_SESSION["usuario"] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18 $tiempo = $_SESSION['time'];
19 }else{
20 $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Expresion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28 echo
29 <script>
30 alert("Tiempo excedido");
31 window.location = "../index.html";
32 </script>
33 '
34
35 session_destroy();
36 die();
37 // En caso que este sea mayor del tiempo setando lo desloga.
38 }else{
39 $_SESSION['time'] = $actual;
40 }
41 }
42 <?php
```

Figura. 167 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente, Elaboración propia.

Por lo tanto, como se muestra en la figura 168, si la información es correcta se mandará a llamar a la tabla de categorías.

```

71 }else{
72
73 $ret = $rdb->retrieve("/categorias", "Nombre", "EQUAL", $nombre);
74 $dato = json_decode($ret, 1);
75

```

Figura. 168 Codificación para el llamado de tablas en Firebase, Fuente. Elaboración propia.

Por consiguiente, si la información es correcta se verificará por medio de sentencias de decisión, como se muestra en la figura 169, si es que los datos proporcionados no están almacenados dentro de las tablas de la base de datos, si están almacenados por medio de un script regresará una alerta y redirigirá al archivo categorias.php.

```

71 }else{
72
73 $ret = $rdb->retrieve("/categorias", "Nombre", "EQUAL", $nombre);
74 $dato = json_decode($ret, 1);
75
76 if (count($dato) > 0) {
77     echo '
78         <script>
79             alert("Categoria en uso");
80             window.location = "categorias.php";
81         </script>
82         '
83         ;
84         exit;
85     }else{
86

```

Figura. 169 Codificación de ingreso de datos a la tabla categorías, Fuente. Elaboración propia.

Por otro lado, como se observa en la figura 170, en caso de no estar almacenados, se insertará por medio de una variable que almacenará los datos correctos y devolverá al archivo categorias.php.

```

86
87 $insert = $rdb->insert( "categorias", [
88     "Nombre" => $nombre,
89     "Descripcion" => $descripcion,
90 ] );
91
92
93 if ( $insert ) {
94     header( "Location: categorias.php" );
95 } else {
96     echo $rdb->getLastError();
97 }
98
99
100
101
102 exit();
103
104 ?>
105

```

Figura. 170 Almacena en categorías. Fuente. Elaboración propia.

Una vez realizado el ingreso de información se coloca un contenedor el cual almacenara el título de la lista de registros y se da formato a la tabla de registros con los campos acorde a los inputs y registros de la base de datos, se asignan identificadores para dar formato mediante css y hacer uso de Java Script, así como se muestra en la figura 171.

```

317 </div>
318 </div>
319
320 <div class="mdl-tabs_panel" id="tablistProvider">
321   <div class="mdl-grid">
322     <div class="mdl-cell mdl-cell--12-col">
323       <div class="full-width panel mdl-shadow--2dp">
324         <div class="full-width panel-title bg-success text-center titles">
325           lista de categorias
326         </div>
327         <div class="full-width panel-content">
328           <div class="table-responsive text-center">
329             <table class="table table-bordered table-hover" id="datatablesSimple">
330               <thead>
331                 <tr>
332                   <th scope="col"><strong>ID</strong></th>
333                   <th scope="col"><strong>Nombre</strong></th>
334                   <th scope="col"><strong>Descripción</strong></th>
335                   <th scope="col"><strong>Acción</strong></th>
336                 </tr>
337               </thead>
338               <tbody id="tr_categorias">
339             </tbody>
340             </table>
341             <button id="btnExportar" class="btn btn-success">
342               <i class="fas fa-file-excel"></i> Exportar datos a Excel
343             </button>
344           </div>
345         </div>
346       </div>
347     </div>
348   </div>
349 </div>
350 </div>
351 </section>
  
```

Figura. 171 Creación de la tabla de registros para las categorías, Fuente. Elaboración propia.

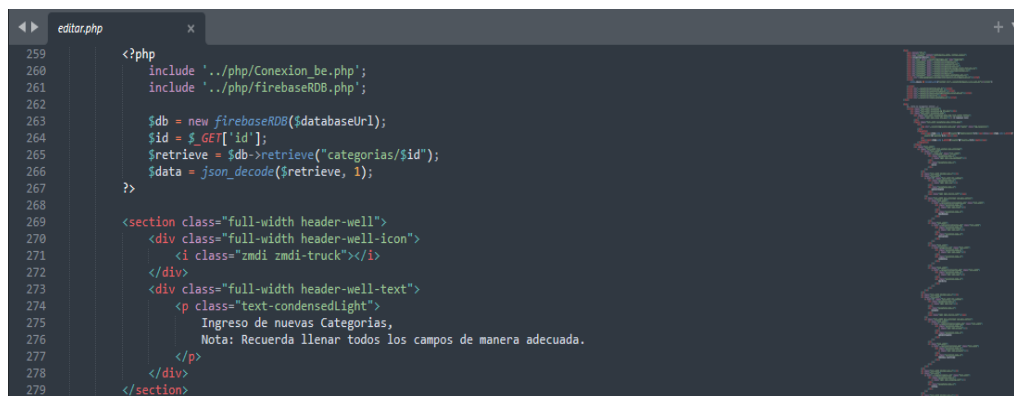
Lo que muestra la figura 172 es la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, mediante el uso de un script colocado posteriormente del llamado “categorias.js”.

```

340 </table>
341 <button id="btnExportar" class="btn btn-success">
342   <i class="fas fa-file-excel"></i> Exportar datos a Excel
343 </button>
344 </div>
345 </div>
346 </div>
347 </div>
348 </div>
349 </div>
350 </div>
351 </section>
352
353 <script type="module" src="categorias.js"></script>
354 <!-- script para exportar a excel -->
355 <script>
356   const $btnExportar = document.querySelector("#btnExportar");
357   $datatablesSimple = document.querySelector("#datatablesSimple");
358
359   $btnExportar.addEventListener("click", function() {
360     let tableExport = new TableExport($datatablesSimple, {
361       exportButtons: false, // No queremos botones
362       filename: "Reporte de Categorías", //Nombre del archivo de Excel
363       sheetname: "Reporte de Categorías", //Título de la hoja
364     });
365     let datos = tableExport.getExportData();
366     let preferenciasDocumento = datos.datatablesSimple.xlsx;
367     tableExport.export2file(preferenciasDocumento.data, preferenciasDocumento.mimeType, preferenciasDocumento.filename,
368                           preferenciasDocumento.fileExtension, preferenciasDocumento.merges, preferenciasDocumento.RTL,
369                           preferenciasDocumento.sheetname);
370   });
371 </script>
372 </body>
373 </html>
  
```

Figura. 172 Visualización de registro. Fuente. Elaboración propia.

En relación con el archivo `categorias.php` se encuentra “`editar.php`”, como se muestra en la figura 173, perteneciente a la misma carpeta, se encuentran las configuraciones iniciales de la barra de navegación, no obstante, se hace un llamado a la conexión de la base de datos dentro de la interpretación de PHP, se especifica la tabla a la cual se tendrá acceso de información.



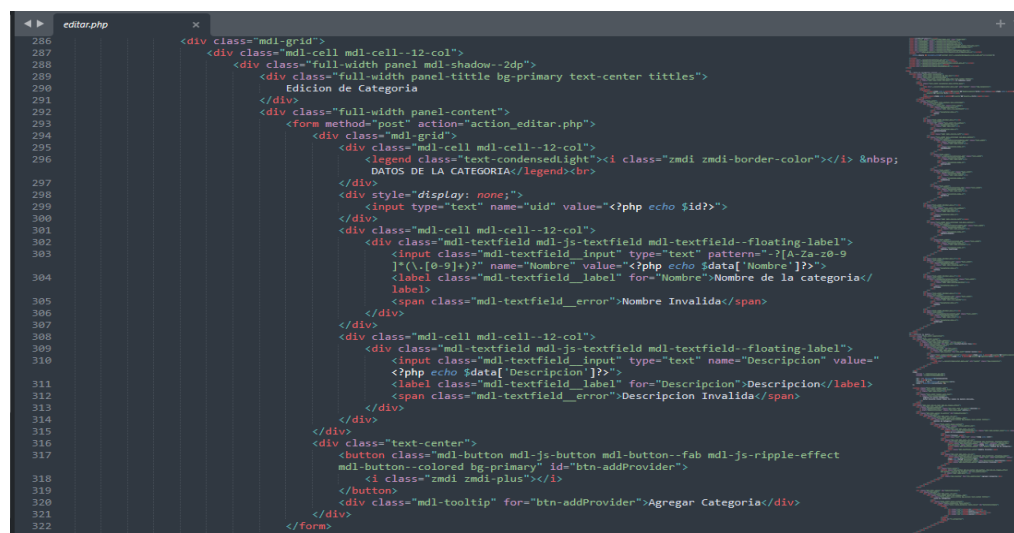
```
<?php
include '../php/Conexion_be.php';
include '../php/firebaseRDB.php';

$db = new firebaseRDB($databaseUrl);
$id = $_GET['id'];
$retrieve = $db->retrieve("categorias/$id");
$data = json_decode($retrieve, 1);
?>

<section class="full-width header-well">
  <div class="full-width header-well-icon">
    <i class="zmdi zmdi-truck"></i>
  </div>
  <div class="full-width header-well-text">
    <p class="text-condensedLight">
      Ingreso de nuevas Categorías,
      Nota: Recuerda llenar todos los campos de manera adecuada.
    </p>
  </div>
</section>
```

Figura. 173 Configuración y llamado a la base de datos de categorías, Fuente. Elaboración propia.

Posteriormente como se muestra en la figura 174, los inputs reciben una alteración de código, permitiendo que los datos almacenados con anterioridad en los campos designados sean llamados y visualizados dentro del mismo input, se conservan las reglas de patrones indicados por el archivo antecesor `categorias.php`.



```
<div class="mdl-grid">
  <div class="mdl-cell mdl-cell-12-col">
    <div class="full-width panel mdl-shadow-2dp">
      <div class="full-width panel-title bg-primary text-center titles">
        Edición de Categoría
      </div>
      <div class="full-width panel-content">
        <form method="post" action="action_editar.php">
          <div class="mdl-grid">
            <div class="mdl-cell mdl-cell-12-col">
              <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp;&nbsp;&nbsp;<br>
                DATOS DE LA CATEGORIA</legend><br>
              <div>
                <div style="display: none;">
                  <input type="text" name="uid" value="<?php echo $id>">
                </div>
                <div class="mdl-cell mdl-cell-12-col">
                  <div class="mdl-textfield mdl-js-textfield mdl-textfield-floating-label">
                    <input class="mdl-textfield_input" type="text" pattern="[A-Za-z0-9]{1}(\.[0-9]+)?" name="Nombre" value="<?php echo $data['Nombre']>">
                    <label class="mdl-textfield_label" for="Nombre">Nombre de la categoría</label>
                    <span class="mdl-textfield_error">Nombre Invalida</span>
                  </div>
                </div>
                <div class="mdl-cell mdl-cell-12-col">
                  <div class="mdl-textfield mdl-js-textfield mdl-textfield-floating-label">
                    <input class="mdl-textfield_input" type="text" name="Descripcion" value="<?php echo $data['Descripcion']>">
                    <label class="mdl-textfield_label" for="Descripcion">Descripcion</label>
                    <span class="mdl-textfield_error">Descripcion Invalida</span>
                  </div>
                </div>
              </div>
              <div class="text-center">
                <button class="mdl-button mdl-js-button mdl-button-fab mdl-js-ripple-effect mdl-button-colored bg-primary" id="btn-addProvider">
                  <i class="zmdi zmdi-plus"></i>
                </button>
                <div class="mdl-tooltip" for="btn-addProvider">Agregar Categoría</div>
              </div>
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
</div>
```

Figura. 174 Codificación de registros en el formulario de categorías, Fuente. Elaboración propia.

Se coloca un botón permitiendo la redirección al archivo “action_editar” continuando con el proceso de actualización de datos del formulario, así como se muestra en la figura 175.

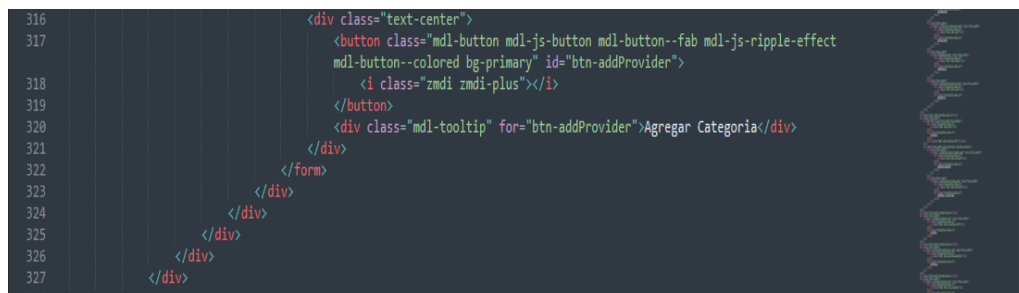


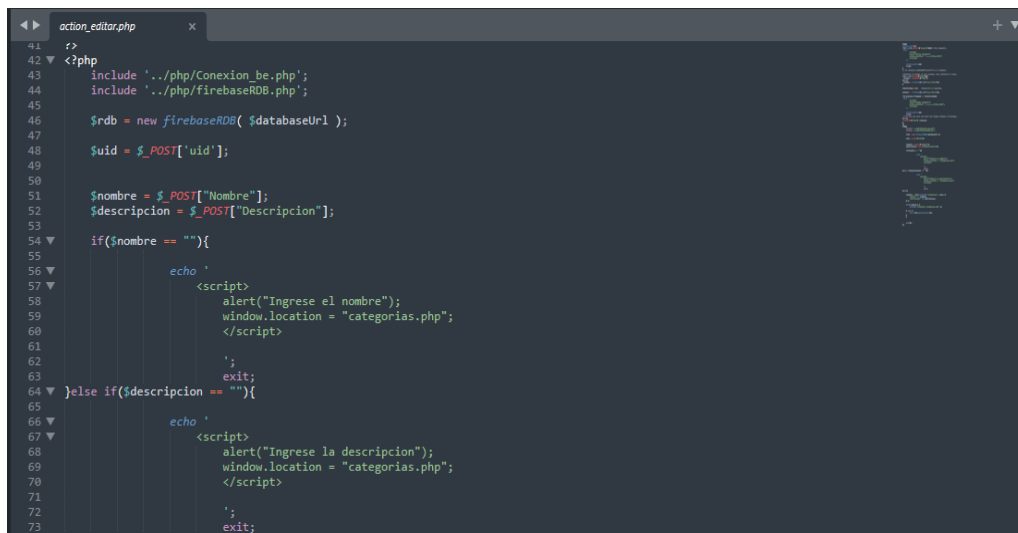
Figura. 175 Creación del botón agregar categoría, Fuente. Elaboración propia.

Al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad, como se observa en la figura 176.



Figura. 176 Codificación de tiempo de inactividad del archivo action_editar.php. Fuente, Elaboración propia.

Se hace el llamado a la conexión de la base de datos y creación de variables que almacenaran los cambios indicados en los input, como se observa en la figura 177, a su vez se verifica que dichos inputs contengan información ya que al no contenerla se coloca una alerta y re direccionamiento al archivo “categorias.php”.



```

41 <?php
42 include '../php/Conexion_be.php';
43 include '../php/firebaseRDB.php';
44
45 $rdb = new firebaseRDB( $databaseUrl );
46
47 $uid = $_POST['uid'];
48
49
50 $nombre = $_POST['Nombre'];
51 $descripcion = $_POST['Descripcion'];
52
53
54 if($nombre == ""){
55     echo '
56     <script>
57         alert("Ingresa el nombre");
58         window.location = "categorias.php";
59     </script>
60     ';
61     exit;
62 }else if($descripcion == ""){
63     echo '
64     <script>
65         alert("Ingresa la descripcion");
66         window.location = "categorias.php";
67     </script>
68     ';
69     exit;
70 }
71
72
73

```

Figura. 177 Conexión a la base de datos y verificación de campos, Fuente. Elaboración propia.

En cambio, al proceso de ingreso de información, se anula la evaluación de datos repetidos y se comienza el proceso de actualización de datos, el cual una vez finalizado dará redireccionamiento al archivo categorias.php, así como se observa en la figura 178.



```

72
73
74 }else{
75     exit;
76
77     $update = $rdb->update( "categorias", $uid, [
78         "Nombre" => $nombre,
79         "Descripcion" => $descripcion,
80     ] );
81
82     if ( $update ) {
83         header( "Location: categorias.php" );
84     } else {
85         echo $rdb->getLastError();
86     }
87
88
89     exit();
90
91

```

Figura. 178 Codificación de actualización de registros de la tabla categorías, Fuente. Elaboración propia.

Como se observa en la figura 179, el llenado de la tabla de los registros se ocupa el archivo “categorias.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros que fueron actualizados y mostrarlos en el orden indicado con anterioridad.

```

323     </div>
324   </div>
325 </div>
326 </div>
327 </div>
328
329 <div class="mdl-tabs_panel" id="tabListProvider">
330   <div class="mdl-grid">
331     <div class="mdl-cell mdl-cell--12-col">
332       <div class="full-width panel mdl-shadow--2dp">
333         <div class="full-width panel-title bg-success text-center titles">
334           Lista de categorias
335         </div>
336         <div class="full-width panel-content">
337           <div class="table-responsive text-center">
338             <table class="table table-bordered table-hover" id="dataTablesSimple">
339               <thead>
340                 <tr>
341                   <th scope="col"><strong>ID</strong></th>
342                   <th scope="col"><strong>Nombre</strong></th>
343                   <th scope="col"><strong>Descripción</strong></th>
344                   <th scope="col"><strong>Acción</strong></th>
345                 </tr>
346               </thead>
347               <tbody id="tr_categorias">
348                 <tbody>
349                 </tbody>
350               </tbody>
351             </table>
352           </div>
353         </div>
354       </div>
355     </div>
356   </div>
357 </section>
358
359 <script type="module" src="categorias.js"></script>
360 </body>
361 </html>
  
```

Figura. 179 Creación de la tabla de registros actualizados para categorías, Fuente. Elaboración propia.

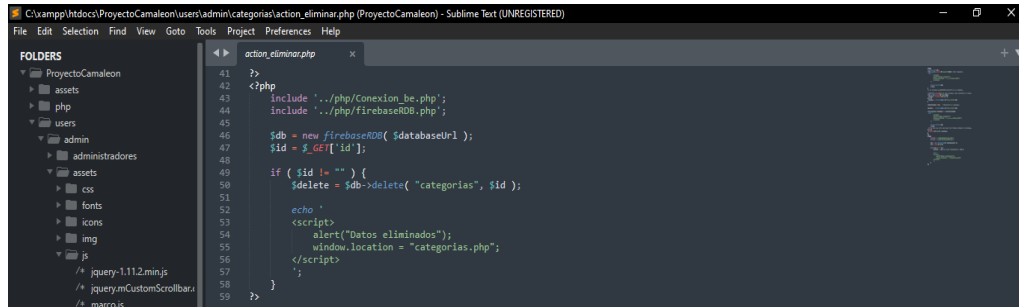
Por otra parte, como se observa en la figura 180, se encuentra el archivo “action_eliminar” ubicado dentro de la misma carpeta de categorías, Al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad.

```

1  <?php
2  session_start();
3  if(!isset($_SESSION['usuario'])){// Esta logeado?.
4    echo '
5      <script>
6        alert("Inicie session");
7        window.location = "../index.html";
8      </script>
9    ';
10
11    session_destroy();
12    die();
13  }
14  // La variable $_SESSION['usuario'] es un ejemplo.
15
16  //Verifico el tiempo si este seteado, caso contrario lo seteo.
17  if(isset($_SESSION['time'])){
18    $tiempo = $_SESSION['time'];
19  }else{
20    $tiempo = strtotime(date("Y-m-d H:i:s"));
21  }
22
23  $inactividad =300; //Expresion en segundos.
24
25  $actual = strtotime(date("Y-m-d H:i:s"));
26
27  if( ($actual-$tiempo) >= $inactividad){
28    echo '
29      <script>
30        alert("Tiempo excedido");
31        window.location = "../index.html";
32      </script>
33    ';
34
35    session_destroy();
36    die();
37    // En caso que este sea mayor del tiempo seteado lo desloga.
38  }else{
39    $_SESSION['time'] =$actual;
40  }
41  ?>
  
```

Figura. 180 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente, Elaboración propia.

Se hace el llamado a la conexión de la base de datos y se comienza el borrado de registro mediante una sentencia de evaluación que hace uso del identificador del registro y la variable delete, como se observa en la figura 181, una vez finalizada la eliminación se coloca una alerta sobre la eliminación y se redirecciona al archivo “categorias.php”.



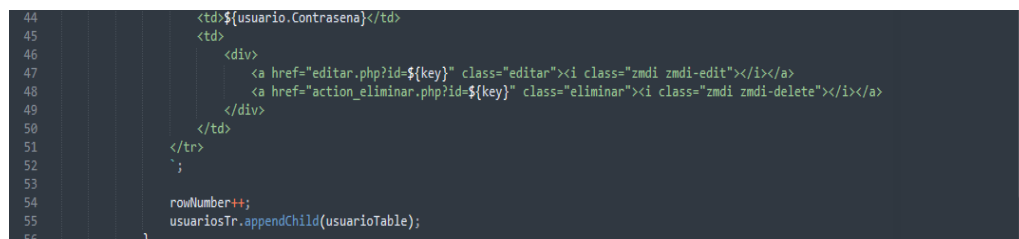
```

41  >
42  <?php
43  include '../php/Conexion_be.php';
44  include '../php/firebaseDB.php';
45
46  $db = new firebaseDB( $databaseUrl );
47  $id = $_GET['id'];
48
49  if ( $id != "" ) {
50      $delete = $db->delete( "categorias", $id );
51
52      echo '
53      <script>
54      alert("Datos eliminados");
55      window.location = "categorias.php";
56      </script>
57      ';
58  }
59  >

```

Figura. 181 Codificación y eliminación de registros de la tabla categorías, Fuente. Elaboración propia.

Para que las funcionalidades de edición y eliminación empiecen su proceso, dentro de la tabla de visualización de registros se visualizan 2 botones de acción para cada proceso correspondiente, como se observa en la figura 182, dichos botones son ubicados en el archivo “categorias.js”.



```

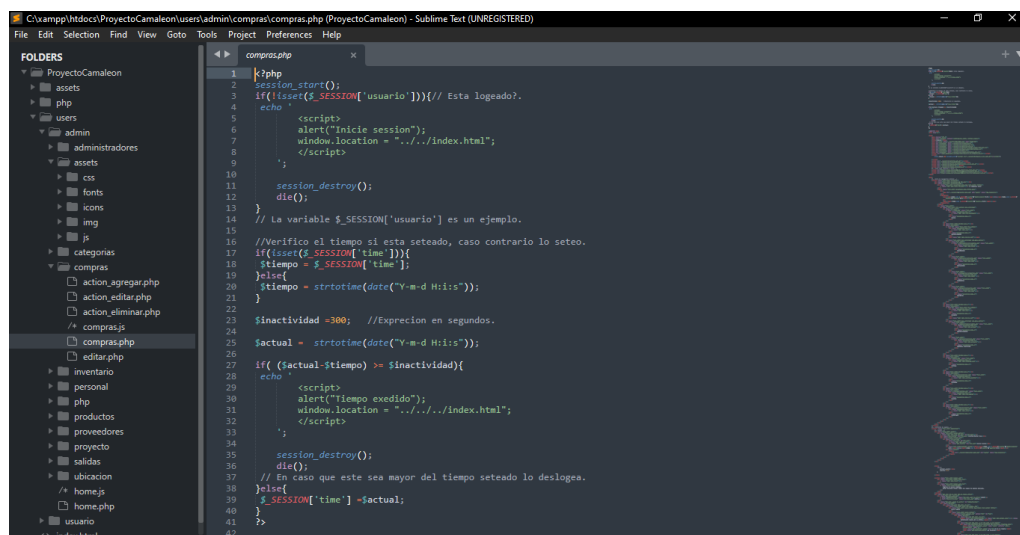
44  <td>${usuario.Contrasena}</td>
45  <td>
46      <div>
47          <a href="editar.php?id=${key}" class="editar"><i class="zmdi zmdi-edit"></i></a>
48          <a href="action_eliminar.php?id=${key}" class="eliminar"><i class="zmdi zmdi-delete"></i></a>
49      </div>
50  </td>
51  </tr>
52  `;
53
54  rowNumber++;
55  usuariosTr.appendChild(usuarioTable);
56  }

```

Figura. 182 Codificación de enlaces de edición y eliminación, Fuente. Elaboración propia.

3.10.6. Compras:

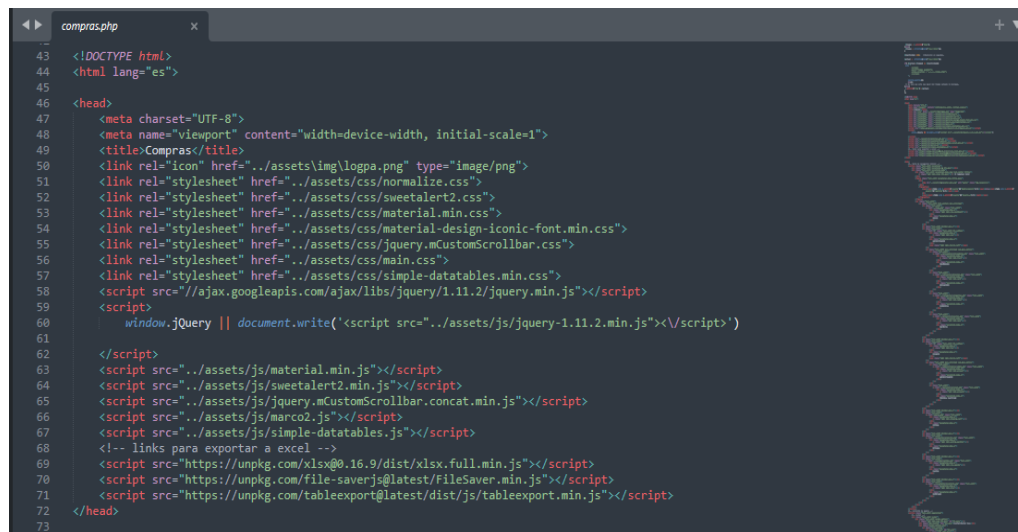
Por parte del archivo “compras.php” se especifica en lenguaje de PHP mediante sentencias de decisión donde se evalúa si una vez en la página de home.php se autentico de manera correcta, en caso de no ser así se coloca una alerta y retornara a index.html. A si mismo evalúa el tiempo de inactividad entre navegación de archivos teniendo un conteo de 300 segundos de inactividad. Si el conteo es completado se retornará a index.html de manera forzada, en caso de navegar entre archivos el conteo se reiniciará de manera automática, Culminando con el cierre del lenguaje de php, así como se muestra en la figura 183.



```
1  <?php
2  session_start();
3  if(isset($_SESSION['usuario'])){// Esta logeado?.
4      echo '
5          <script>
6              alert("Inicie session");
7              window.location = "../index.html";
8          </script>
9      '
10
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Exprecion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28     echo '
29         <script>
30             alert("Tiempo exadido");
31             window.location = "../index.html";
32         </script>
33     '
34
35     session_destroy();
36     die();
37 // En caso que este sea mayor del tiempo seteado lo desloga.
38 }else{
39     $_SESSION['time'] = $actual;
40 }
41
42 >>
```

Figura. 183 Configuración de lenguaje y tiempo de inactividad del archivo compras.php, Fuente. Elaboración propia.

Posteriormente se coloca la estructura de `<!DOCTYPE html>`, así como se muestra en la figura 184, la cual informara al navegador el tipo de versión HTML que fue implementado en el archivo, a su vez se indica con una etiqueta de tipo `lang` el tipo de idioma con el que será escrita la página web en este caso se hace uso del idioma español. A su vez se muestra la apertura de un `head` en el cual se proveerá la información como lo es el título y lo enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assets” y de las subcarpetas de “img”, “css” y “js”.



```
43 <!DOCTYPE html>
44 <html lang="es">
45
46 <head>
47   <meta charset="UTF-8">
48   <meta name="viewport" content="width=device-width, initial-scale=1">
49   <title>Compras</title>
50   <link rel="icon" href="../../assets/img/logopa.png" type="image/png">
51   <link rel="stylesheet" href="../../assets/css/normalize.css">
52   <link rel="stylesheet" href="../../assets/css/sweetalert2.css">
53   <link rel="stylesheet" href="../../assets/css/material.min.css">
54   <link rel="stylesheet" href="../../assets/css/material-design-iconic-font.min.css">
55   <link rel="stylesheet" href="../../assets/css/jquery.mCustomScrollbar.css">
56   <link rel="stylesheet" href="../../assets/css/main.css">
57   <link rel="stylesheet" href="../../assets/css/simple-datatables.min.css">
58   <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
59   <script>
60     window.jQuery || document.write("<script src='../../assets/js/jquery-1.11.2.min.js'></script>")
61   </script>
62   <script src="../../assets/js/material.min.js"></script>
63   <script src="../../assets/js/sweetalert2.min.js"></script>
64   <script src="../../assets/js/jquery.mCustomScrollbar.concat.min.js"></script>
65   <script src="../../assets/js/marco2.js"></script>
66   <script src="../../assets/js/simple-datatables.js"></script>
67   <!-- links para exportar a excel -->
68   <script src="https://unpkg.com/xlsx@0.16.9/dist/xlsx.full.min.js"></script>
69   <script src="https://unpkg.com/file-saver@latest/FileSaver.min.js"></script>
70   <script src="https://unpkg.com/tableexport@latest/dist/js/tableexport.min.js"></script>
71 </head>
72
73
```

Figura. 184 Estructura de lenguaje HTML e Idioma del archivo `compras.php`. Fuente. Elaboración propia.

Posteriormente se coloca una etiqueta `nav` para proporcionar una navegación estructurada en la página, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta `assets/css`, así como se muestra en las figuras 185, 186 y 187.

```
compras.php
91
92 </figure>
93 <nav class="full-width">
94   <ul class="full-width list-unstyled menu-principal">
95     <li class="full-width">
96       <a href=".." class="full-width">
97         <div class="navlateral-body-cl">
98           <i class="zmdi zmdi-view-dashboard"></i>
99         </div>
100         INICIO
101       </a>
102     </li>
103   </ul>
104   <li class="full-width divider-menu-h"></li>
105   <li class="full-width">
106     <a href="#" class="full-width btn-subMenu">
107       <div class="navlateral-body-cl">
108         <i class="zmdi zmdi-case"></i>
109       </div>
110       <div class="navlateral-body-cr">
111         ADMINISTRACION
112       </div>
113     </a>
114     <span class="zmdi zmdi-chevron-left"></span>
115     <ul class="full-width menu-principal sub-menu-options">
116       <li class="full-width">
117         <a href=".." class="full-width">
118           <div class="navlateral-body-cl">
119             <i class="zmdi zmdi-truck"></i>
120           </div>
121           <div class="navlateral-body-cr">
122             PROVEEDORES
123           </div>
124         </a>
125       </li>
126       <li class="full-width">
127         <a href=".." class="full-width">
128           <div class="navlateral-body-cl">
129             <i class="zmdi zmdi-card"></i>
130           </div>
131           <div class="navlateral-body-cr">
132             UBICACIONES
133           </div>
134         </a>
135       </li>
136     </ul>
137   </li>
138 </ul>
139 </nav>
```

Figura. 185 Configuración de la barra de navegación lateral pt1. Fuente. Elaboración propia.

```
compras.php
140
141 </div>
142 <div class="navlateral-body-cr">
143   CATEGORIAS
144 </div>
145 </li>
146 <li class="full-width">
147   <a href=".." class="full-width">
148     <div class="navlateral-body-cl">
149       <i class="zmdi zmdi-case"></i>
150     </div>
151     <div class="navlateral-body-cr">
152       PROYECTOS
153     </div>
154   </a>
155 </li>
156 </ul>
157 </li>
158 <li class="full-width divider-menu-h"></li>
159 <li class="full-width">
160   <a href="#" class="full-width btn-subMenu">
161     <div class="navlateral-body-cl">
162       <i class="zmdi zmdi-face"></i>
163     </div>
164     <div class="navlateral-body-cr">
165       USUARIOS
166     </div>
167   </a>
168   <span class="zmdi zmdi-chevron-left"></span>
169   <ul class="full-width menu-principal sub-menu-options">
170     <li class="full-width">
171       <a href=".." class="full-width">
172         <div class="navlateral-body-cl">
173           <i class="zmdi zmdi-account"></i>
174         </div>
175         <div class="navlateral-body-cr">
176           ADMINISTRADORES
177         </div>
178       </a>
179     </li>
180     <li class="full-width">
181       <a href=".." class="full-width">
182         <div class="navlateral-body-cl">
183           <i class="zmdi zmdi-account"></i>
184         </div>
185         <div class="navlateral-body-cr">
186           PERSONAL
187         </div>
188       </a>
189     </li>
190   </ul>
191   </li>
192 </ul>
193 </li>
194 </ul>
195 </div>
```

Figura. 186 Configuración de la barra de navegación lateral pt2. Fuente. Elaboración propia.

```

198 <div class="navLateral-body-cr">
199     COMPRAS
200 </div>
201 </a>
202 </li>
203 <li class="full-width divider-menu-h"></li>
204 <li class="full-width">
205     <a href=".."./productos/productos.php" class="full-width">
206         <div class="navLateral-body-cl">
207             <i class="zmdi zmdi-washing-machine"></i>
208         </div>
209         <div class="navLateral-body-cr">
210             ENTRADAS
211         </div>
212 </a>
213 </li>
214 <li class="full-width divider-menu-h"></li>
215 <li class="full-width">
216     <a href=".."./salidas/salidas.php" class="full-width">
217         <div class="navLateral-body-cl">
218             <i class="zmdi zmdi-view-agenda"></i>
219         </div>
220         <div class="navLateral-body-cr">
221             SALIDAS
222         </div>
223 </a>
224 </li>
225 <li class="full-width divider-menu-h"></li>
226 <li class="full-width">
227     <a href=".."./inventario/inventario.php" class="full-width">
228         <div class="navLateral-body-cl">
229             <i class="zmdi zmdi-store"></i>
230         </div>
231         <div class="navLateral-body-cr">
232             INVENTARIO
233         </div>
234 </a>
235 </li>
236 </ul>
237 </nav>
238 </div>
239 </section>

```

Figura. 187 Configuración de la barra de navegación lateral pt3. Fuente. Elaboración propia.

Por parte del contenido de la página web se coloca una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, como se observa en la figura 188, la cual contendrá contenedores para aplicar estilos con css, además de continuar con las animaciones de la barra de navegación, se presenta la configuración para ocultar y mostrar la barra de navegación. Se programa la interfaz de cierre de sesión mediante la implementación de un botón con identificador. Posteriormente se coloca una imagen con el avatar de identificación del administrador y se manda a llamar los datos propios de este, desde la base de datos Firebase.

```

240 <!-- contenido de pagina -->
241 <section class="full-width pageContent">
242 <!-- navBar -->
243 <div class="full-width navBar">
244     <div class="full-width navBar-options">
245         <i class="zmdi zmdi-swap btn-menu" id="btn-menu"></i>
246         <div class="mdl-tooltip" for="btn-menu">Ocultar/Mostar Menu</div>
247         <nav class="navBar-options-list">
248             <ul class="list-unstyled">
249                 <li class="btn-exit" id="btn-exit">
250                     <i class="zmdi zmdi-power"></i>
251                     <div class="mdl-tooltip" for="btn-exit">Cerrar Sesión</div>
252                 </li>
253                 <li class="text-condensed light noLink"><small><small><?php echo $_SESSION['usuario']['NombreCompleto']></small></small></li>
254                 <li class="noLink"><small><small><?php echo $_SESSION['usuario']['Apellido'];></small></small></li>
255                 <li class="noLink">
256                     <figure>
257                         
258                     </figure>
259                 </li>
260             </ul>
261         </nav>
262     </div>

```

Figura. 188 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Las animaciones correspondientes al cierre de sesión y ocultar/mostrar la barra de navegación son estructuradas en el archivo “marco.js” ubicado en la carpeta assets/js, así como se muestra en la figura 189.

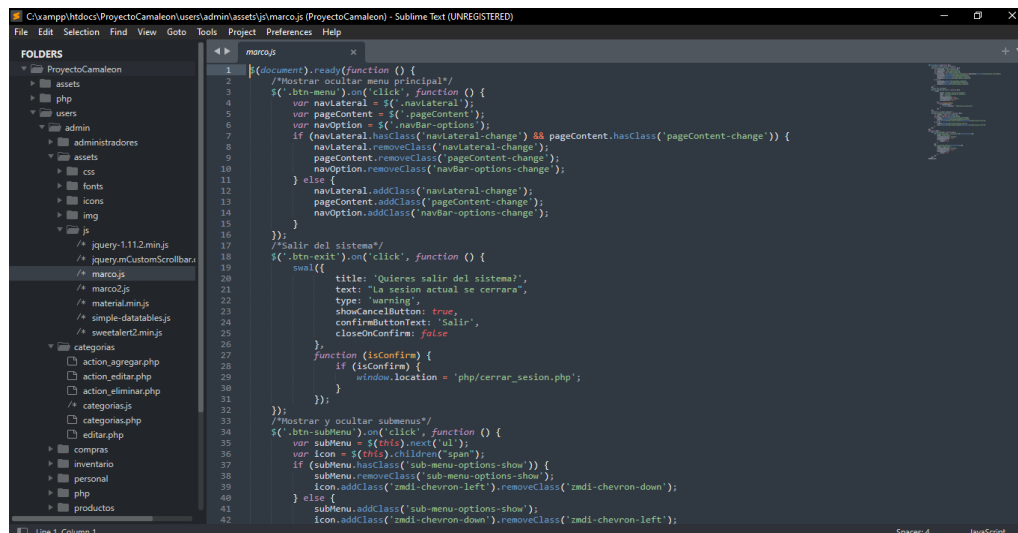


Figura. 189 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Posteriormente como se observa en la figura 190, se coloca una etiqueta de section con sus respectivos contenedores para dar formato a la descripción del formulario.

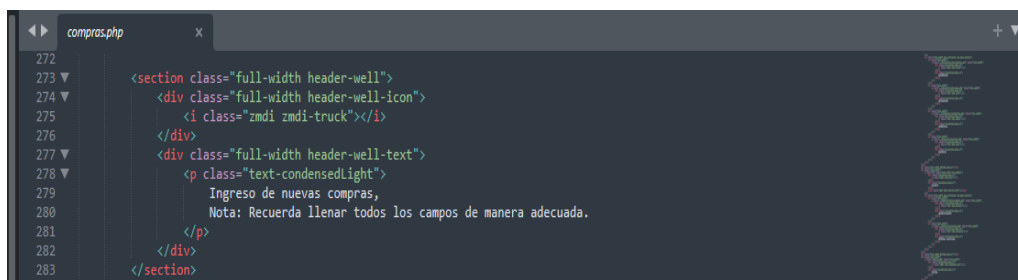


Figura. 190 Creación de contenedor de indicaciones del formulario. Fuente. Elaboración propia.

A su vez como se muestra en la figura 191, se coloca un nuevo contenedor el cual contendrá dos enlaces que permitirán navegar dentro de la misma página, uno para visualizar el contenido de los registros del formulario en una tabla y el segundo que muestra el formulario a ser completado.

```

    </div>
  </section>
  <div class="mdl-tabs mdl-js-tabs mdl-js-ripple-effect">
    <div class="mdl-tabs__tab-bar">
      <a href="#tabNewAdmin" class="mdl-tabs__tab is-active">NUEVO</a>
      <a href="#tabListAdmin" class="mdl-tabs__tab">LISTA</a>
    </div>
    <div class="mdl-tabs__panel is-active" id="tabNewAdmin">
      <div class="mdl-grid">
        <div class="mdl-cell mdl-cell--12-col">
          <div class="full-width panel mdl-shadow--2dp">
            <div class="full-width panel-title bg-primary text-center titles">

```

Figura. 191 Creación de enlaces navegadores del archivo compras.php. Fuente. Elaboración propia.

Por un lado, como se muestra en la figura 192, una vez colocados los enlaces se crea un contenedor donde se define el título del formulario y se expresa en texto los datos que deberán ser llenados a continuación.

```

283 </section>
284 <div class="mdl-tabs mdl-js-tabs mdl-js-ripple-effect">
285   <div class="mdl-tabs__tab-bar">
286     <a href="#tabNewProvider" class="mdl-tabs__tab is-active">NUEVO</a>
287     <a href="#tabListProvider" class="mdl-tabs__tab">LISTA</a>
288   </div>
289   <div class="mdl-tabs__panel is-active" id="tabNewProvider">
290     <div class="mdl-grid">
291       <div class="mdl-cell mdl-cell--12-col">
292         <div class="full-width panel mdl-shadow--2dp">
293           <div class="full-width panel-title bg-primary text-center titles">
294             Nueva Compra
295           </div>
296           <div class="full-width panel-content">
297             <form action="action_agregar.php" method="POST" id="form">
298               <div class="mdl-grid">
299                 <div class="mdl-cell mdl-cell--12-col">
300                   <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp; INFORMACION BASICA DE LA COMPRA</legend><br>
301

```

Figura. 192 Creación de título del formulario e indicaciones. Fuente. Elaboración propia.

Así mismo como se observa en la figura 193, la creación de nuevos contenedores que almacenaran los inputs necesarios para el registro de información.

```

301 </div>
302 <div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
303   <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
304     <input class="mdl-textfield__input" type="number" pattern="[0-9-
305       ]*(\.[0-9]+)?" name="Id">
306     <label class="mdl-textfield__label" for="Id">Id de la Compra</label>
307     <span class="mdl-textfield__error">Id Invalido</span>
308   </div>
309 </div>
310 <div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
311   <div class="mdl-textfield mdl-js-textfield">
312     <input type="date" class="mdl-textfield__input" name="Fecha_de_Compra">
313   </div>
314 </div>
315 <div class="mdl-cell mdl-cell--12-col">
316   <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp; USUARIO</legend><br>
317 </div>
318 <div class="mdl-cell mdl-cell--12-col">
319   <div class="mdl-textfield mdl-js-textfield">
320     <select class="mdl-textfield__input" id="usuario_select" name="Usuario">
321       <option value="">Selecciona Usuario</option>
322     </select>
323   </div>
324 </div>
325 <div class="mdl-cell mdl-cell--12-col">
326   <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp; PRODUCTOS</legend><br>
327 </div>
328 <div class="mdl-cell mdl-cell--12-col">
329   <div class="mdl-textfield mdl-js-textfield">
330     <select class="mdl-textfield__input" id="producto_select" name="Producto">
331       <option value="">Selecciona el Producto</option>
332     </select>
333   </div>
334 </div>
335 <div style="display: none;">
336   <input class="mdl-textfield__input" name="input_key" id="input_key">
337   <input class="mdl-textfield__input" name="cantidad_actual" id="cantidad_actual">
338 </div>
339 <div class="mdl-cell mdl-cell--12-col">
    <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp;

```

Figura. 193 Creación de los campos para el formulario de compras. Fuente. Elaboración propia.

La estructura de cada input se definirá por el tipo de dato a almacenar, ya sea numérico, texto, correo o contraseña. Se definen con la expresión regular de pattern para indicar el tipo de simbología que deberá de contener, esto no aplica para los campos de correo y contraseña. En caso de incumplir dicha estructura se colocará abajo del input una alerta de texto erróneo, así como se muestra en la figura 194.

```
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
  <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
    <input class="mdl-textfield__input" type="text" pattern="-?[A-Za-záéíóúÁÉÍÓÚ
      ]*(\.[0-9]+)?" name="Nombrecompleto">
    <label class="mdl-textfield__label" for="Nombrecompleto">Nombre(s)</label>
    <span class="mdl-textfield__error">Nombre invalido</span>
  </div>
</div>
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
```

Figura. 194 Codificación de la estructura de los campos para el formulario. Fuente. Elaboración propia.

Posteriormente como se muestra en la figura 195, se coloca dentro de un contenedor una lista de selección, esta puede contener información predeterminada dentro de las opciones o hacer uso de un js el cual llamara a información dentro de la base de datos para ser mostrada dentro de las opciones de la lista.

```
348 <div class="mdl-cell mdl-cell--12-col">
349   <legend class="text-condensedlight"><i class="zmdi zmdi-border-color"></i> &nbsp;MEDIDA</legend><br>
350 </div>
351 <div class="mdl-cell mdl-cell--12-col">
352   <div class="mdl-textfield mdl-js-textfield">
353     <select class="mdl-textfield__input" name="Medida">
354       <option value="">Selecciona la Unidad</option>
355       <option value="Minilitros">Mini Litros</option>
356       <option value="Gramos">Gramos</option>
357       <option value="Gramos">Piezas</option>
358     </select>
359   </div>
360 </div>
361 <div class="mdl-cell mdl-cell--12-col">
```

Figura. 195 Codificación de la estructura de listas de selección. Fuente. Elaboración propia.

Como se observa en la figura 196, al finalizar las inserciones de los input se crea un botón dentro de un contenedor que comenzara el procesamiento de información del formulario.

```
396 <div class="text-center">
397   <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
398     mdl-button--colored bg-primary" id="btn-compra">
399     <i class="zmdi zmdi-plus"></i>
400   </button>
401   <div class="mdl-tooltip" for="btn-compra">Agregar Producto</div>
402 </div>
403 </div>
404 </div>
405 </div>
406 </div>
407 </div>
408
```

Figura. 196 Creación del botón agregar producto. Fuente. Elaboración propia.

No obstante, para que el procesamiento de datos este completo el uso del archivo “action_agregar.php” cumple con la función de verificar el tiempo de inactividad dentro de la página y almacenar los datos obtenidos de los inputs en variables. Como se muestra en la figura 197, las cuales serán evaluadas mediante sentencias condicionales identificando si la variable recibió información o esta es nula, si a información obtenida resulta ser nula se regresará al archivo “compras.php”.



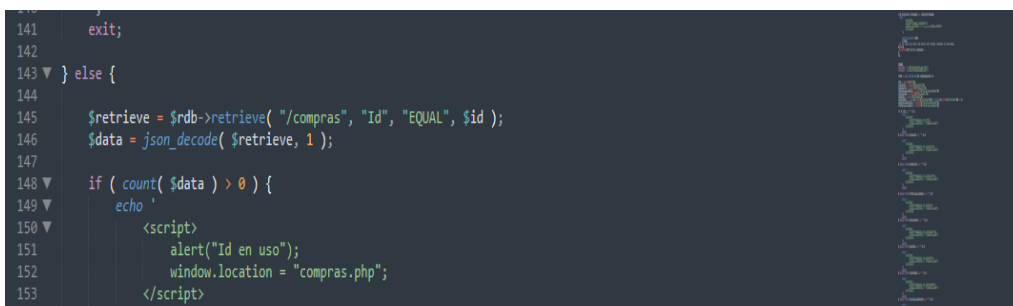
```

1  <?php
2  session_start();
3  if(!isset($_SESSION['usuario'])){// Esta logeado?.
4      echo '
5          <script>
6              alert("Inicie session");
7              window.location = "../index.html";
8          </script>
9      ';
10     session_destroy();
11     die();
12 }
13 // La variable $_SESSION['usuario'] es un ejemplo.
14 //Verifico el tiempo si este seteado, caso contrario lo seteo.
15 if(isset($_SESSION['time'])){
16     $tiempo = $_SESSION['time'];
17 }else{
18     $tiempo = strtotime(date("Y-m-d H:i:s"));
19 }
20 $inactividad =300; //Exprecion en segundos.
21 $actual = strtotime(date("Y-m-d H:i:s"));
22 if( ($actual-$tiempo) >= $inactividad){
23     echo '
24         <script>
25             alert("Tiempo excedido");
26             window.location = "../index.html";
27         </script>
28     ';
29     session_destroy();
30     die();
31     // En caso que este sea mayor del tiempo seteado lo deslogea.
32 }else{
33     $_SESSION['time'] = $actual;
34 }
35 ?>

```

Figura. 197 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente. Elaboración propia.

Por lo tanto, si la información es correcta se mandará a llamar a la tabla de categorías, así como se muestra en la figura 198.



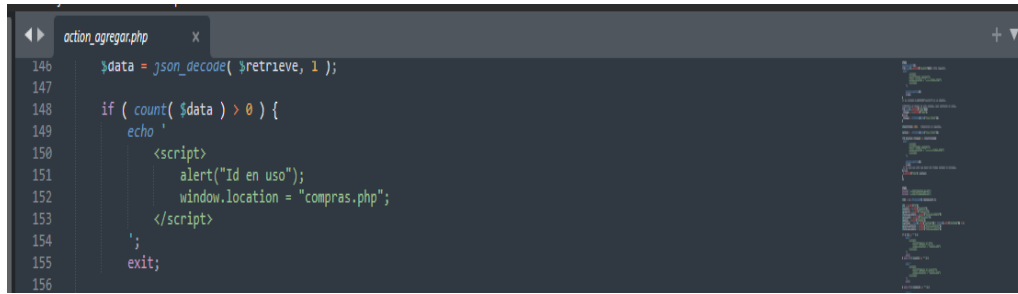
```

141     exit;
142 } else {
143     $retrieve = $rdb->retrieve( "/compras", "Id", "EQUAL", $id );
144     $data = json_decode( $retrieve, 1 );
145     if ( count( $data ) > 0 ) {
146         echo '
147             <script>
148                 alert("Id en uso");
149                 window.location = "compras.php";
150             </script>
151         ';
152     }
153 }

```

Figura. 198 Codificación para el llamado de tablas en Firebase. Fuente. Elaboración propia.

Por consiguiente, como se muestra en la figura 199, si la información es correcta se verificará por medio de sentencias de decisión si es que los datos proporcionados no están almacenados dentro de las tablas de la base de datos, si están almacenados por medio de un script regresara una alerta y redirigirá al archivo compras.php.



```

146 $data = json_decode( $retrieve, 1 );
147
148 if ( count( $data ) > 0 ) {
149     echo '
150         <script>
151             alert("Id en uso");
152             window.location = "compras.php";
153         </script>
154     ';
155     exit;
156 }
    
```

Figura. 199 Codificación de verificación para datos repetidos de los campos. Fuente. Elaboración propia.

Por otro lado, en caso de no estar almacenados, se insertará por medio de una variable que almacenara los datos correctos, a su vez actualizara la cantidad de productos que se ingrese mediante la creación de 2 variables que almacenara y actualizaran las cantidades, posteriormente se devolverá al archivo compras.php, así como se muestra en la figura 200.



```

$insert = $rdb->insert( "compras", [
    "Id" => $id,
    "Usuario" => $usuario,
    "Producto" => $producto,
    "Fecha_de_Compra" => $fecha_de_Compra,
    "Proveedor" => $proveedor,
    "Medida" => $medida,
    "Cantidad" => $cantidad,
    "Precio_Unitario" => $precio_Unitario,
    "Total_de_Compra" => $total_de_Compra,
] );

$idproduct = $_POST["input_key"];
$realcount = isset($_POST["cantidad_actual"]) ? intval($_POST["cantidad_actual"]) : 0;

$cantidadnueva = $realcount + $cantidad;

$update = $rdb->update( "productos", $idproduct, [
    "CantidadDinamica" => $cantidadnueva,
] );

if ( $insert || $update ) {
    header( "Location: compras.php" );
} else {
    echo $rdb->getLastError();
}
    
```

Figura. 200 Actualización de cantidades e ingreso de datos a las tablas. Fuente. Elaboración propia.

Una vez realizado el ingreso de información se coloca un contenedor el cual almacenara el título de la lista de registros y se da formato a la tabla de registros con los campos acorde a los inputs y registros de la base de datos, como se muestra en la figura 201, se asignan identificadores para dar formato mediante css y hacer uso de Java Script.


```

405 </div>
406 </div>
407 </div>
408
409 <div class="mdl-tabs_panel" id="tabListProvider">
410 <div class="mdl-grid">
411 <div class="mdl-cell mdl-cell--12-col">
412 <div class="full-width panel mdl-shadow--2dp">
413 <div class="full-width panel-title bg-success text-center titles">
414 Lista de compras
415 </div>
416 <div class="full-width panel-content">
417 <div class="table-responsive text-center">
418 <table class="table table-bordered table-hover" id="datatablesSimple">
419 <thead>
420 <tr>
421 <th scope="col"><strong>ID</strong></th>
422 <th scope="col"><strong>Usuario</strong></th>
423 <th scope="col"><strong>Producto</strong></th>
424 <th scope="col"><strong>Unidad de Medida</strong></th>
425 <th scope="col"><strong>Cantidad</strong></th>
426 <th scope="col"><strong>Precio Unitario</strong></th>
427 <th scope="col"><strong>Fecha de Compra</strong></th>
428 <th scope="col"><strong>Proveedor</strong></th>
429 <th scope="col"><strong>Total de Compra</strong></th>
430 </tr>
431 </thead>
432 <tbody id="tr_compras">
433 </tbody>
434 </table>
435 <button id="btnExportar" class="btn btn-success">
436 <i class="fas fa-file-excel"></i> Exportar datos a Excel
437 </button>
438 </div>
439 </div>
440 </div>
441 </div>
442 </div>
443 </div>
444 </div>
445 </div>
446 </section>

```

Figura. 201 Creación de la tabla de registros para compras. Fuente. Elaboración propia.

Para llenar la tabla de los registros se ocupa el archivo “compras.js” el cual importa la configuración de conexión de firebase a la base de datos y manda a llamar a los campos de los registros de cada insert para almacenarlos y mostrarlos en el orden indicado con anterioridad, así como se muestra en la figura 202.

```

38 let isLoadingProveedor = false;
39 let isLoadingUsuario = false;
40
41 function CargarDatos() {
42   const usuariosRef = ref(database, "usuarios");
43   const comprasRef = ref(database, "compras");
44   const productosRef = ref(database, "productos");
45   const proveedoresRef = ref(database, "proveedores");
46
47   get(comprasRef).then(snapshot => {
48     const compras = snapshot.val();
49
50     for (const key in compras) {
51       if (Object.hasOwnProperty.call(compras, key)) {
52         const compra = compras[key];
53
54         const comprasTable = document.createElement("tr");
55         comprasTable.innerHTML = `
56 <tr>
57 <td>${compra.Id}</td>
58 <td>${compra.Usuario}</td>
59 <td>${compra.Producto}</td>
60 <td>${compra.Unidad}</td>
61 <td>${compra.Cantidad}</td>
62 <td>${compra.Precio_Unitario}</td>
63 <td>${compra.Fecha_de_Compra}</td>
64 <td>${compra.Proveedor}</td>
65 <td>${compra.Total_de_Compra}</td>
66 </tr>
67 `;
68
69         comprasTr.appendChild(comprasTable);
70       }
71     }
72
73     const dataTable = new simpleDatatables.DataTable("#datatablesSimple");
74   }).catch(error => {
75     console.error('Error obteniendo datos:', error);
76   });
77
78   get(productosRef).then(snapshot => {
79     const productos = snapshot.val();

```

Figura. 202 Codificación de llenado de la tabla para la tabla compras. Fuente. Elaboración propia.

Dentro del archivo compras.php, como se muestra en la figura 203, se coloca un scrip el cual contendrá las funcionalidades para calcular de manera automática el precio con IVA incluido de las cantidades que se ingresen.

```

448 <script>
449   const cantidad = document.getElementById("Cant");
450   const precio_unitario = document.getElementById("PrecUni");
451   const total = document.getElementById("Total");
452
453   function calcularTotal() {
454
455     const cantidadValor = parseFloat(cantidad.value) || 0;
456     const precioValor = parseFloat(precio_unitario.value) || 0;
457
458     const cantidad_precio = cantidadValor * precioValor;
459     const iva = cantidad_precio * 0.16;
460     const total_final = cantidad_precio + iva;
461
462     total.value = isNaN(total_final) ? '' : total_final.toFixed(2);
463   }
464
465   cantidad.addEventListener("input", calcularTotal);
466   precio_unitario.addEventListener("input", calcularTotal);
467
468 </script>

```

Figura. 203 Codificación del cálculo del precio con IVA. Fuente. Elaboración propia.

De ahí que como se muestra en la figura 204, dentro de la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, mediante el uso de un script colocado posteriormente del llamado “compras.js”.

```

469 <script type="module" src="compras.js"></script>
470 <!-- script para exportar a excel -->
471 <script>
472   const $btnExportar = document.querySelector("#btnExportar"),
473   $datatablesSimple = document.querySelector("#datatablesSimple");
474
475   $btnExportar.addEventListener("click", function() {
476     let tableExport = new TableExport($datatablesSimple, {
477       exportButtons: false, // No queremos botones
478       filename: "Reporte de Compras", //Nombre del archivo de Excel
479       sheetname: "Reporte de Compras", //Título de la hoja
480     });
481     let datos = tableExport.getExportData();
482     let preferenciasDocumento = datos.datatablesSimple.xlsx;
483     tableExport.export2file(preferenciasDocumento.data, preferenciasDocumento.mimeType, preferenciasDocumento.filename,
484                           preferenciasDocumento.fileExtension, preferenciasDocumento.merges, preferenciasDocumento.RTL,
485                           preferenciasDocumento.sheetname);
486   });
487 </script>
488 </body>
489 </html>

```

Figura. 204 Codificación de descarga y conversión a formato Excel. Fuente. Elaboración propia.

De ahí que, dentro de la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, mediante el uso de un script colocado posteriormente del llamado “compras.js”.

3.10.7. Inventario:

Por parte del archivo “inventario.php” se especifica en lenguaje de php mediante sentencias de decisión donde se evalúa si una vez en la página de home.php se autentico de manera correcta, en caso de no ser así se coloca una alerta y retornara a index.html. A si como se observa en la figura 205, se evalúa el tiempo de inactividad entre navegación de archivos teniendo un conteo de 300 segundos de inactividad. Si el conteo es completado se retornará a index.html de manera forzada, en caso de navegar entre archivos el conteo se reiniciará de manera automática.

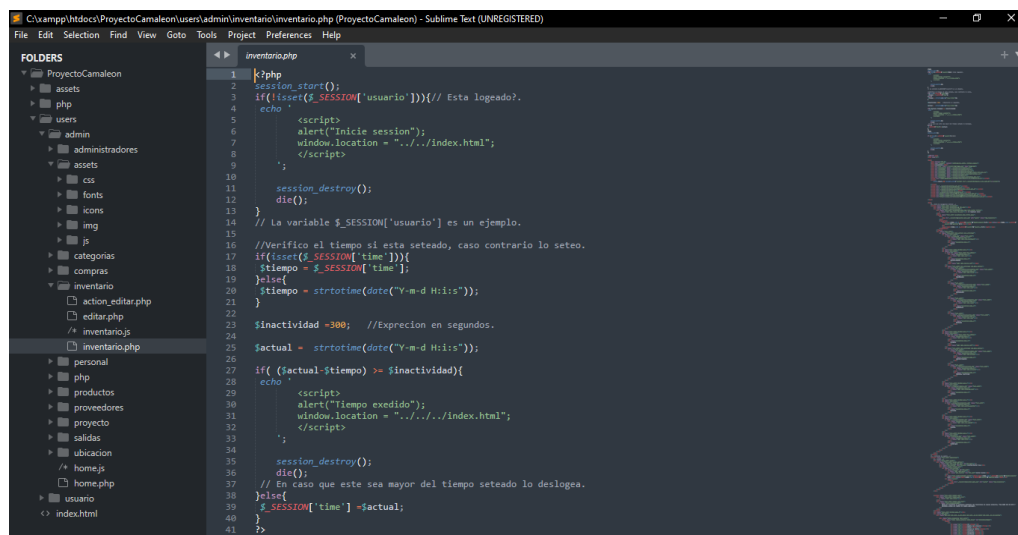


Figura. 205 Configuración de lenguaje y tiempo de inactividad del archivo inventario.php, Fuente. Elaboración propia.

Posteriormente como se muestra en la figura 206, se coloca la estructura de !DOCTYPE html> la cual informara al navegador el tipo de versión HTML que fue implementado en el archivo, a su vez se indica con una etiqueta de tipo lang el tipo de idioma con el que será escrita la página web en este caso se hace uso del idioma español. A su vez se muestra la apertura de un head en el cual se proveerá la información como lo es el título y lo enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assents” y de las subcarpetas de “img”, “css” y “js”.

```

57
58 <!DOCTYPE html>
59 <html lang="es">
60
61 <head>
62   <meta charset="UTF-8">
63   <meta name="viewport" content="width=device-width, initial-scale=1">
64   <title>Inventario</title>
65   <link rel="icon" href="../../assets/img/logopa.png" type="image/png">
66   <link rel="stylesheet" href="../../assets/css/normalize.css">
67   <link rel="stylesheet" href="../../assets/css/sweetalert2.min.css">
68   <link rel="stylesheet" href="../../assets/css/material.min.css">
69   <link rel="stylesheet" href="../../assets/css/material-design-iconic-font.min.css">
70   <link rel="stylesheet" href="../../assets/css/jquery.mCustomScrollbar.css">
71   <link rel="stylesheet" href="../../assets/css/main.css">
72   <link rel="stylesheet" href="../../assets/css/simple-datatables.min.css">
73   <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
74   <script>
75     window.jQuery || document.write('<script src="../../assets/js/jquery-1.11.2.min.js"></script>')
76   </script>
77   <script src="../../assets/js/material.min.js"></script>
78   <script src="../../assets/js/sweetalert2.min.js"></script>
79   <script src="../../assets/js/jquery.mCustomScrollbar.concat.min.js"></script>
80   <script src="../../assets/js/marco2.js"></script>
81   <script src="../../assets/js/simple-datatables.js"></script>
82   <script src="https://unpkg.com/xlsx@0.16.9/dist/xlsx.full.min.js"></script>
83   <script src="https://unpkg.com/file-saver@latest/FileSaver.min.js"></script>
84   <script src="https://unpkg.com/tableexport@latest/dist/js/tableexport.min.js"></script>
85
86 </head>
87
88

```

Figura. 206 Estructura de lenguaje HTML e Idioma del archivo invntario.php. Fuente. Elaboración propia.

Una vez definido el encabezado de la página mediante la etiqueta head, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web. Posteriormente se coloca un elemento que ayudara a representar el contenido principal de la página web que es el “body”. Posteriormente se abre una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, se colocan contenedores para aplicar estilos correspondientes a las sentencias, se agregan imágenes que harán la función del avatar del administrador, así como el nombre del usuario administrador que se autentico con anterioridad los datos son llamados desde la base de datos de Firebase, así como se muestra en la figura 207.

```

89 <body>
90   <!-- barra de navegacion lateral -->
91   <section class="full-width navLateral">
92     <div class="full-width navLateral-bg btn-menu"></div>
93     <div class="full-width navLateral-body">
94       <div class="full-width navLateral-body-logo text-center tittles">
95         <i class="zmdi zmdi-close btn-menu"></i> CC Camaleon Color
96       </div>
97       <figure class="full-width navLateral-body-tittle-menu">
98         <div>
99           
100         </div>
101         <figcaption>
102           <span><?php echo $_SESSION['usuario']['NombreCompleto'];></span>&nbsp;<span><?php echo $_SESSION['
103             usuario']['Apellido'];></span></span>
104           <br>
105           <span><small><?php echo $_SESSION['usuario']['Usuario_a'];></small></span>
106         </figcaption>
107       </figure>

```

Figura. 207 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.

Se coloca una etiqueta nav para proporcionar una navegación estructurada en la página, así como se muestra en las figuras 208,209,210 y 211, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta assets/css.

```

107 <nav class="full-width">
108 <ul class="full-width list-unstyled menu-principal">
109 <li class="full-width">
110 <a href=".."home.php" class="full-width">
111 <div class="navLateral-body-cl">
112 <i class="zmdi zmdi-view-dashboard"></i>
113 </div>
114 <div class="navLateral-body-cr">
115 INICIO
116 </div>
117 </a>
118 </li>
119 <li class="full-width divider-menu-h"></li>
120 <li class="full-width">
121 <a href=".."proveedores/proveedores.php" class="full-width btn-subMenu">
122 <div class="navLateral-body-cl">
123 <i class="zmdi zmdi-truck"></i>
124 </div>
125 <div class="navLateral-body-cr">
126 ADMINISTRACION
127 </div>
128 <span class="zmdi zmdi-chevron-left"></span>
129 </a>
130 </li>
131 <li class="full-width menu-principal sub-menu-options">
132 <a href=".."categorias/categorias.php" class="full-width">
133 <div class="navLateral-body-cl">
134 <i class="zmdi zmdi-truck"></i>
135 </div>
136 <div class="navLateral-body-cr">
137 PROVEEDORES
138 </div>
139 </a>
140 </li>

```

Figura. 208 Configuración de la barra de navegación lateral pt1. Fuente. Elaboración propia.

```

150 </li>
151 <li class="full-width">
152 <a href=".."proyectos/proyectos.php" class="full-width">
153 <div class="navLateral-body-cl">
154 <i class="zmdi zmdi-label"></i>
155 </div>
156 <div class="navLateral-body-cr">
157 CATEGORIAS
158 </div>
159 </a>
160 </li>
161 <li class="full-width">
162 <a href=".."proyecto/proyectos.php" class="full-width">
163 <div class="navLateral-body-cl">
164 <i class="zmdi zmdi-case"></i>
165 </div>
166 <div class="navLateral-body-cr">
167 ENTRADAS
168 </div>
169 </a>
170 </li>
171 </ul>
172 </nav>
173 <li class="full-width divider-menu-h"></li>
174 <li class="full-width">
175 <a href="#" class="full-width btn-subMenu">
176 <div class="navLateral-body-cl">
177 <i class="zmdi zmdi-face"></i>
178 </div>
179 <div class="navLateral-body-cr">
180 USUARIOS
181 </div>
182 <span class="zmdi zmdi-chevron-left"></span>
183 </a>
184 </li>
185 <li class="full-width menu-principal sub-menu-options">
186 <a href=".."administradores/admin.php" class="full-width">
187 <div class="navLateral-body-cl">
188 <i class="zmdi zmdi-account"></i>
189 </div>
190 <div class="navLateral-body-cr">
191 ADMINISTRADORES

```

Figura. 209 Configuración de la barra de navegación lateral pt2. Fuente. Elaboración propia.

```

200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
categorias.php
</div>
</li>
<li class="full-width divider-menu-h"></li>
<li class="full-width">
<a href="productos/productos.php" class="full-width">
<div class="navLateral-body-cl">
<i class="zmdi zmdi-washing-machine"></i>
</div>
<div class="navLateral-body-cr">
ENTRADAS
</div>
</a>
</li>
<li class="full-width divider-menu-h"></li>
<li class="full-width">
<a href="salidas/salidas.php" class="full-width">
<div class="navLateral-body-cl">
<i class="zmdi zmdi-view-agenda"></i>
</div>
<div class="navLateral-body-cr">
SALIDAS
</div>
</a>
</li>
<li class="full-width divider-menu-h"></li>
<li class="full-width">
<a href="inventario/inventario.php" class="full-width">
<div class="navLateral-body-cl">
<i class="zmdi zmdi-store"></i>
</div>
<div class="navLateral-body-cr">
INVENTARIO
</div>
</a>
</li>
</ul>
</nav>
</div>
</section>
<!-- contenido de pagina -->
<section class="full-width napeContent">

```

Figura. 210 Configuración de la barra de navegación lateral pt3. Fuente. Elaboración propia.

```

207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
categorias.php
<li class="full-width divider-menu-h"></li>
<li class="full-width">
<a href="compras/compras.php" class="full-width">
<div class="navLateral-body-cl">
<i class="zmdi zmdi-shopping-cart"></i>
</div>
<div class="navLateral-body-cr">
COMPRAS
</div>
</a>
</li>
<li class="full-width divider-menu-h"></li>
<li class="full-width">
<a href="productos/productos.php" class="full-width">
<div class="navLateral-body-cl">
<i class="zmdi zmdi-washing-machine"></i>
</div>
<div class="navLateral-body-cr">
ENTRADAS
</div>
</a>
</li>
<li class="full-width divider-menu-h"></li>
<li class="full-width">
<a href="salidas/salidas.php" class="full-width">
<div class="navLateral-body-cl">
<i class="zmdi zmdi-view-agenda"></i>
</div>
<div class="navLateral-body-cr">
SALIDAS
</div>
</a>
</li>
</ul>
</nav>
</div>
</section>
<!-- contenido de pagina -->
<section class="full-width napeContent">

```

Figura. 211 Configuración de la barra de navegación lateral pt4. Fuente. Elaboración propia.

Por parte del contenido de la página web, como se muestra en la figura 212, se coloca una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, la cual contendrá contenedores para aplicar estilos con css, además de continuar con las animaciones de la barra de navegación, se presenta la configuración para ocultar y mostrar la barra de navegación. Se programa la interfaz de cierre de sesión mediante la implementación de un botón con identificador. Posteriormente se coloca una imagen con el avatar de identificación del administrador y se manda a llamar los datos propios de este, desde la base de datos Firebase.

```

255 <!-- contenido de pagina -->
256 <section class="full-width pageContent">
257 <!-- navbar -->
258 <div class="full-width navbar">
259 <div class="full-width navbar-options">
260 <i class="zmdi zmdi-swap btn-menu" id="btn-menu"></i>
261 <div class="mdl-tooltip" for="btn-menu">Ocultar/Mostrar Menu</div>
262 <nav class="navbar-options-list">
263 <ul class="list-unstyled">
264 <li class="btn-exit" id="btn-exit">
265 <i class="zmdi zmdi-power"></i>
266 <div class="mdl-tooltip" for="btn-exit">Cerrar Sesión</div>
267 </li>
268 <li class="text-condensedLight noLink"><small><small><?php echo $_SESSION['usuario']['NombreCompleto']></small><small><?php echo $_SESSION['usuario']['Apellido']></small></li>
269 <li class="noLink">
270 <figure>
271 
272 </figure>
273 </li>
274 </ul>
275 </nav>
276 </div>
277 </div>

```

Figura. 212 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Las animaciones correspondientes al cierre de sesión y ocultar/mostrar la barra de navegación son estructuradas en el archivo “marco.js” ubicado en la carpeta assets/js, así como se muestra en la página 213.

```

1 (document).ready(function () {
2 //Mostrar ocultar menu principal*/
3 $(document).on('click', function () {
4 var navLateral = $('#navLateral');
5 var pageContent = $('#pageContent');
6 var navOption = $('#navBar-options');
7 if (navLateral.hasClass('navLateral-change') && pageContent.hasClass('pageContent-change')) {
8 navLateral.removeClass('navLateral-change');
9 pageContent.removeClass('pageContent-change');
10 navOption.removeClass('navBar-options-change');
11 } else {
12 navLateral.addClass('navLateral-change');
13 pageContent.addClass('pageContent-change');
14 navOption.addClass('navBar-options-change');
15 }
16 });
17 /*Salir del sistema*/
18 $(document).on('click', function () {
19 swal({
20 title: 'Quieres salir del sistema?',
21 text: 'La sesión actual se cerrará',
22 type: 'warning',
23 showCancelButton: true,
24 confirmButtonText: 'Salir',
25 closeOnConfirm: false
26 },
27 function (isConfirm) {
28 if (isConfirm) {
29 window.location = 'php/cerrar_sesion.php';
30 }
31 });
32 });
33 /*Mostrar y ocultar submenús*/
34 $(document).on('click', function () {
35 var subMenu = $(this).next('ul');
36 var icon = $(this).children('span');
37 if (subMenu.hasClass('sub-menu-options-show')) {
38 subMenu.removeClass('sub-menu-options-show');
39 icon.addClass('zmdi-chevron-left').removeClass('zmdi-chevron-down');
40 } else {
41 subMenu.addClass('sub-menu-options-show');
42 icon.addClass('zmdi-chevron-down').removeClass('zmdi-chevron-left');
43 }
44 });
45 });

```

Figura. 213 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Se muestra en la figura 214, se coloca una etiqueta de section con sus respectivos contenedores para dar formato a la tabla.

```

277 </div>
278
279 <section class="full-width header-well">
280 <div class="full-width header-well-icon">
281 <i class="zmdi zmdi-store"></i>
282 </div>
283 <div class="full-width header-well-text">
284 <p class="text-condensedLight">
285 NOTA: El inventario se actualiza mediante las inserciones de nuevos productos, "HAZ BUEN USO DE ESTE Y
286 RECUERDA LLENAR LOS CAMPOS DE MANERA ADECUADA".
287 </p>
288 </div>
289 </section>

```

Figura. 214 Creación del contenedor de indicaciones de la tabla inventario. Fuente. Elaboración propia.

Posteriormente como se muestra en la figura 215, se coloca un contenedor el cual almacenara el título de la lista de registros y se da formato a la tabla de registros con los campos acorde a los inputs y registros de la base de datos, se asignan identificadores para dar formato

```

277 </div>
278
279
280 <section class="full-width header-well">
281   <div class="full-width header-well-icon">
282     <i class="zmdi zmdi-store"></i>
283   </div>
284   <div class="full-width header-well-text">
285     <p class="text-condensedlight">
286       NOTA: El inventario se actualiza mediante las inserciones de nuevos productos, "HAZ BUEN USO DE ESTE Y
287       RECUERDA LLENAR LOS CAMPOS DE MANERA ADECUADA".
288     </p>
289   </div>
290 </section>
291 <div class="full-width divider-menu-h"></div>
292 <div class="mdl-grid">
293   <div class="mdl-cell mdl-cell--4-col-phone mdl-cell--8-col-tablet mdl-cell--12-col-desktop">
294     <div class="table-responsive text-center">
295       <table class="table table-bordered table-hover" id="datatablesSimple">
296         <thead>
297           <tr>
298             <th scope="col"><strong>ID</strong></th>
299             <th scope="col"><strong>Nombre del producto</strong></th>
300             <th scope="col"><strong>Stock</strong></th>
301             <th scope="col"><strong>Unidad de Medida</strong></th>
302             <th scope="col"><strong>Proyecto</strong></th>
303             <th scope="col"><strong>Ubicacion</strong></th>
304             <th scope="col"><strong>Categoria</strong></th>
305             <th scope="col"><strong>Proveedor</strong></th>
306             <th scope="col"><strong>Acción</strong></th>
307           </tr>
308         </thead>
309         <tbody id="tr_inventario">
310           </tbody>
311         </table>
312         <button id="btnExportar" class="btn btn-success">
313           <i class="fas fa-file-excel"></i> Exportar datos a Excel
314         </button>
315       </div>
316     </div>
  
```

Figura. 215 Creación de la tabla de registros para el inventario. Fuente. Elaboración propia.

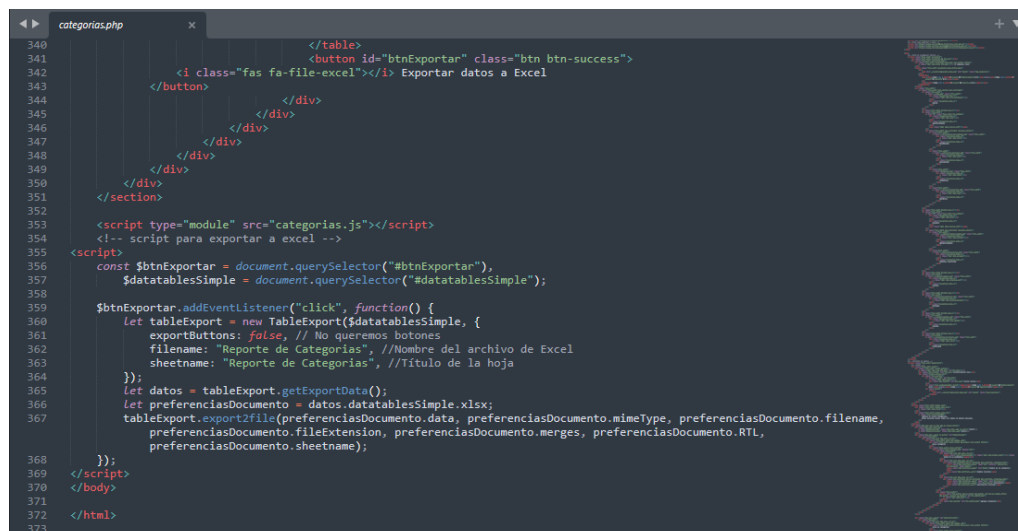
Para llenar la tabla de los registros se ocupa el archivo “inventario.js”, así como se muestra en la figura 216, el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros de cada insert para almacenarlos y mostrarlos en el orden indicado con anterioridad, a su vez se agregan las instrucciones para que muestre en color rojo si el campo “cantidad” es menor a 200 y en amarillo si es mayor a 200 pero menor a 501.

```

40 const compra = compras[keyCompra];
41
42 const relacionUnica = `${inventario.Proveedor}_${inventario.Nombre}_${compra.Proveedor}_${compra.Produ
43
44 if (!relacionesProcesadas.has(relacionUnica) && inventario.Proveedor == compra.Proveedor && inventario
45 const inventarioTable = document.createElement("tr");
46 const cantidadNumerica = parseInt(inventario.CantidadDinamica, 10);
47
48 if ((cantidadNumerica >= 201 && cantidadNumerica <= 500) && (compra.Medida == 'Minilitros' || compra
49   inventarioTable.innerHTML =
50   <tr>
51     <td><strong>${rowNumber}</strong></td>
52     <td>${inventario.Nombre}</td>
53     <td>
54       <div style="color: #daa520; background-color: yellow;">
55         ${inventario.CantidadDinamica}
56       </div>
57     </td>
58     <td>${compra.Medida}</td>
59     <td>${inventario.Proyecto}</td>
60     <td>${inventario.Ubicacion}</td>
61     <td>${inventario.Categoria}</td>
62     <td>${inventario.Proveedor}</td>
63     <td>
64       <div>
65         <a href="editar.php?id=${keyInventario}" class="editar"><i class="zmdi zmdi-edit"></i>
66         <a href="salidas/salida.php?id=${keyInventario}" class="salida"><i class="zmdi
67       </div>
68     </td>
69   </tr>
70   </tbody>
71   </table>
72   </div>
73   </div>
74   </div>
75   </div>
76   </div>
77   </div>
78   </div>
79   </div>
80   </div>
81   </div>
82   </div>
83   </div>
84   </div>
85   </div>
86   </div>
  
```

Figura. 216 Codificación y evaluación del ingreso de datos a la tabla inventario. Fuente. Elaboración propia

Como se observa en la figura 217, dentro de la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, mediante el uso de un script colocado posteriormente del llamado “inventario.js”.



```

340         </table>
341         <button id="btnExportar" class="btn btn-success">
342             <i class="fas fa-file-excel"></i> Exportar datos a Excel
343         </button>
344     </div>
345 </div>
346 </div>
347 </div>
348 </div>
349 </div>
350 </div>
351 </section>
352 </script>
353 <script type="module" src="categorias.js"></script>
354 <!-- script para exportar a excel -->
355 <script>
356     const $btnExportar = document.querySelector("#btnExportar"),
357           $datatablesSimple = document.querySelector("#datatablesSimple");
358
359     $btnExportar.addEventListener("click", function() {
360         let tableExport = new TableExport($datatablesSimple, {
361             exportButtons: false, // No queremos botones
362             filename: "Reporte de Categorías", //Nombre del archivo de Excel
363             sheetname: "Reporte de Categorías", //Titulo de la hoja
364         });
365         let datos = tableExport.getExportData();
366         let preferenciasDocumento = datos.datatablesSimple.xlsx;
367         tableExport.export2file(preferenciasDocumento.data, preferenciasDocumento.mimeType, preferenciasDocumento.filename,
368             preferenciasDocumento.fileExtension, preferenciasDocumento.merges, preferenciasDocumento.RTL,
369             preferenciasDocumento.sheetname);
370     });
371 </script>
372 </body>
373 </html>

```

Figura. 217 Codificación y conversión de registros a formato Excel. Fuente. Elaboración propia.

En relación con el archivo inventario.php se encuentra “editar.php” perteneciente a la misma carpeta, así como se muestra en la figura 218, se encuentran las configuraciones iniciales de la barra de navegación, no obstante, se hace un llamado a la conexión de la base de datos dentro de la interpretación de PHP, se especifica la tabla a la cual se tendrá acceso de información.



```

274 <?php
275 include '../php/Conexion_be.php';
276 include '../php/firebaseRDB.php';
277
278 $db = new firebaseRDB($databaseUrl);
279 $id = $_GET['id'];
280 $retrieve = $db->retrieve("productos/$id");
281 $data = json_decode($retrieve, 1);
282
283 function getProductData($db) {
284     $productData = $db->retrieve("productos");
285     return json_decode($productData, true);
286 }
287
288 $productos = getProductData($db);
289
290 ?>
291

```

Figura. 218 Creación y configuración del archivo editar.php. Fuente. Elaboración propia.

Los inputs reciben una alteración de código, como se muestra en la figura 219, permitiendo que los datos almacenados con anterioridad en los campos designados sean llamados y visualizados dentro del mismo input, se conservan las reglas de patrones indicados, los inputs son desactivados de manera que solo muestren información y no permiten el ingreso de esta a excepción del input cantidad que mostrara la cantidad actual y permitirá su alteración.

```

327 <div style="display: none;">
328 <input type="text" name="uid" value="php echo $id?&gt;"&gt;
329 &lt;/div&gt;
330 &lt;div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet"&gt;
331 &lt;div class="mdl-textfield mdl-js-textfield"&gt;
332 &lt;input type="date" class="mdl-textfield__input" name="Fecha_de_salida" id="Fecha"&gt;
333 &lt;/div&gt;
334 &lt;/div&gt;
335 &lt;div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet"&gt;
336 &lt;div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label"&gt;
337 &lt;input class="mdl-textfield__input" type="text" pattern="[A-Za-z0-9]{1,30}" name="Nombre" value="<?php echo $data['Nombre']?&gt;"&gt;
338 &lt;label class="mdl-textfield__label" for="Nombre"&gt;Nombre del Producto&lt;/label&gt;
339 &lt;span class="mdl-textfield__error"&gt;Nombre Invalido&lt;/span&gt;
340 &lt;/div&gt;
341 &lt;/div&gt;
342 &lt;div class="mdl-cell mdl-cell--4-col dis"&gt;
343 &lt;div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label"&gt;
344 &lt;input class="mdl-textfield__input" type="text" pattern="[A-Za-z0-9áéíóú]{1,30}" name="Proveedor" value="<?php echo $data['Proveedor']?&gt;"&gt;
345 &lt;label class="mdl-textfield__label" for="Proveedor"&gt;Nombre del Proveedor&lt;/label&gt;
346 &lt;span class="mdl-textfield__error"&gt;Nombre Invalido&lt;/span&gt;
347 &lt;/div&gt;
348 &lt;/div&gt;
349 &lt;div class="mdl-cell mdl-cell--4-col dis"&gt;
350 &lt;div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label"&gt;
351 &lt;input class="mdl-textfield__input" type="text" name="Ubicacion" value="<?php echo $data['Ubicacion']?&gt;"&gt;
352 &lt;label class="mdl-textfield__label" for="Ubicacion"&gt;Nombre de la Ubicacion&lt;/label&gt;
353 &lt;span class="mdl-textfield__error"&gt;Nombre Invalido&lt;/span&gt;
354 &lt;/div&gt;
355 &lt;/div&gt;
356 &lt;div class="mdl-cell mdl-cell--4-col dis"&gt;
357 &lt;div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label"&gt;
358 &lt;input class="mdl-textfield__input" type="text" name="Categoria" value="<?php echo $data['Categoria']?&gt;"&gt;
359 &lt;label class="mdl-textfield__label" for="Categoria"&gt;Nombre de la Categoria&lt;/label&gt;
</pre

```

Figura. 219 Codificación y llamado de registros en los campos del formulario. Fuente. Elaboración propia.

Se coloca un botón permitiendo la redirección al archivo “action_editar” continuando con el proceso de actualización de datos del formulario, así como se muestra en la figura 220.

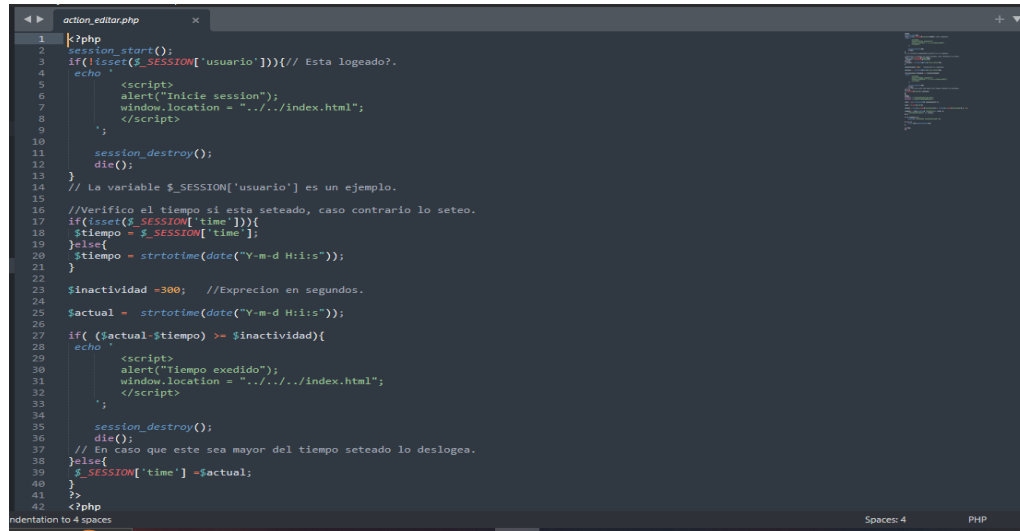
```

371 <div class="text-center">
372 <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
373 mdl-button--colored bg-primary" id="btn-addProduct">
374 <i class="zmdi zmdi-plus"></i>
375 </button>
376 <div class="mdl-tooltip" for="btn-addProduct">Agregar Producto</div>
377 </div>
378 </div>
379 </div>
380 </div>
381 </div>
382 </div>
383 </div>
384 </div>

```

Figura. 220 Creación del botón agregar producto. Fuente. Elaboración propia.

Al igual que el archivo `action_agregar` se conservan las instrucciones sobre inactividad, como se muestra en la figura 221.

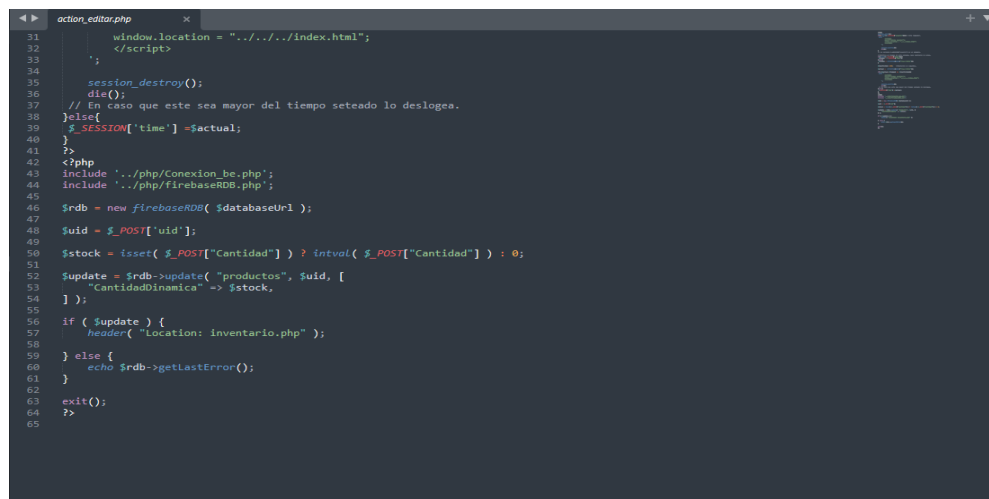


```

1  <?php
2  session_start();
3  if(!isset($_SESSION['usuario'])){// Esta logeado?.
4      echo '
5          <script>
6              alert("Inicie session");
7              window.location = "../index.html";
8          </script>
9      '
10
11      session_destroy();
12      die();
13  }
14  // La variable $_SESSION['usuario'] es un ejemplo.
15
16  //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17  if(isset($_SESSION['time'])){
18      $tiempo = $_SESSION['time'];
19  }else{
20      $tiempo = strtotime(date("Y-m-d H:i:s"));
21  }
22
23  $inactividad = 300; //Expreccion en segundos.
24  $actual = strtotime(date("Y-m-d H:i:s"));
25
26  if( ($actual-$tiempo) >= $inactividad){
27      echo '
28          <script>
29              alert("Tiempo exedido");
30              window.location = "../index.html";
31          </script>
32      '
33
34      session_destroy();
35      die();
36  }
37  // En caso que este sea mayor del tiempo seteado lo deslogea.
38  }else{
39      $_SESSION['time'] = $actual;
40  }
41  }
42  <?php
    
```

Figura. 221 Codificación de tiempo de inactividad del archivo `action_agregar.php`. Fuente. Elaboración propia.

Se hace el llamado a la conexión de la base de datos y creación de variables que almacenaran los cambios indicados en los input, a su vez actualiza el campo de cantidad colocando una alerta y re direccionamiento al archivo “`inventario.php`”, así como se muestra en la figura 222.



```

31      window.location = "../index.html";
32  </script>
33  '
34
35  session_destroy();
36  die();
37  // En caso que este sea mayor del tiempo seteado lo deslogea.
38  }else{
39      $_SESSION['time'] = $actual;
40  }
41  }
42  <?php
43  include '../php/Conexion_be.php';
44  include '../php/firebaseRDB.php';
45
46  $rdb = new firebaseRDB( $databaseUrl );
47
48  $uid = $_POST['uid'];
49
50  $stock = isset( $_POST["Cantidad"] ) ? intval( $_POST["Cantidad"] ) : 0;
51
52  $update = $rdb->update( "productos", $uid, [
53      "CantidadDinamica" => $stock,
54  ] );
55
56  if ( $update ) {
57      header( "Location: inventario.php" );
58  } else {
59      echo $rdb->getLastError();
60  }
61
62  exit();
63  }
64  }
65  }
    
```

Figura. 222 Codificación de ingreso de datos actualizados a la tabla `productos`. Fuente. Elaboración propia.

Se muestra en la figura 223, el llenado de la tabla de los registros se ocupa el archivo “categorias.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros que fueron actualizados y mostrarlos en el orden indicado con anterioridad.

```

280 <section class="full-width header-well">
281 <div class="full-width header-well-icon">
282 <i class="zmdi zmdi-store"></i>
283 </div>
284 <div class="full-width header-well-text">
285 <p class="text-condensedlight">
286 NOTA: El inventario se actualiza mediante las inserciones de nuevos productos, "HAZ BUEN USO DE ESTE Y
287 RECUERDA LLENAR LOS CAMPOS DE MANERA ADECUADA".
288 </p>
289 </div>
290 </section>
291 <div class="full-width divider-menu-h"></div>
292 <div class="mdl-grid">
293 <div class="mdl-cell mdl-cell--4-col-phone mdl-cell--8-col-tablet mdl-cell--12-col-desktop">
294 <div class="table-responsive text-center">
295 <table class="table table-bordered table-hover" id="datatablesSimple">
296 <thead>
297 <tr>
298 <th scope="col"><strong>ID</strong></th>
299 <th scope="col"><strong>Nombre del producto</strong></th>
300 <th scope="col"><strong>Stock</strong></th>
301 <th scope="col"><strong>Unidad de Medida</strong></th>
302 <th scope="col"><strong>Proyecto</strong></th>
303 <th scope="col"><strong>Ubicación</strong></th>
304 <th scope="col"><strong>Categoría</strong></th>
305 <th scope="col"><strong>Proveedor</strong></th>
306 <th scope="col"><strong>Acción</strong></th>
307 </tr>
308 </thead>
309 <tbody id="tr_inventario">
310 </tbody>
311 </table>
312 <button id="btnExportar" class="btn btn-success">
313 <i class="fas fa-file-excel"></i> Exportar datos a Excel
314 </button>
315 </div>
316 </div>
317 </div>
318 </section>
319
  
```

Figura. 223 Creación de la tabla de registros actualizados para el inventario. Fuente. Elaboración propia

No obstante, como se muestra en la figura 22

4, para que la funcionalidad de edición comience su proceso, dentro de la tabla de visualización de registros se visualizan 2 botones de acción para cada proceso correspondiente, dichos botones son ubicados en el archivo “inventario.js”.

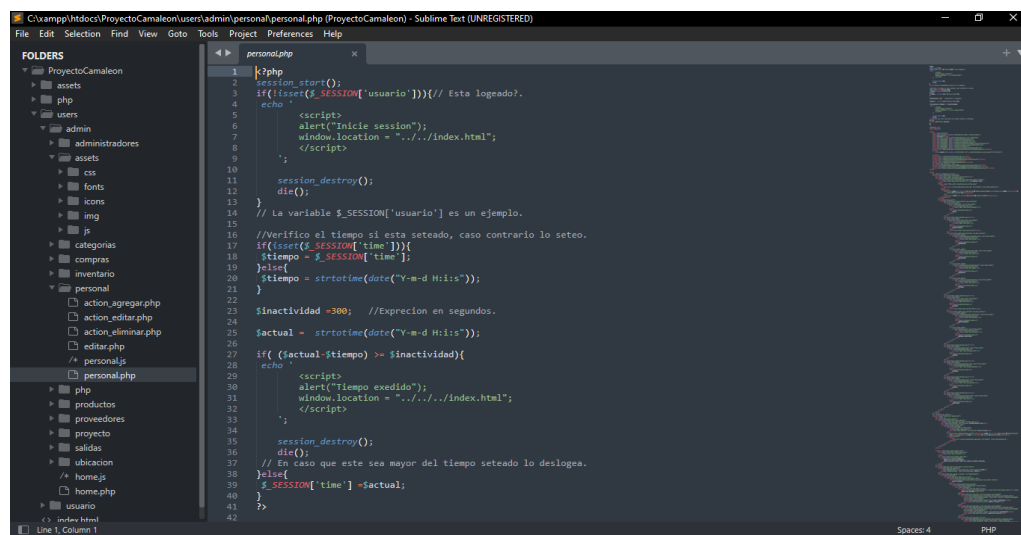
```

43 <td>{usuario.Correo_a}</td>
44 <td>{usuario.Contrasena}</td>
45 <td>
46 <div>
47 <a href="editar.php?id={key}" class="editar"><i class="zmdi zmdi-edit"></i></a>
48 <a href="action_eliminar.php?id={key}" class="eliminar"><i class="zmdi zmdi-delete"></i></a>
49 </div>
50 </td>
51 </tr>
52 </tbody>
53 </table>
54 </div>
55 </div>
56
  
```

Figura. 224 Codificación de enlaces de edición y eliminación. Fuente. Elaboración propia.

3.10.8. Personal:

Por parte del archivo “personal.php” se especifica en lenguaje de php mediante sentencias de decisión donde se evalúa si una vez en la página de home.php se autentico de manera correcta, en caso de no ser así se coloca una alerta y retornara a index.html. A si mismo evalúa el tiempo de inactividad entre navegación de archivos teniendo un conteo de 300 segundos de inactividad. Si el conteo es completado se retornará a index.html de manera forzada, en caso de navegar entre archivos el conteo se reiniciará de manera automática, Culminando con el cierre del lenguaje de php, así como se muestra en la figura 225.



```

1  <?php
2  session_start();
3  if(isset($_SESSION['usuario'])){// Esta logeado!
4      echo '
5          <script>
6              alert("Inicio session");
7              window.location = "../index.html";
8          </script>
9      '
10
11      session_destroy();
12      die();
13  }
14  // La variable $_SESSION['usuario'] es un ejemplo.
15
16  //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17  if(isset($_SESSION['time'])){
18      $tiempo = $_SESSION['time'];
19  }else{
20      $tiempo = strtotime(date("Y-m-d H:i:s"));
21  }
22
23  $inactividad =300; //Expresion en segundos.
24
25  $actual = strtotime(date("Y-m-d H:i:s"));
26
27  if( ($actual-$tiempo) >= $inactividad){
28      echo
29          <script>
30              alert("Tiempo excedido");
31              window.location = "../index.html";
32          </script>
33      '
34
35      session_destroy();
36      die();
37  }
38  // En caso que este sea mayor del tiempo setando lo desloguea.
39  }else{
40      $_SESSION['time'] = $actual;
41  }
42  >
  
```

Figura. 225 Configuración de lenguaje y tiempo de inactividad del archivo personal.php. Fuente. Elaboración propia.

Posteriormente se coloca la estructura de `<!DOCTYPE html>` la cual informara al navegador el tipo de versión HTML que fue implementado en el archivo, como se muestra en la figura 226, a su vez se indica con una etiqueta de tipo lang el tipo de idioma con el que será escrita la página web en este caso se hace uso del idioma español. A su vez se muestra la apertura de un head en el cual se proveerá la información como lo es el título y lo enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assets” y de las sub carpetas de “img”, “css” y “js”.

```

personal.php
43 <!DOCTYPE html>
44 <html lang="es">
45
46 <head>
47 <meta charset="UTF-8">
48 <meta name="viewport" content="width=device-width, initial-scale=1">
49 <title>Personal</title>
50 <link rel="icon" href="../../assets/img/logopa.png" type="image/png">
51 <link rel="stylesheet" href="../../assets/css/normalize.css">
52 <link rel="stylesheet" href="../../assets/css/sweetalert2.min.css">
53 <link rel="stylesheet" href="../../assets/css/material.min.css">
54 <link rel="stylesheet" href="../../assets/css/material-design-iconic-font.min.css">
55 <link rel="stylesheet" href="../../assets/css/jquery.mCustomScrollbar.css">
56 <link rel="stylesheet" href="../../assets/css/main.css">
57 <link rel="stylesheet" href="../../assets/css/simple-datatables.min.css">
58 <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
59 <script>
60 window.jQuery || document.write('<script src="../../assets/js/jquery-1.11.2.min.js"></script>')
61
62 </script>
63 <script src="../../assets/js/material.min.js"></script>
64 <script src="../../assets/js/sweetalert2.min.js"></script>
65 <script src="../../assets/js/jquery.mCustomScrollbar.concat.min.js"></script>
66 <script src="../../assets/js/marco2.js"></script>
67 <script src="../../assets/js/simple-datatables.js"></script>
68 <!-- links para exportar a excel -->
69 <script src="https://unpkg.com/xlsx@0.16.9/dist/xlsx.full.min.js"></script>
70 <script src="https://unpkg.com/file-saverjs@latest/FileSaver.min.js"></script>
71 <script src="https://unpkg.com/tableexport@latest/dist/js/tableexport.min.js"></script>
72 </head>

```

Figura. 226 Estructura de lenguaje HTML e Idioma del archivo personal.php. Fuente. Elaboración propia.

Una vez definido el encabezado de la página mediante la etiqueta head, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web. Posteriormente se coloca un elemento que ayudara a representar el contenido principal de la página web que es el “body”. Posteriormente se abre una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, se colocan contenedores para aplicar estilos correspondientes a las sentencias, se agregan imágenes que harán la función del avatar del administrador, así como el nombre del usuario administrador que se autentico con anterioridad los datos son llamados desde la base de datos de Firebase, así como se muestra en la figura 227.

```

72 <body>
73 <!-- barra de navegacion lateral -->
74 <section class="full-width navlateral">
75 <div class="full-width navlateral-bg btn-menu"></div>
76 <div class="full-width navlateral-body">
77 <div class="full-width navlateral-body-logo text-center titles">
78 <i class="zmdi zmdi-close btn-menu"></i> CC Camaleon Color
79 </div>
80 <div class="full-width navlateral-body-title-menu">
81 <div>
82 
83 </div>
84 <figcaption>
85 <span><?php echo $_SESSION['usuario']['NombreCompleto'];?></span>&nbsp;<span><?php echo $_SESSION['usuario']['Apellido'];?></span></span>
86 <br>
87 <span><?php echo $_SESSION['usuario']['Usuario_a'];?></small></span>
88 </figcaption>
89 </div>
90

```

Figura. 227 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.

Se muestra en las figuras 228, 229 y 230, 231 se coloca una etiqueta nav para proporcionar una navegación estructurada en la página, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta assets/css.

```

91 <nav class="full-width">
92 <ul class="full-width list-unstyled menu-principal">
93 <li class="full-width">
94 <a href="#" class="full-width">
95 <div class="navLateral-body-cl">
96 <i class="zmdi zmdi-view-dashboard"></i>
97 </div>
98 <div class="navLateral-body-cr">
99 INICIO
100 </div>
101 </a>
102 </li>
103 <li class="full-width divider-menu-h"></li>
104 <li class="full-width">
105 <a href="#" class="full-width btn-subMenu">
106 <div class="navLateral-body-cl">
107 <i class="zmdi zmdi-case"></i>
108 </div>
109 <div class="navLateral-body-cr">
110 ADMINISTRACION
111 </div>
112 <span class="zmdi zmdi-chevron-left"></span>
113 </a>
114 <ul class="full-width menu-principal sub-menu-options">
115 <li class="full-width">
116 <a href="#" class="full-width">
117 <div class="navLateral-body-cl">
118 <i class="zmdi zmdi-truck"></i>
119 </div>
120 <div class="navLateral-body-cr">
121 PROVEEDORES
122 </div>
123 </a>
124 </li>
125 <li class="full-width">
126 <a href="#" class="full-width">
127 <div class="navLateral-body-cl">
128 <i class="zmdi zmdi-card"></i>
129 </div>
130 <div class="navLateral-body-cr">
131 UBICACIONES
132 </div>
133 </a>
134 </li>
135 </ul>
136 </li>
137 </ul>
138 </nav>

```

Figura. 228 Configuración de la barra de navegación lateral pt1. Fuente. Elaboración propia.

```

129 </div>
130 <div class="navLateral-body-cr">
131 UBICACIONES
132 </div>
133 </a>
134 </li>
135 <li class="full-width">
136 <a href="#" class="full-width">
137 <div class="navLateral-body-cl">
138 <i class="zmdi zmdi-label"></i>
139 </div>
140 <div class="navLateral-body-cr">
141 CATEGORIAS
142 </div>
143 </a>
144 </li>
145 <li class="full-width">
146 <a href="#" class="full-width">
147 <div class="navLateral-body-cl">
148 <i class="zmdi zmdi-case"></i>
149 </div>
150 <div class="navLateral-body-cr">
151 PROYECTOS
152 </div>
153 </a>
154 </li>
155 </ul>
156 </li>
157 <li class="full-width divider-menu-h"></li>
158 <li class="full-width">
159 <a href="#" class="full-width btn-subMenu">
160 <div class="navLateral-body-cl">
161 <i class="zmdi zmdi-face"></i>
162 </div>
163 <div class="navLateral-body-cr">
164 USUARIOS
165 </div>
166 <span class="zmdi zmdi-chevron-left"></span>
167 </a>
168 <ul class="full-width menu-principal sub-menu-options">
169 <li class="full-width">
170 <a href="admin.php" class="full-width">

```

Figura. 229 Configuración de la barra de navegación lateral pt2. Fuente. Elaboración propia.

```

178         </li>
179         <li class="full-width">
180             <a href="../../personal/personal.php" class="full-width">
181                 <div class="navLateral-body-cl">
182                     <i class="zmdi zmdi-accounts"></i>
183                 </div>
184                 <div class="navLateral-body-cr">
185                     PERSONAL CALIFICADO
186                 </div>
187             </a>
188         </li>
189     </ul>
190 </li>
191 <li class="full-width divider-menu-h"></li>
192 <li class="full-width">
193     <a href="../../compras/compras.php" class="full-width">
194         <div class="navLateral-body-cl">
195             <i class="zmdi zmdi-shopping-cart"></i>
196         </div>
197         <div class="navLateral-body-cr">
198             COMPRAS
199         </div>
200     </a>
201 </li>
202 <li class="full-width divider-menu-h"></li>
203 <li class="full-width">
204     <a href="../../productos/productos.php" class="full-width">
205         <div class="navLateral-body-cl">
206             <i class="zmdi zmdi-washing-machine"></i>
207         </div>
208         <div class="navLateral-body-cr">
209             ENTRADAS
210         </div>
211     </a>
212 </li>
213 <li class="full-width divider-menu-h"></li>
214 <li class="full-width">
215     <a href="../../salidas/salidas.php" class="full-width">
216         <div class="navLateral-body-cl">
217             <i class="zmdi zmdi-view-agenda"></i>
218         </div>
219         <div class="navLateral-body-cr">
220             SALIDAS
221         </div>
222     </a>
223 </li>
224 <li class="full-width divider-menu-h"></li>
225 <li class="full-width">
226     <a href="../../inventario/inventario.php" class="full-width">
227         <div class="navLateral-body-cl">
228             <i class="zmdi zmdi-store"></i>
229         </div>
230         <div class="navLateral-body-cr">
231             INVENTARIO
232         </div>
233     </a>
234 </li>
235 </ul>
236 </div>
237 </div>
238 </section>
239 </div>

```

Figura. 230 Configuración de la barra de navegación lateral pt3. Fuente. Elaboración propia.

```

198     <div class="navLateral-body-cl">
199         <i class="zmdi zmdi-shopping-cart"></i>
200     </div>
201     <div class="navLateral-body-cr">
202         COMPRAS
203     </div>
204 </a>
205 </li>
206 <li class="full-width divider-menu-h"></li>
207 <li class="full-width">
208     <a href="../../productos/productos.php" class="full-width">
209         <div class="navLateral-body-cl">
210             <i class="zmdi zmdi-washing-machine"></i>
211         </div>
212         <div class="navLateral-body-cr">
213             ENTRADAS
214         </div>
215     </a>
216 </li>
217 <li class="full-width divider-menu-h"></li>
218 <li class="full-width">
219     <a href="../../salidas/salidas.php" class="full-width">
220         <div class="navLateral-body-cl">
221             <i class="zmdi zmdi-view-agenda"></i>
222         </div>
223         <div class="navLateral-body-cr">
224             SALIDAS
225         </div>
226     </a>
227 </li>
228 <li class="full-width divider-menu-h"></li>
229 <li class="full-width">
230     <a href="../../inventario/inventario.php" class="full-width">
231         <div class="navLateral-body-cl">
232             <i class="zmdi zmdi-store"></i>
233         </div>
234         <div class="navLateral-body-cr">
235             INVENTARIO
236         </div>
237     </a>
238 </li>
239 </ul>
240 </div>
241 </div>
242 </section>
243 </div>

```

Figura. 231 Configuración de la barra de navegación lateral pt4. Fuente. Elaboración propia.

Figura. 232 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia

C:\xampp\htdocs\ProyectoCamaleon\users\admin\personal\personal.js (ProyectoCamaleon) - Sublime Text (UNREGISTERED)

233 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

La creación de nuevos contenedores que almacenaran los inputs necesarios para el registro de información, como se muestra en la figura 237.

```

291 </div>
292 <div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
293   <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
294     <input class="mdl-textfield__input" type="text" pattern="[A-Za-záéíóúÁÉÍÓÚ
295       ]*(\.[0-9]+)?" name="Nombrecompleto">
296     <label class="mdl-textfield__label" for="Nombrecompleto">Nombre(s)</label>
297     <span class="mdl-textfield__error">Nombre invalido</span>
298   </div>
299 </div>
300 <div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
301   <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
302     <input class="mdl-textfield__input" type="text" pattern="[A-Za-záéíóúÁÉÍÓÚ
303       ]*(\.[0-9]+)?" name="Apellido">
304     <label class="mdl-textfield__label" for="Apellido">Apellido(s)</label>
305     <span class="mdl-textfield__error">Apellido Invalido</span>
306   </div>
307 </div>
308 <div class="mdl-cell mdl-cell--12-col">
309   <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
310     <input class="mdl-textfield__input" type="email" name="Correo_e">
311     <label class="mdl-textfield__label" for="Correo_e">Correo electrónico </label>
312     <span class="mdl-textfield__error">Correo Invalido</span>
313   </div>
314 </div>
315 <div class="mdl-cell mdl-cell--12-col">
316   <legend class="text-condensedlight"><i class="zmdi zmdi-border-color"></i> &nbsp;DETALLES DE LA CUENTA</legend><br>
317 </div>
318 <div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">

```

Figura. 237 Creación de los campos para el formulario de personal. Fuente. Elaboración propia.

Siendo de esta manera que la estructura de cada input se definirá por el tipo de dato a almacenar, como se observa en la figura 238, ya sea numérico, texto, correo o contraseña. Se definen con la expresión regular de pattern para indicarle el tipo de simbología que deberá de contener, esto no aplica para los campos de correo y contraseña ya que poseen una estructura específica. En caso de incumplir dicha estructura se colocará abajo del input una alerta de texto erróneo.

```

<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
  <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
    <input class="mdl-textfield__input" type="text" pattern="[A-Za-záéíóúÁÉÍÓÚ
      ]*(\.[0-9]+)?" name="Nombrecompleto">
    <label class="mdl-textfield__label" for="Nombrecompleto">Nombre(s)</label>
    <span class="mdl-textfield__error">Nombre invalido</span>
  </div>
</div>

```

Figura. 238 Codificación de la estructura de los campos para el formulario. Fuente. Elaboración propia.

Se muestra en la figura 239 que, al finalizar las inserciones de los input se crea un botón dentro de un contenedor que comenzara el procesamiento de información del formulario.

```

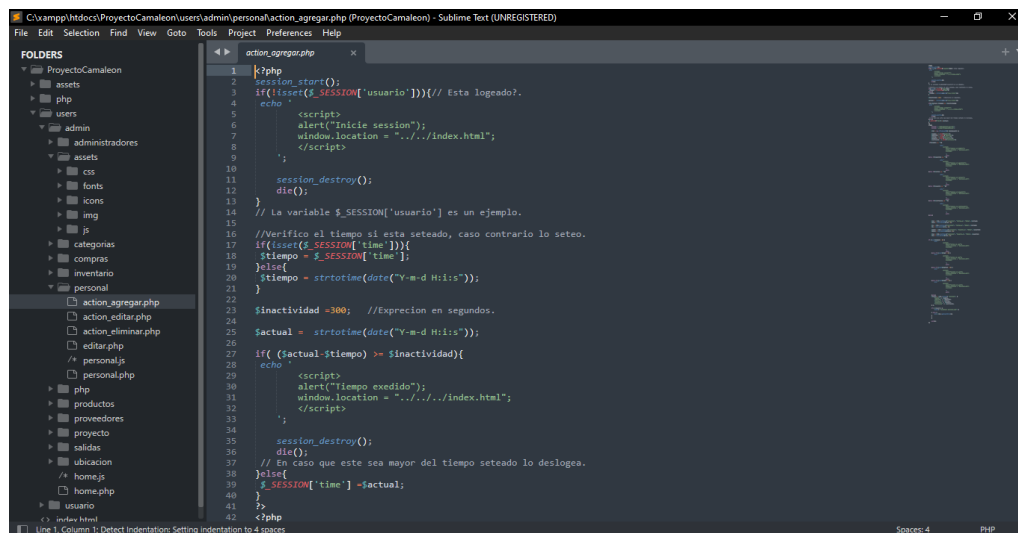
332 <div class="text-center">
333   <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
334     mdl-button--colored bg-primary" id="btn-addClient">
335     <i class="zmdi zmdi-plus"></i>
336   </button>
337   <div class="mdl-tooltip" for="btn-addClient">Agregar Personal</div>
338 </div>
339 </div>
340 </div>
341 </div>
342 </div>
343 </div>
344

```

Figura. 239 Creación del botón agregar personal. Fuente. Elaboración propia.

La figura 240 muestra que, para que el procesamiento de datos este completo el uso del archivo “action_agregar.php” cumple con la función de verificar el tiempo de inactividad

dentro de la página y almacenar los datos obtenidos de los inputs en variables, las cuales serán evaluadas mediante sentencias condicionales identificando si la variable recibió información o esta es nula, si a información obtenida resulta ser nula se regresara al archivo “personal.php”.

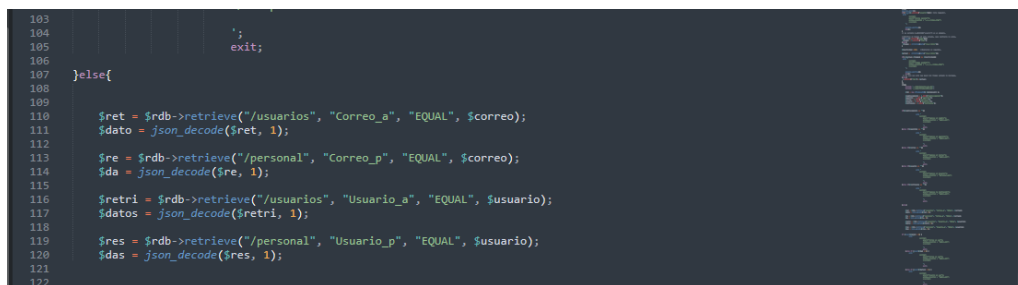


```

1  <?php
2  session_start();
3  if(isset($_SESSION['usuario'])){// esta logeado?.
4      echo
5      <script>
6          alert("Inicie session");
7          window.location = "../index.html";
8      </script>
9      ;
10
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Expresion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28     echo
29     <script>
30         alert("tiempo excedido");
31         window.location = "../index.html";
32     </script>
33     ;
34
35     session_destroy();
36     die();
37 // En caso que este sea mayor del tiempo seteado lo desloga.
38 }else{
39     $_SESSION['time'] = $actual;
40 }
41 ?>
42 </php
  
```

Figura. 240 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente. Elaboración propia

Por lo tanto, si la información es correcta se mandará a llamar a las tablas de administradores y de personal, así como se observa en la figura 241.



```

103
104
105     exit;
106
107 }else{
108
109
110     $ret = $rdb->retrieve("/usuarios", "Correo_a", "EQUAL", $correo);
111     $dato = json_decode($ret, 1);
112
113     $re = $rdb->retrieve("/personal", "Correo_p", "EQUAL", $correo);
114     $da = json_decode($re, 1);
115
116     $retri = $rdb->retrieve("/usuarios", "Usuario_a", "EQUAL", $usuario);
117     $datos = json_decode($retri, 1);
118
119     $res = $rdb->retrieve("/personal", "Usuario_p", "EQUAL", $usuario);
120     $das = json_decode($res, 1);
121
122
  
```

Figura. 241 Codificación para el llamado de tablas en Firebase. Fuente. Elaboración propia.

Por consiguiente, si la información es correcta se verificará por medio de sentencias de decisión si es que los datos proporcionados no están almacenados dentro de las tablas de la base de datos como se muestra en la figura 242, si están almacenados por medio de un script regresará una alerta y redirigirá al archivo personal.php.

```
122
123 if (count($dato) > 0) {
124     echo
125     <script>
126         alert("Correo en uso");
127         window.location = "admin.php";
128     </script>
129     ';
130 }else if (count($da) > 0) {
131     echo
132     <script>
133         alert("Correo en uso");
134         window.location = "admin.php";
135     </script>
136     ';
137     exit;
138 }else if (count($datos) > 0) {
139     echo
140     <script>
141         alert("Usuario en uso");
142         window.location = "admin.php";
143     </script>
144     ';
145     exit;
146 }else if (count($das) > 0) {
147     echo
148     <script>
149         alert("Usuario en uso");
150         window.location = "admin.php";
151     </script>
152     ';
153     exit;
154 }
```

Figura. 242 Codificación de verificación para datos repetidos de los campos. Fuente. Elaboración propia.

En caso de no estar almacenados, se insertará por medio de una variable que almacenará los datos correctos y devolverá al archivo personal.php, así como se muestra en la figura 243.

```
160 $insert = $rdb->insert( "personal", [
161     "Nombre" => $nombre,
162     "Apellido" => $apellido,
163     "Correo_p" => $correo,
164     "Usuario_p" => $usuario,
165     "Contraseña" => $contraseña,
166 ] );
167
168 if ( $insert ) {
169     header( "Location: personal.php" );
170 } else {
171     echo $rdb->getLastError();
172 }
173 }
174 }
175 }
176
177 exit();
178 ?>
179
```

Figura. 243 Codificación de ingreso de datos a la tabla personal. Fuente. Elaboración propia.

Una vez realizado el ingreso de información, así como se observa en la figura 244, se coloca un contenedor el cual almacenara el título de la lista de registros y se da formato a la tabla de registros con los campos acorde a los inputs y registros de la base de datos, se asignan identificadores para dar formato mediante css y hacer uso de Java Script.

```

344 <div class="mdl-tabs_panel" id="tabListClient">
345 <div class="mdl-grid">
346 <div class="mdl-cell mdl-cell--12-col">
347 <div class="full-width panel mdl-shadow--2dp">
348 <div class="full-width panel-title bg-success text-center titles">
349 Lista de Personal
350 </div>
351 <div class="full-width panel-content">
352 <div class="table-responsive text-center">
353 <table class="table table-bordered table-hover" id="datatablesSimple">
354 <thead>
355 <tr>
356 <th scope="col"><strong>ID</strong></th>
357 <th scope="col"><strong>Nombre(s)</strong></th>
358 <th scope="col"><strong>Apellido(s)</strong></th>
359 <th scope="col"><strong>Usuario</strong></th>
360 <th scope="col"><strong>Correo</strong></th>
361 <th scope="col"><strong>Contraseña</strong></th>
362 <th scope="col"><strong>Acción</strong></th>
363 </tr>
364 </thead>
365 <tbody id="tr_personal">
366 <tr>
367 <td></td>
368 <td></td>
369 <td></td>
370 <td></td>
371 <td></td>
372 <td></td>
373 </tr>
374 </tbody>
375 </table>
376 </div>
377 </div>
378 </div>
379 </section>
380 <script type="module" src="personal.js"></script>
381
382

```

Figura. 244 Creación de la tabla de registros para el personal, Fuente. Elaboración propia.

Para llenar la tabla de los registros se ocupa el archivo “personal.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros de cada insert para almacenarlos y mostrarlos en el orden indicado con anterioridad, así como se muestra en la figura 245.

```

26 const personalesRef = ref(database, "personal");
27
28 get(personalesRef).then(snapshot => {
29   const personales = snapshot.val();
30   let rowNumber = 1;
31
32   for (const key in personales) {
33     if (Object.hasOwnProperty.call(personales, key)) {
34       const personal = personales[key];
35
36       const personalTable = document.createElement("tr");
37       personalTable.innerHTML = `
38 <tr>
39 <td><strong>${rowNumber}</strong></td>
40 <td>${personal.Nombre}</td>
41 <td>${personal.Apellido}</td>
42 <td>${personal.Usuario}</td>
43 <td>${personal.Correo}</td>
44 <td>${personal.Contrasena}</td>
45 <td>
46 <div>
47 <a href="editar.php?id=${key}" class="editar"><i class="zmdi zmdi-edit"></i></a>
48 <a href="action_eliminar.php?id=${key}" class="eliminar"><i class="zmdi zmdi-delete"></i></a>
49 </div>
50 </td>
51 </tr>
52 `;
53       rowNumber++;
54       personalTr.appendChild(personalTable);
55     }
56   }
57
58   const dataTable = new simpleDatatables.DataTable("#datatablesSimple");
59   dataTable.addTable(personalTr);
60   dataTable.render();
61   console.error("Error obteniendo datos:", error);
62   CargarDatos();
63 }
64
65
66

```

Figura. 245 Configuración para la visualización de los registros de la tabla de personal, Fuente. Elaboración propia.

En la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, así como se observa en la figura 246, mediante el uso de un script colocado posteriormente del llamado “personal.js”.

```

367         </table>
368         <button id="btnExportar" class="btn btn-success">
369             <i class="fas fa-file-excel"></i> Exportar datos a Excel
370         </button>
371     </div>
372 </div>
373 </div>
374 </div>
375 </div>
376 </div>
377 </section>
378 </script>
379 <script type="module" src="usuarios.js"></script>
380
381 <!-- script para exportar a excel -->
382 <script>
383     const $btnExportar = document.querySelector("#btnExportar");
384     $datatablesSimple = document.querySelector("#datatablesSimple");
385
386     $btnExportar.addEventListener("click", function() {
387         let tableExport = new TableExport($datatablesSimple, {
388             exportButtons: false, // No queremos botones
389             filename: "Reporte de Administradores", //Nombre del archivo de Excel
390             sheetname: "Reporte de Administradores", //Título de la hoja
391         });
392         let datos = tableExport.getExportData();
393         let preferenciasDocumento = datos.datatablesSimple.xlsx;
394         tableExport.export2file(preferenciasDocumento.data, preferenciasDocumento.mimeType, preferenciasDocumento.filename,
395             preferenciasDocumento.fileExtension, preferenciasDocumento.merges, preferenciasDocumento.RTL,
396             preferenciasDocumento.sheetname);
397     });
398 </script>
399 </body>
400 </html>
401

```

Figura. 246 Codificación y conversión de registros a formato Excel, Fuente. Elaboración propia.

En relación con el archivo admin.php se encuentra “editar.php” perteneciente a la misma carpeta, se encuentran las configuraciones iniciales de la barra de navegación como se muestra en la figura 247, no obstante, se hace un llamado a la conexión de la base de datos dentro de la interpretación de php, se especifica la tabla a la cual se tendrá acceso de información.

```

ProjectoCamaleon\users\admin\administradores\editar.php (ProjectoCamaleon) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS
ProjectoCamaleon
├── assets
├── php
├── users
└── admin
    ├── administradores
    │   ├── action_agregar.php
    │   ├── action_editar.php
    │   ├── action_eliminar.php
    │   ├── admin.php
    │   ├── editar.php
    │   ├── html2pdf.bundle.min.js
    │   └── usuarios.js
    ├── assets
    ├── categorias
    ├── compras
    └── inventario

editar.php
259
260
261 <?php
262     include '../php/Conexion_be.php';
263     include '../php/firebaseDB.php';
264
265     $db = new firebaseDB($databaseUrl);
266     $id = $_GET['id'];
267     $retrieve = $db->retrieve('usuarios/$id');
268     $data = json_decode($retrieve, 1);
269
270 <section class="full-width header-well">
271     <div class="full-width header-well-icon">
272         <i class="mdi mdi-account"></i>
273     </div>
274     <div class="full-width header-well-text">
275         <p class="text-condensed-light">
276             Edición de administradores,
277             Nota: Recuerda llenar todos los campos de manera adecuada.
278         </p>
279     </div>
280 </section>

```

Figura. 247 Creación y configuración del archivo editar.php, Fuente. Elaboración propia.

Los inputs reciben una alteración de código, permitiendo que los datos almacenados con anterioridad en los campos designados sean llamados y visualizados dentro del mismo input, se conservan las reglas de patrones indicados por el archivo antecesor personal.php, así como se muestra en la figura 248.

```

291 Edición de Administrador
292
293 </div>
294 <div class="full-width panel-content">
295 <form method="post" action="action_editar.php">
296 <div class="mdl-grid">
297 <div class="mdl-cell mdl-cell--12-col">
298 <legend class="text-condensedlight"><i class="zmdi zmdi-border-color"></i> &nbsp;&nbsp;&
299 </div>
300 <div style="display: none;">
301 <input type="text" name="uid" value="<?php echo $id?>">
302 </div>
303 <div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
304 <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
305 <input class="mdl-textfield_input" type="text" pattern="[A-Za-z&#224;éíóú&#223;
306 <label class="mdl-textfield_label" for="Nombrecompleto">Nombre(s)</label>
307 <span class="mdl-textfield_error">Nombre inválido</span>
308 </div>
309 </div>
310 <div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
311 <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
312 <input class="mdl-textfield_input" type="text" pattern="[A-Za-z&#224;éíóú&#223;
313 <label class="mdl-textfield_label" for="Apellido">Apellido(s)</label>
314 <span class="mdl-textfield_error">Apellido inválido</span>
315 </div>
316 </div>
317 <div class="mdl-cell mdl-cell--12-col">
318 <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
319 <input class="mdl-textfield_input" type="email" name="Correo_e" value="<?php
320 <echo $data['Correo_e']?>">
321 <label class="mdl-textfield_label" for="Correo_e">Correo Electronico </label>
322 <span class="mdl-textfield_error">Correo inválido</span>
323 </div>
324 </div>
325 <div class="mdl-cell mdl-cell--12-col">
326 <legend class="text-condensedlight"><i class="zmdi zmdi-border-color"></i> &nbsp;&nbsp;&
327 </div>
328 </div>
329 </div>
330 </div>
331 </div>
332 </div>
333 </div>
334 </div>
335 </div>
336 </div>
337 </div>
338 </div>
339 </div>
340 </div>
341 </div>
342 </div>
343 </div>
344 </div>
345 </div>
346 </div>
347 </div>
348 </div>
349 </div>
350 </div>
351 </div>
352 </div>
353 </div>
354 </div>
355 </div>
356 </div>
357 </div>
358 </div>
359 </div>
360 </div>
361 </div>
362 </div>
363 </div>
364 </div>
365 </div>
366 </div>
367 </div>
368 </div>
369 </div>
370 </div>
371 </div>
372 </div>
373 </div>
374 </div>
375 </div>
376 </div>
377 </div>
378 </div>
379 </div>
380 </div>
381 </div>
382 </div>
383 </div>
384 </div>
385 </div>
386 </div>
387 </div>
388 </div>
389 </div>
390 </div>
391 </div>
392 </div>
393 </div>
394 </div>
395 </div>
396 </div>
397 </div>
398 </div>
399 </div>
400 </div>
401 </div>
402 </div>
403 </div>
404 </div>
405 </div>
406 </div>
407 </div>
408 </div>
409 </div>
410 </div>
411 </div>
412 </div>
413 </div>
414 </div>
415 </div>
416 </div>
417 </div>
418 </div>
419 </div>
420 </div>
421 </div>
422 </div>
423 </div>
424 </div>
425 </div>
426 </div>
427 </div>
428 </div>
429 </div>
430 </div>
431 </div>
432 </div>
433 </div>
434 </div>
435 </div>
436 </div>
437 </div>
438 </div>
439 </div>
440 </div>
441 </div>
442 </div>
443 </div>
444 </div>
445 </div>
446 </div>
447 </div>
448 </div>
449 </div>
450 </div>
451 </div>
452 </div>
453 </div>
454 </div>
455 </div>
456 </div>
457 </div>
458 </div>
459 </div>
460 </div>
461 </div>
462 </div>
463 </div>
464 </div>
465 </div>
466 </div>
467 </div>
468 </div>
469 </div>
470 </div>
471 </div>
472 </div>
473 </div>
474 </div>
475 </div>
476 </div>
477 </div>
478 </div>
479 </div>
480 </div>
481 </div>
482 </div>
483 </div>
484 </div>
485 </div>
486 </div>
487 </div>
488 </div>
489 </div>
490 </div>
491 </div>
492 </div>
493 </div>
494 </div>
495 </div>
496 </div>
497 </div>
498 </div>
499 </div>
500 </div>
501 </div>
502 </div>
503 </div>
504 </div>
505 </div>
506 </div>
507 </div>
508 </div>
509 </div>
510 </div>
511 </div>
512 </div>
513 </div>
514 </div>
515 </div>
516 </div>
517 </div>
518 </div>
519 </div>
520 </div>
521 </div>
522 </div>
523 </div>
524 </div>
525 </div>
526 </div>
527 </div>
528 </div>
529 </div>
530 </div>
531 </div>
532 </div>
533 </div>
534 </div>
535 </div>
536 </div>
537 </div>
538 </div>
539 </div>
540 </div>
541 </div>
542 </div>
543 </div>
544 </div>
545 </div>
546 </div>
547 </div>
548 </div>
549 </div>
550 </div>
551 </div>
552 </div>
553 </div>
554 </div>
555 </div>
556 </div>
557 </div>
558 </div>
559 </div>
560 </div>
561 </div>
562 </div>
563 </div>
564 </div>
565 </div>
566 </div>
567 </div>
568 </div>
569 </div>
570 </div>
571 </div>
572 </div>
573 </div>
574 </div>
575 </div>
576 </div>
577 </div>
578 </div>
579 </div>
580 </div>
581 </div>
582 </div>
583 </div>
584 </div>
585 </div>
586 </div>
587 </div>
588 </div>
589 </div>
590 </div>
591 </div>
592 </div>
593 </div>
594 </div>
595 </div>
596 </div>
597 </div>
598 </div>
599 </div>
600 </div>
601 </div>
602 </div>
603 </div>
604 </div>
605 </div>
606 </div>
607 </div>
608 </div>
609 </div>
610 </div>
611 </div>
612 </div>
613 </div>
614 </div>
615 </div>
616 </div>
617 </div>
618 </div>
619 </div>
620 </div>
621 </div>
622 </div>
623 </div>
624 </div>
625 </div>
626 </div>
627 </div>
628 </div>
629 </div>
630 </div>
631 </div>
632 </div>
633 </div>
634 </div>
635 </div>
636 </div>
637 </div>
638 </div>
639 </div>
640 </div>
641 </div>
642 </div>
643 </div>
644 </div>
645 </div>
646 </div>
647 </div>
648 </div>
649 </div>
650 </div>
651 </div>
652 </div>
653 </div>
654 </div>
655 </div>
656 </div>
657 </div>
658 </div>
659 </div>
660 </div>
661 </div>
662 </div>
663 </div>
664 </div>
665 </div>
666 </div>
667 </div>
668 </div>
669 </div>
670 </div>
671 </div>
672 </div>
673 </div>
674 </div>
675 </div>
676 </div>
677 </div>
678 </div>
679 </div>
680 </div>
681 </div>
682 </div>
683 </div>
684 </div>
685 </div>
686 </div>
687 </div>
688 </div>
689 </div>
690 </div>
691 </div>
692 </div>
693 </div>
694 </div>
695 </div>
696 </div>
697 </div>
698 </div>
699 </div>
700 </div>
701 </div>
702 </div>
703 </div>
704 </div>
705 </div>
706 </div>
707 </div>
708 </div>
709 </div>
710 </div>
711 </div>
712 </div>
713 </div>
714 </div>
715 </div>
716 </div>
717 </div>
718 </div>
719 </div>
720 </div>
721 </div>
722 </div>
723 </div>
724 </div>
725 </div>
726 </div>
727 </div>
728 </div>
729 </div>
730 </div>
731 </div>
732 </div>
733 </div>
734 </div>
735 </div>
736 </div>
737 </div>
738 </div>
739 </div>
740 </div>
741 </div>
742 </div>
743 </div>
744 </div>
745 </div>
746 </div>
747 </div>
748 </div>
749 </div>
750 </div>
751 </div>
752 </div>
753 </div>
754 </div>
755 </div>
756 </div>
757 </div>
758 </div>
759 </div>
760 </div>
761 </div>
762 </div>
763 </div>
764 </div>
765 </div>
766 </div>
767 </div>
768 </div>
769 </div>
770 </div>
771 </div>
772 </div>
773 </div>
774 </div>
775 </div>
776 </div>
777 </div>
778 </div>
779 </div>
780 </div>
781 </div>
782 </div>
783 </div>
784 </div>
785 </div>
786 </div>
787 </div>
788 </div>
789 </div>
790 </div>
791 </div>
792 </div>
793 </div>
794 </div>
795 </div>
796 </div>
797 </div>
798 </div>
799 </div>
800 </div>
801 </div>
802 </div>
803 </div>
804 </div>
805 </div>
806 </div>
807 </div>
808 </div>
809 </div>
810 </div>
811 </div>
812 </div>
813 </div>
814 </div>
815 </div>
816 </div>
817 </div>
818 </div>
819 </div>
820 </div>
821 </div>
822 </div>
823 </div>
824 </div>
825 </div>
826 </div>
827 </div>
828 </div>
829 </div>
830 </div>
831 </div>
832 </div>
833 </div>
834 </div>
835 </div>
836 </div>
837 </div>
838 </div>
839 </div>
840 </div>
841 </div>
842 </div>
843 </div>
844 </div>
845 </div>
846 </div>
847 </div>
848 </div>
849 </div>
850 </div>
851 </div>
852 </div>
853 </div>
854 </div>
855 </div>
856 </div>
857 </div>
858 </div>
859 </div>
860 </div>
861 </div>
862 </div>
863 </div>
864 </div>
865 </div>
866 </div>
867 </div>
868 </div>
869 </div>
870 </div>
871 </div>
872 </div>
873 </div>
874 </div>
875 </div>
876 </div>
877 </div>
878 </div>
879 </div>
880 </div>
881 </div>
882 </div>
883 </div>
884 </div>
885 </div>
886 </div>
887 </div>
888 </div>
889 </div>
890 </div>
891 </div>
892 </div>
893 </div>
894 </div>
895 </div>
896 </div>
897 </div>
898 </div>
899 </div>
900 </div>
901 </div>
902 </div>
903 </div>
904 </div>
905 </div>
906 </div>
907 </div>
908 </div>
909 </div>
910 </div>
911 </div>
912 </div>
913 </div>
914 </div>
915 </div>
916 </div>
917 </div>
918 </div>
919 </div>
920 </div>
921 </div>
922 </div>
923 </div>
924 </div>
925 </div>
926 </div>
927 </div>
928 </div>
929 </div>
930 </div>
931 </div>
932 </div>
933 </div>
934 </div>
935 </div>
936 </div>
937 </div>
938 </div>
939 </div>
940 </div>
941 </div>
942 </div>
943 </div>
944 </div>
945 </div>
946 </div>
947 </div>
948 </div>
949 </div>
950 </div>
951 </div>
952 </div>
953 </div>
954 </div>
955 </div>
956 </div>
957 </div>
958 </div>
959 </div>
960 </div>
961 </div>
962 </div>
963 </div>
964 </div>
965 </div>
966 </div>
967 </div>
968 </div>
969 </div>
970 </div>
971 </div>
972 </div>
973 </div>
974 </div>
975 </div>
976 </div>
977 </div>
978 </div>
979 </div>
980 </div>
981 </div>
982 </div>
983 </div>
984 </div>
985 </div>
986 </div>
987 </div>
988 </div>
989 </div>
990 </div>
991 </div>
992 </div>
993 </div>
994 </div>
995 </div>
996 </div>
997 </div>
998 </div>
999 </div>
1000 </div>

```

Figura. 248 Codificación y llamado de registros en los campos del formulario personal, Fuente. Elaboración propia.

Se crea un botón el cual permite agrega la información proporcionada con anterioridad mediante la redirección al archivo “agregar_personal.php” como se muestra en la figura 249, continuando con el proceso de ingreso de datos.

```

366 <tbody id="tr_personal">
367 </tbody>
368 </table>
369 <button id="btnExportar" class="btn btn-success">
370 <i class="fas fa-file-excel"></i> Exportar datos a Excel
371 </button>
372 </div>
373 </div>
374 </div>

```

Figura. 249 Creación del botón agregar personal, Fuente. Elaboración propia.

Al igual que el archivo `action_agregar` se conservan las instrucciones sobre inactividad, así como se muestra en la figura 250.



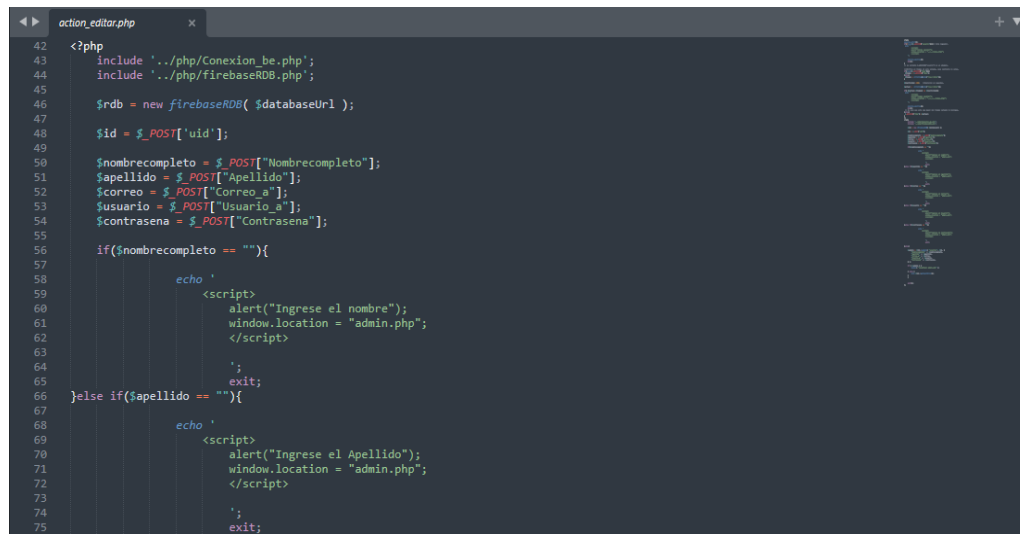
```

1 <?php
2 session_start();
3 if(isset($_SESSION['usuario'])){// Esta logeado?..
4     echo '
5     <script>
6         alert("Inicie session");
7         window.location = "../index.html";
8     </script>
9     '
10
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Expresion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28     echo '

```

Figura. 250 Codificación de tiempo de inactividad del archivo `action_agregar.php`. Fuente, Elaboración propia.

Se hace el llamado a la conexión de la base de datos y creación de variables que almacenaran los cambios indicados en los inputs, a su vez se verifica que dichos inputs contengan información ya que al no contenerla se coloca una alerta y redireccionamiento al archivo “`personal.php`”, así como se observa en la figura 251.



```

42 <?php
43 include '../php/Conexion_be.php';
44 include '../php/firebaseROB.php';
45
46 $rdb = new firebaseROB( $databaseUrl );
47
48 $id = $_POST['uid'];
49
50 $nombrecompleto = $_POST["Nombrecompleto"];
51 $apellido = $_POST["Apellido"];
52 $correo = $_POST["Correo_a"];
53 $usuario = $_POST["Usuario_a"];
54 $contrasena = $_POST["Contrasena"];
55
56 if($nombrecompleto == ""){
57     echo '
58     <script>
59         alert("Ingrese el nombre");
60         window.location = "admin.php";
61     </script>
62     '
63     exit;
64 }else if($apellido == ""){
65     echo '
66     <script>
67         alert("Ingrese el Apellido");
68         window.location = "admin.php";
69     </script>
70     '
71     exit;
72 }

```

Figura. 251 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia.

Por otra parte, como se muestra en la figura 254, se encuentra el archivo “action_eliminar” ubicado dentro de la misma carpeta de administración, Al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad.

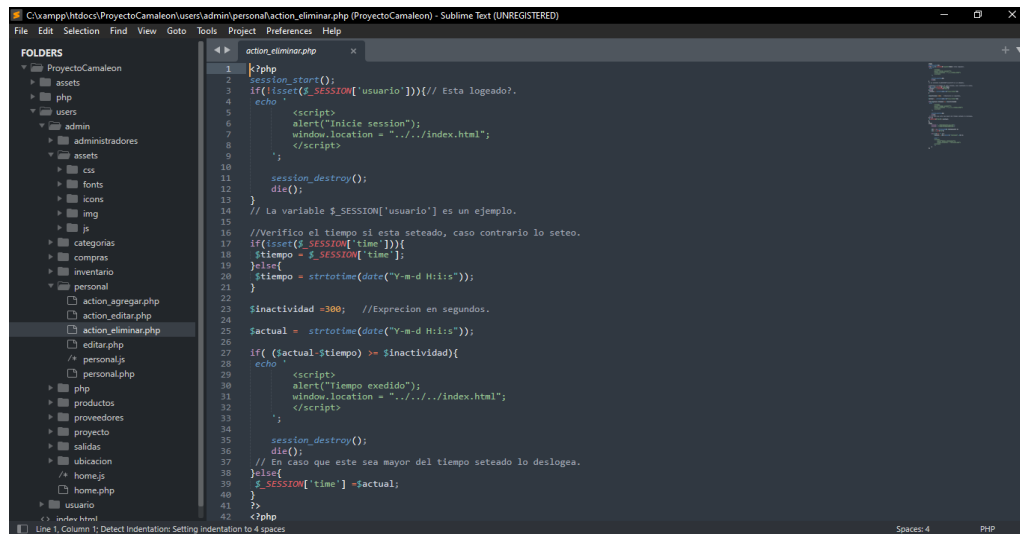


Figura. 254 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente, Elaboración propia.

Se hace el llamado a la conexión de la base de datos y se comienza el borrado de registro mediante una sentencia de evaluación que hace uso del identificador del registro y la variable delete como se muestra en la figura 255, una vez finalizada la eliminación se coloca una alerta sobre la eliminación y se redirecciona al archivo “personal.php”.

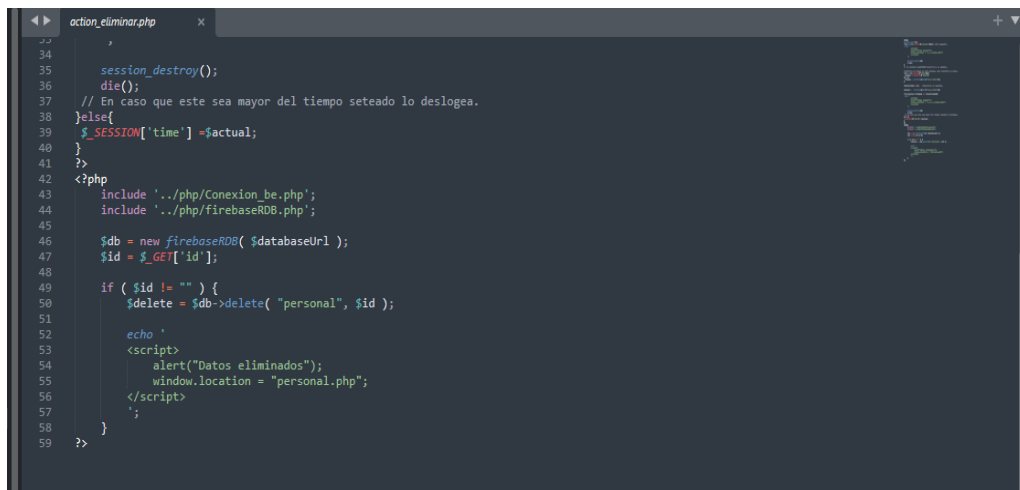


Figura. 255 Codificación y eliminación de registros de la tabla personal, Fuente. Elaboración propia.

No obstante, para que las funcionalidades de edición y eliminación empiecen su proceso, dentro de la tabla de visualización de registros se visualizan 2 botones de acción para cada proceso correspondiente, dichos botones son ubicados en el archivo “personal.js” en figura 256.

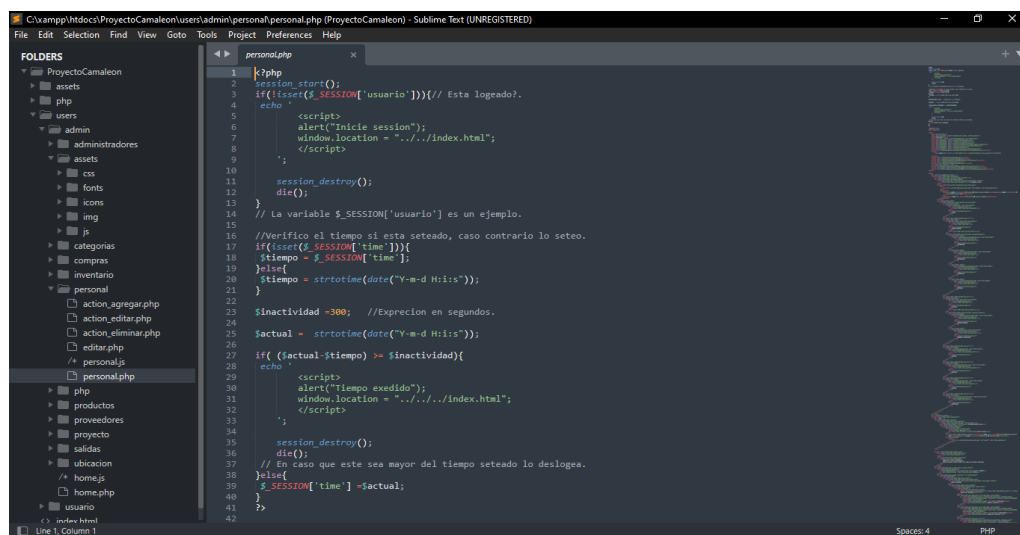
```

44         <td>{usuario.Contrasena}</td>
45     </td>
46     <div>
47         <a href="editar.php?id={key}" class="editar"><i class="zmdi zmdi-edit"></i></a>
48         <a href="action_eliminar.php?id={key}" class="eliminar"><i class="zmdi zmdi-delete"></i></a>
49     </div>
50 </td>
51 </tr>
52 `;
53
54     rowNumber++;
55     usuariosTr.appendChild(usuarioTable);
56 }
    
```

Figura. 256 Codificación de enlaces de edición y eliminación, Fuente. Elaboración propia.

3.10.9. Productos:

Por parte del archivo “productos.php” se especifica en lenguaje de php mediante sentencias de decisión donde se evalúa si una vez en la página de home.php se autentico de manera correcta, en caso de no ser así se coloca una alerta y retornara a index.html. A sí mismo como se muestra en la figura 257, evalúa el tiempo de inactividad entre navegación de archivos teniendo un conteo de 300 segundos de inactividad. Si el conteo es completado se retornará a index.html de manera forzada, en caso de navegar entre archivos el conteo se reiniciará de manera automática, Culminando con el cierre del lenguaje de php.



```

1  <?php
2  session_start();
3  if(!isset($_SESSION['usuario'])){// Esta logeado?.
4      echo `
5          <script>
6              alert("Inicie session");
7              window.location = "../index.html";
8          </script>
9      `;
10
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Expreccion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if (($actual $tiempo) >= $inactividad){
28     echo `
29         <script>
30             alert("Tiempo excedido");
31             window.location = "../index.html";
32         </script>
33     `;
34     session_destroy();
35     die();
36 // En caso que este sea mayor del tiempo seteado lo desloga.
37 }else{
38     $_SESSION['time'] = $actual;
39 }
40 }
41 ?>
42
    
```

Figura. 257 Configuración de lenguaje y tiempo de inactividad del archivo productos.php, Fuente. Elaboración propia.

Se coloca la estructura de `<!DOCTYPE html>` la cual informara al navegador el tipo de versión HTML que fue implementado en el archivo, se muestra la apertura de un head en el cual se proveerá la información como lo es el título y lo enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assents” y de las sub carpetas de “img”, “css” y “js”, como se observa en la figura 258.

```

47 <meta charset="UTF-8">
48 <meta name="viewport" content="width=device-width, initial-scale=1">
49 <title>Personal</title>
50 <link rel="icon" href="../../assets/img/logo.png" type="image/png">
51 <link rel="stylesheet" href="../../assets/css/normalize.css">
52 <link rel="stylesheet" href="../../assets/css/sweetalert2.css">
53 <link rel="stylesheet" href="../../assets/css/material.min.css">
54 <link rel="stylesheet" href="../../assets/css/material-design-iconic-font.min.css">
55 <link rel="stylesheet" href="../../assets/css/jquery.mCustomScrollbar.css">
56 <link rel="stylesheet" href="../../assets/css/main.css">
57 <link rel="stylesheet" href="../../assets/css/simple-datatables.min.css">
58 <script src="//ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
59 <script>
60     window.jQuery || document.write('<script src="../../assets/js/jquery-1.11.2.min.js"></script>')
61 </script>
62 <script src="../../assets/js/material.min.js"></script>
63 <script src="../../assets/js/sweetalert2.min.js"></script>
64 <script src="../../assets/js/jquery.mCustomScrollbar.concat.min.js"></script>
65 <script src="../../assets/js/marco2.js"></script>
66 <script src="../../assets/js/simple-datatables.js"></script>
67 <!-- links para exportar a excel -->
68 <script src="https://unpkg.com/xlsx@0.16.9/dist/xlsx.full.min.js"></script>
69 <script src="https://unpkg.com/file-saver@latest/filesaver.min.js"></script>
70 <script src="https://unpkg.com/tableexport@latest/dist/js/tableexport.min.js"></script>
71

```

Figura. 258 Ingreso de Título y enlaces de funciones del archivo productos.php. Fuente. Elaboración propia.

Una vez definido el encabezado de la página mediante la etiqueta head como se observa en la figura 259, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web. Posteriormente se coloca un elemento que ayudara a representar el contenido principal de la página web que es el “body”. Posteriormente se abre una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, se colocan contenedores para aplicar estilos correspondientes a las sentencias, se agregan imágenes que harán la función del avatar del administrador, así como el nombre del usuario administrador que se autentico con anterioridad los datos son llamados desde la base de datos de Firebase.

```

72
73 <body>
74 <!-- barra de navegacion lateral -->
75 <section class="full-width navLateral">
76 <div class="full-width navLateral-bg btn-menu"></div>
77 <div class="full-width navLateral-body">
78 <div class="full-width navLateral-body-logo text-center tittles">
79 <i class="zmdi zmdi-close btn-menu"></i> CC Camaleon Color
80 </div>
81 <figure class="full-width navLateral-body-tittle-menu">
82 <div>
83 
84 </div>
85 <figcaption>
86 <span><span><?php echo $_SESSION['usuario']['NombreCompleto'];></span>&nbsp;<span><?php echo $_SESSION['
87 <span><small><?php echo $_SESSION['usuario']['Usuario_a'];></small></span>
88 </span></figcaption>
89 </figure>
90

```

Figura. 259 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.

Posteriormente se coloca una etiqueta nav para proporcionar una navegación estructurada en la página, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta assets/css, así como se muestra en las figuras 260, 261 y 263.

```

100 </div>
101 </a>
102 </li>
103 <li class="full-width divider-menu-h"></li>
104 <li class="full-width">
105 <a href="#" class="full-width btn-subMenu">
106 <div class="navLateral-body-cl">
107 <i class="zmdi zmdi-case"></i>
108 </div>
109 <div class="navLateral-body-cr">
110 ADMINISTRACION
111 </div>
112 </a>
113 <span class="zmdi zmdi-chevron-left"></span>
114 </li>
115 <ul class="full-width menu-principal sub-menu-options">
116 <li class="full-width">
117 <a href="../../proveedores/proveedores.php" class="full-width">
118 <div class="navLateral-body-cl">
119 <i class="zmdi zmdi-truck"></i>
120 </div>
121 <div class="navLateral-body-cr">
122 PROVEEDORES
123 </div>
124 </li>
125 <li class="full-width">
126 <a href="../../ubicacion/ubicacion.php" class="full-width">
127 <div class="navLateral-body-cl">
128 <i class="zmdi zmdi-card"></i>
129 </div>
130 <div class="navLateral-body-cr">
131 UBICACIONES
132 </div>
133 </li>
134 </ul>
135 </li>
136 <li class="full-width">
137 <a href="../../categorias/categorias.php" class="full-width">
138 <div class="navLateral-body-cl">
139 <i class="zmdi zmdi-label"></i>
140 </div>
141 <div class="navLateral-body-cr">
142 CATEGORIAS
143 </div>
144 </li>
145 <li class="full-width">
146 <a href="../../proyecto/proyectos.php" class="full-width">
147 <div class="navLateral-body-cl">
148 <i class="zmdi zmdi-case"></i>
149 </div>
150 <div class="navLateral-body-cr">
151 PROYECTOS
152 </div>
153 </li>
154 </ul>
155 </li>
156 <li class="full-width divider-menu-h"></li>
157 <li class="full-width">
158 <a href="#" class="full-width btn-subMenu">
159 <div class="navLateral-body-cl">
160 <i class="zmdi zmdi-face"></i>
161 </div>
162 <div class="navLateral-body-cr">
163 USUARIOS
164 </div>
165 </a>
166 <span class="zmdi zmdi-chevron-left"></span>
167 </li>
168 <ul class="full-width menu-principal sub-menu-options">
169 <li class="full-width">
170 <a href="admin.php" class="full-width">

```

Figura. 260 Configuración de la barra de navegación lateral pt1. Fuente. Elaboración propia.

```

129 </div>
130 <div class="navLateral-body-cr">
131 UBICACIONES
132 </div>
133 </a>
134 </li>
135 <li class="full-width">
136 <a href="../../categorias/categorias.php" class="full-width">
137 <div class="navLateral-body-cl">
138 <i class="zmdi zmdi-label"></i>
139 </div>
140 <div class="navLateral-body-cr">
141 CATEGORIAS
142 </div>
143 </li>
144 <li class="full-width">
145 <a href="../../proyecto/proyectos.php" class="full-width">
146 <div class="navLateral-body-cl">
147 <i class="zmdi zmdi-case"></i>
148 </div>
149 <div class="navLateral-body-cr">
150 PROYECTOS
151 </div>
152 </li>
153 </ul>
154 </li>
155 <li class="full-width divider-menu-h"></li>
156 <li class="full-width">
157 <a href="#" class="full-width btn-subMenu">
158 <div class="navLateral-body-cl">
159 <i class="zmdi zmdi-face"></i>
160 </div>
161 <div class="navLateral-body-cr">
162 USUARIOS
163 </div>
164 </a>
165 <span class="zmdi zmdi-chevron-left"></span>
166 </li>
167 <ul class="full-width menu-principal sub-menu-options">
168 <li class="full-width">
169 <a href="admin.php" class="full-width">
170 </li>

```

Figura. 261 Configuración de la barra de navegación lateral pt2. Fuente. Elaboración propia.

```

178 </li>
179 <li class="full-width">
180 <a href="personal/personal.php" class="full-width">
181 <div class="navLateral-body-cl">
182 <i class="zmdi zmdi-accounts"></i>
183 </div>
184 <div class="navLateral-body-cr">
185 PERSONAL CALIFICADO
186 </div>
187 </a>
188 </li>
189 </ul>
190 </li>
191 <li class="full-width divider-menu-h"></li>
192 <li class="full-width">
193 <a href="compras/compras.php" class="full-width">
194 <div class="navLateral-body-cl">
195 <i class="zmdi zmdi-shopping-cart"></i>
196 </div>
197 <div class="navLateral-body-cr">
198 COMPRAS
199 </div>

```

Figura. 262 Configuración de la barra de navegación lateral pt3. Fuente. Elaboración propia.

```

200 </a>
201 </li>
202 <li class="full-width divider-menu-h"></li>
203 <li class="full-width">
204 <a href="productos/productos.php" class="full-width">
205 <div class="navLateral-body-cl">
206 <i class="zmdi zmdi-washing-machine"></i>
207 </div>
208 <div class="navLateral-body-cr">
209 ENTRADAS
210 </div>
211 </a>
212 </li>
213 <li class="full-width divider-menu-h"></li>
214 <li class="full-width">
215 <a href="salidas/salidas.php" class="full-width">
216 <div class="navLateral-body-cl">
217 <i class="zmdi zmdi-view-agenda"></i>
218 </div>
219 <div class="navLateral-body-cr">
220 SALIDAS
221 </div>
222 </a>
223 </li>
224 <li class="full-width divider-menu-h"></li>
225 <li class="full-width">
226 <a href="inventario/inventario.php" class="full-width">
227 <div class="navLateral-body-cl">
228 <i class="zmdi zmdi-store"></i>
229 </div>
230 <div class="navLateral-body-cr">
231 INVENTARIO

```

Figura. 263 Configuración de la barra de navegación lateral pt4. Fuente. Elaboración propia.

Por parte del contenido de la página web como se observa en la figura 264, se coloca una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, la cual contendrá contenedores para aplicar estilos con css, además de continuar con las animaciones de la barra de navegación, se presenta la configuración para ocultar y mostrar la barra de navegación. Se programa la interfaz de cierre de sesión mediante la implementación de un botón con identificador. Posteriormente se coloca una imagen con el avatar de identificación del administrador y se manda a llamar los datos propios de este, desde la base de datos Firebase.

```

239 <!-- contenido de pagina -->
240 <section class="full-width pageContent">
241 <!-- navBar -->
242 <div class="full-width navBar">
243 <div class="zmdi zmdi-swap btn-menu" id="btn-menu"></div>
244 <div class="mdl-tooltip" for="btn-menu">Ocultar/Mostar Menu</div>
245 <nav class="navBar-options-list">
246 <ul class="list-unstyled">
247 <li class="btn-exit" id="btn-exit">
248 <i class="zmdi zmdi-power"></i>
249 <div class="mdl-tooltip" for="btn-exit">Cerrar Sesion</div>
250 </li>
251 <li class="text-condensedLight noLink"><small><small><?php echo $_SESSION['usuario']['NombreCompleto']></small></small></li>
252 <li class="noLink">
253 <figure>
254 
255 </figure>

```

Figura. 264 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Las animaciones correspondientes al cierre de sesión y ocultar/mostrar la barra de navegación son estructuradas en el archivo “marco.js” ubicado en la carpeta assets/js, así como se observa en la figura 265.

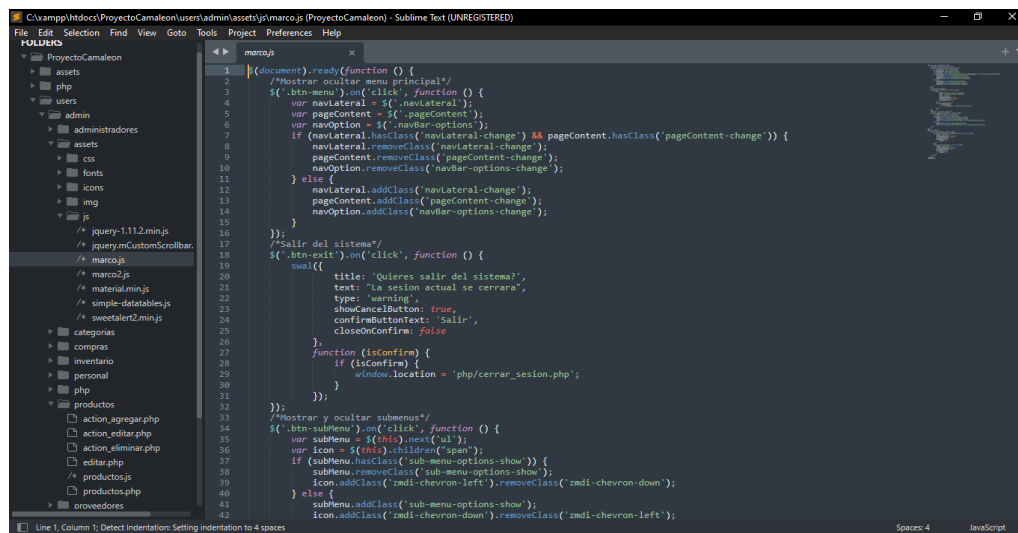


Figura. 265 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Posteriormente como se observa en la figura 266, se coloca una etiqueta de section con sus respectivos contenedores para dar formato a la descripción del formulario.

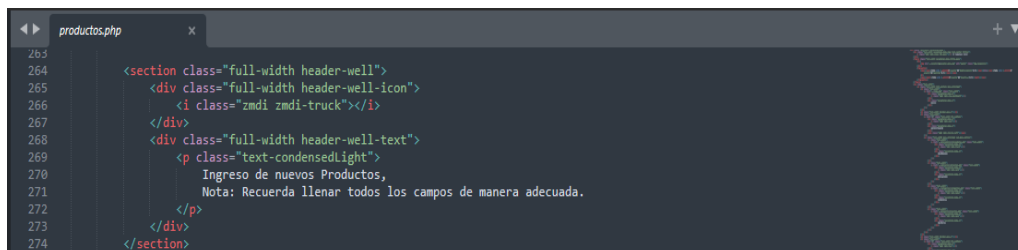


Figura. 266 Creación del contenedor de indicaciones para el formulario, Fuente. Elaboración propia.

A su vez como se muestra en la figura 267, se coloca un nuevo contenedor el cual contendrá dos enlaces que permitirán navegar dentro de la misma página, uno para visualizar el contenido de los registros del formulario en una tabla y el segundo que muestra el formulario a ser completado.

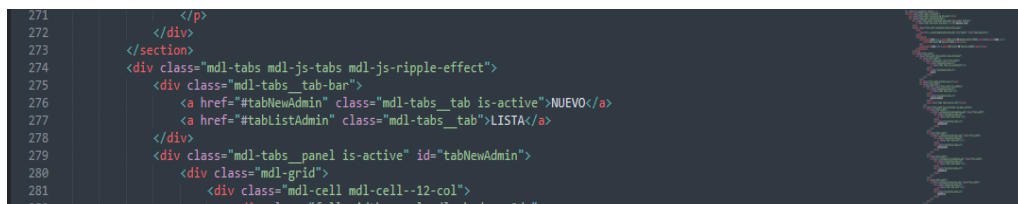
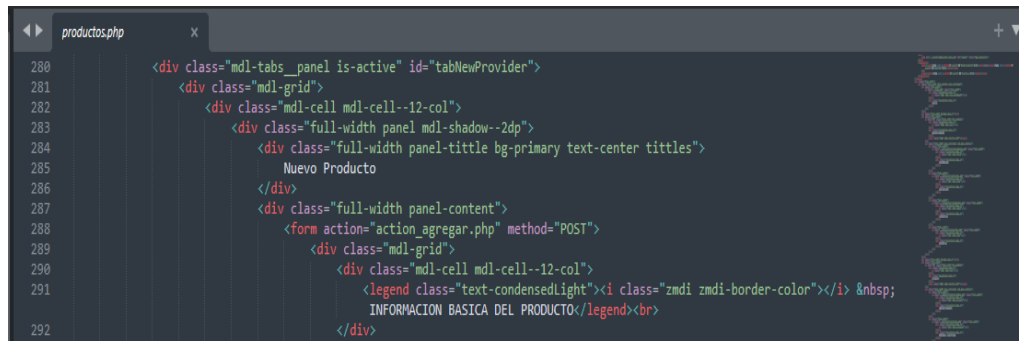


Figura. 267 Creación de enlaces navegadores del archivo productos.php. Fuente. Elaboración propia.

Por un lado, como se muestra en la figura 268, una vez colocados los enlaces se crea un contenedor donde se define el título del formulario y se expresa en texto los datos que deberán ser llenados a continuación.



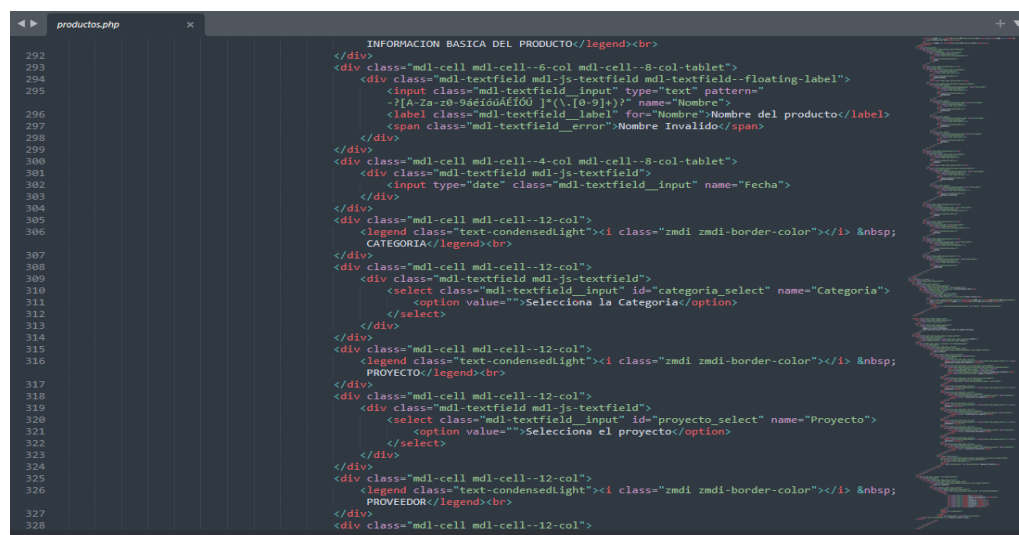
```

280 <div class="mdl-tabs_panel is-active" id="tabNewProvider">
281 <div class="mdl-grid">
282 <div class="mdl-cell mdl-cell--12-col">
283 <div class="full-width panel mdl-shadow--2dp">
284 <div class="full-width panel-tittle bg-primary text-center tittles">
285 Nuevo Producto
286 </div>
287 <div class="full-width panel-content">
288 <form action="action.agregar.php" method="POST">
289 <div class="mdl-grid">
290 <div class="mdl-cell mdl-cell--12-col">
291 <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp;   
INFORMACION BASICA DEL PRODUCTO</legend><br>
292 </div>

```

Figura. 268 Creación de título del formulario e indicaciones. Fuente. Elaboración propia.

Así mismo como se muestra en la figura 269, la creación de nuevos contenedores que almacenaran los inputs necesarios para el registro de información.



```

292 </div>
293 <div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
294 <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
295 <input class="mdl-textfield_input" type="text" pattern=
"?[A-Za-z0-9áéíóúâëîï]?(\\.?[0-9])?" name="Nombre">
296 <label class="mdl-textfield_label" for="Nombre">Nombre del producto</label>
297 <span class="mdl-textfield_error">Nombre Invalido</span>
298 </div>
299 </div>
300 <div class="mdl-cell mdl-cell--4-col mdl-cell--8-col-tablet">
301 <div class="mdl-textfield mdl-js-textfield">
302 <input type="date" class="mdl-textfield_input" name="Fecha">
303 </div>
304 </div>
305 <div class="mdl-cell mdl-cell--12-col">
306 <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp;   
CATEGORIA</legend><br>
307 <div class="mdl-cell mdl-cell--12-col">
308 <div class="mdl-textfield mdl-js-textfield">
309 <select class="mdl-textfield_input" id="categoria_select" name="Categoria">
310 <option value="">Selecciona la Categoria</option>
311 </select>
312 </div>
313 </div>
314 <div class="mdl-cell mdl-cell--12-col">
315 <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp;   
PROYECTO</legend><br>
316 <div class="mdl-cell mdl-cell--12-col">
317 <div class="mdl-textfield mdl-js-textfield">
318 <select class="mdl-textfield_input" id="proyecto_select" name="Proyecto">
319 <option value="">Selecciona el proyecto</option>
320 </select>
321 </div>
322 </div>
323 <div class="mdl-cell mdl-cell--12-col">
324 <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp;   
PROVEEDOR</legend><br>
325 <div class="mdl-cell mdl-cell--12-col">
326 </div>
327 </div>
328 </div>

```

Figura. 269 Creación de los campos para el formulario de productos, Fuente. Elaboración propia.

Siendo de esta manera que la estructura de cada input se definirá por el tipo de dato a almacenar, ya sea numérico, texto, correo o contraseña. Se definen con la expresión regular de pattern para indicarle el tipo de simbología que deberá de contener, esto no aplica para los campos de correo y contraseña ya que poseen una estructura específica, a su vez como se muestra en la figura 270, se coloca el formato las listas de selección las cuales pueden contener información predestinada mediante etiquetas “option” o en su caso incluir un archivo js implementando información desde la base de datos.

En caso de incumplir dicha estructura se colocará abajo del input una alerta de texto erróneo.

```
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
  <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
    <input class="mdl-textfield__input" type="text" pattern="-?[A-Za-záéíóúÁÉÍÓÚ
    ]*(\.[0-9]+)?" name="Nombrecompleto">
    <label class="mdl-textfield__label" for="Nombrecompleto">Nombre(s)</label>
    <span class="mdl-textfield__error">Nombre invalido</span>
  </div>
</div>
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
```

Figura. 270 Codificación de la estructura de los campos para el formulario, Fuente. Elaboración propia.

Al finalizar las inserciones de los input se crea un botón dentro de un contenedor que comenzara el procesamiento de información del formulario, como se muestra en la figura 271.

```
345
346
347   <div class="text-center">
348     <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
349       mdl-button--colored bg-primary" id="btn-addProduct">
350       <i class="zmdi zmdi-plus"></i>
351     </button>
352     <div class="mdl-tooltip" for="btn-addProduct">Agregar Producto</div>
353   </div>
354 </div>
355 </div>
356 </div>
357 </div>
```

Figura. 271 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente. Elaboración propia.

Se observa en la figura 272 que, para que el procesamiento de datos este completo el uso del archivo “action_agregar.php” cumple con la función de verificar el tiempo de inactividad dentro de la página y almacenar los datos obtenidos de los inputs y listas de selección en variables, las cuales serán evaluadas mediante sentencias condicionales identificando si la variable recibió información o esta es nula, si a información obtenida resulta ser nula se regresara al archivo “productos.php”.

```
17 if(isset($_SESSION['time'])){
18   $tiempo = $_SESSION['time'];
19 }else{
20   $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Expiracion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if(( $actual-$tiempo) >= $inactividad){
28   echo '
29     <script>
30       alert("Tiempo excedido");
31       window.location = "../../index.html";
32     </script>
33   ';
34   session_destroy();
35   die();
36   // En caso que este sea mayor del tiempo seteado lo desloga.
37 }else{
38   $_SESSION['time'] = $actual;
39 }
40
41 <?php
```

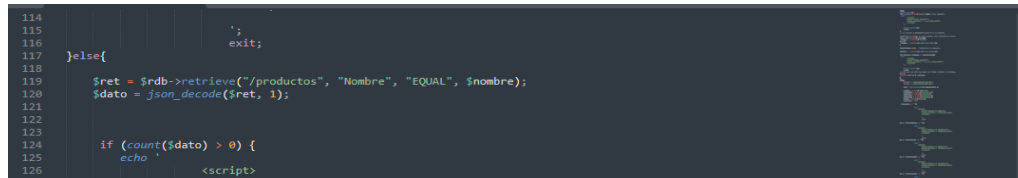
Figura. 272 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente. Elaboración propia.

Por lo tanto, como se muestra en la figura 273, si la información es correcta se mandará a llamar a la tabla de productos.

```

114
115
116
117
118
119
120
121
122
123
124
125
126

```



```

    ';
    exit;
} else {
    $ret = $rdb->retrieve("/productos", "Nombre", "EQUAL", $nombre);
    $dato = json_decode($ret, 1);
    if (count($dato) > 0) {
        echo '
        <script>

```

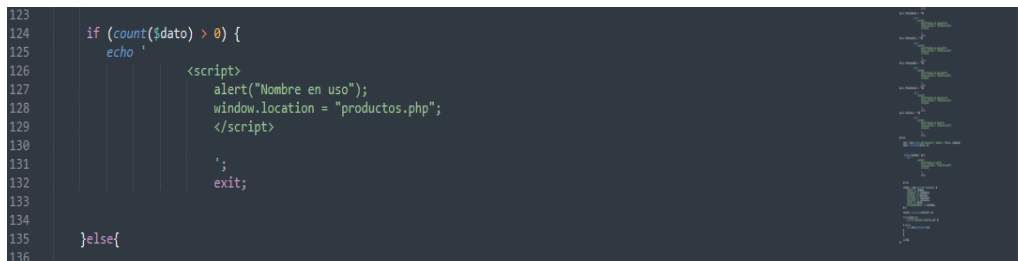
Figura. 273 Codificación para el llamado de tablas en Firebase, Fuente. Elaboración propia.

Por consiguiente, como se observa en la figura 274, si la información es correcta se verificará por medio de sentencias de decisión si es que los datos proporcionados no están almacenados dentro de las tablas de la base de datos, si están almacenados por medio de un script regresara una alerta y redirigía al archivo productos.php.

```

123
124
125
126
127
128
129
130
131
132
133
134
135
136

```



```

    if (count($dato) > 0) {
        echo '
        <script>
            alert("Nombre en uso");
            window.location = "productos.php";
        </script>
        ';
        exit;
    } else {

```

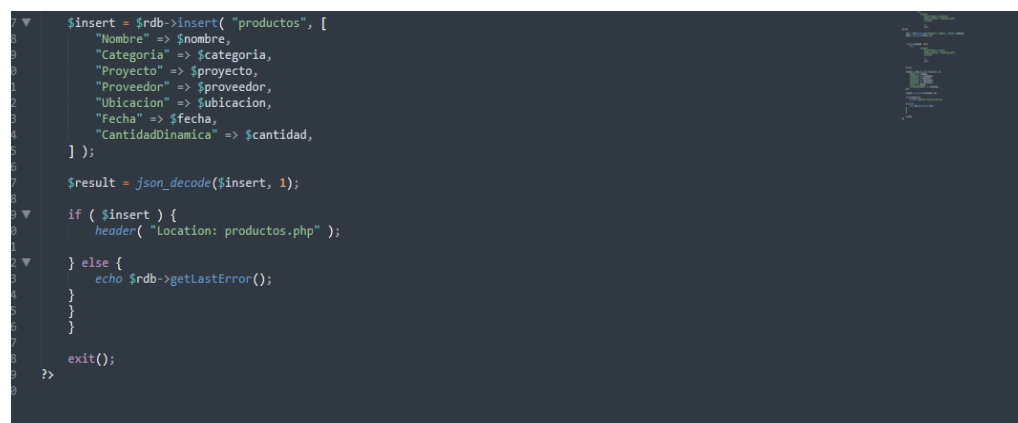
Figura. 274 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia.

Por otro lado, en caso de no estar almacenados, se insertará por medio de una variable que almacenará los datos correctos y devolverá al archivo productos.php, como se muestra en la figura 275.

```

7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```



```

$insert = $rdb->insert( "productos", [
    "Nombre" => $nombre,
    "Categoria" => $categoria,
    "Proyecto" => $proyecto,
    "Proveedor" => $proveedor,
    "Ubicacion" => $ubicacion,
    "Fecha" => $fecha,
    "CantidadDinamica" => $cantidad,
]);
$result = json_decode($insert, 1);
if ( $insert ) {
    header( "Location: productos.php" );
} else {
    echo $rdb->getLastError();
}
exit();
?>

```

Figura. 275 Codificación de ingreso de datos a la tabla productos, Fuente. Elaboración propia.

Una vez realizado el ingreso de información se coloca un contenedor el cual almacenara el título de la lista de registros y se da formato a la tabla de registros con los campos acorde a los inputs y registros de la base de datos, así como se muestra en la figura 276, se asignan identificadores para dar formato mediante css y hacer uso de Java Script.

```

359 <div class="mdl-grid">
360 <div class="mdl-cell mdl-cell-12-col">
361 <div class="full-width panel mdl-shadow--2dp">
362 <div class="full-width panel-title bg-success text-center titles">
363     Lista de Productos
364 </div>
365 <div class="full-width panel-content">
366 <div class="table-responsive text-center">
367 <table class="table table-bordered table-hover" id="datatablesSimple">
368 <thead>
369 <tr>
370 <th scope="col"><strong>ID</strong></th>
371 <th scope="col"><strong>Nombre del producto</strong></th>
372 <th scope="col"><strong>Fecha</strong></th>
373 <th scope="col"><strong>Categoría</strong></th>
374 <th scope="col"><strong>Proyecto</strong></th>
375 <th scope="col"><strong>Proveedor</strong></th>
376 <th scope="col"><strong>Ubicación</strong></th>
377 <th scope="col"><strong>Acción</strong></th>
378 </tr>
379 </thead>
380 <tbody id="tr_productos">
381 </tbody>
382 </table>
383 <div class="fas fa-file-excel"></div> Exportar datos a Excel
384 </div>
385 </div>
386 </div>
387 </div>
388 </div>
389 </div>
390 </div>
391 </div>
392 </div>
393 </div>
394 </div>
395 </div>
396 </div>
397 </div>
398 <script type="module" src="productos.js"></script>
399 <script>
400 const $btnExportar = document.querySelector("#btnExportar");

```

Figura. 276 Creación de la tabla de registros para productos, Fuente. Elaboración propia.

Para llenar la tabla de los registros se ocupa el archivo “productos.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros de cada insert para almacenarlos y mostrarlos en el orden indicado con anterioridad, como se muestra en la figura 277.

```

42 function cargarDatos() {
43     const productosRef = ref(database, "productos");
44     const categoriasRef = ref(database, "categorias");
45     const proyectosRef = ref(database, "proyectos");
46     const proveedoresRef = ref(database, "proveedores");
47     const ubicacionesRef = ref(database, "ubicacion");
48
49     get(productosRef).then(snapshot => {
50         const productos = snapshot.val();
51         let rowNumber = 1;
52
53         for (const key in productos) {
54             if (Object.hasOwnProperty.call(productos, key)) {
55                 const producto = productos[key];
56
57                 const productosTable = document.createElement("tr");
58                 productosTable.innerHTML = `
59 <tr>
60 <td><strong>${rowNumber}</strong></td>
61 <td>${producto.Nombre}</td>
62 <td>${producto.Fecha}</td>
63 <td>${producto.Categoría}</td>
64 <td>${producto.Proyecto}</td>
65 <td>${producto.Proveedor}</td>
66 <td>${producto.Ubicación}</td>
67 <td>
68 <div>
69 <a href="editar.php?id=${key}" class="editar"><i class="zmdi zmdi-edit"></i></a>
70 <a href="action_eliminar.php?id=${key}" class="eliminar"><i class="zmdi zmdi-delete"></i></a>
71 </div>
72 </td>
73 </tr>
74 `;
75                 rowNumber++;
76                 productosTr.appendChild(productosTable);
77             }
78         }
79         const dataTable = new simpleDatatables.DataTable("#datatablesSimple");
80     }).catch(error => {

```

Figura. 277 Conexión a la base de datos e incorporación de registros de la tabla productos, Fuente. Elaboración propia.

De ahí que como se muestra en la figura 278, dentro de la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, mediante el uso de un script colocado posteriormente del llamado “productos.js”.

```

378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417

```

Figura. 278 Codificación y conversión de registros a formato Excel. Fuente. Elaboración propia.

En relación con el archivo productos.php se encuentra “editar.php” perteneciente a la misma carpeta, se encuentran las configuraciones iniciales de la barra de navegación como se muestra en la figura 279, no obstante, se hace un llamado a la conexión de la base de datos dentro de la interpretación de PHP, se especifica la tabla a la cual se tendrá acceso de información.

```

258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273

```

Figura. 279 Creación y configuración del archivo editar.php, Fuente. Elaboración propia.

Posteriormente los inputs reciben una alteración de código como se observa en la figura 280, permitiendo que los datos almacenados con anterioridad en los campos designados sean llamados y visualizados dentro del mismo input y listas de selección, se conservan las reglas de patrones indicados por el archivo antecesor productos.php.

```

293 <form action="action_editar.php" method="POST">
294 <div class="mdl-grid">
295 <div class="mdl-cell mdl-cell--12-col">
296 <legend class="text-condensed-light"><i class="zmdi zmdi-border-color"></i> &nbsp;
INFORMACION BASICA DEL PRODUCTO</legend><br>
297 </div>
298 <div style="display: none;">
299 <input type="text" name="uid" value="{<php echo $id}>">
300 </div>
301 <div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
302 <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
303 <input class="mdl-textfield__input" type="text" pattern=
"[A-Za-z0-9áéíóúÁÉÍÓÚ]*([0-9])?" name="Nombre" value="{<php echo $data[
'Nombre']}>">
304 <label class="mdl-textfield__label" for="Nombre">Nombre del producto</label>
305 <span class="mdl-textfield__error">Nombre Invalido</span>
306 </div>
307 </div>
308 <div class="mdl-cell mdl-cell--4-col mdl-cell--8-col-tablet">
309 <div class="mdl-textfield mdl-js-textfield">
310 <input type="date" class="mdl-textfield__input" name="Fecha" value="{<php
echo $data['Fecha']}>">
311 </div>
312 </div>
313 <div class="mdl-cell mdl-cell--12-col">
314 <legend class="text-condensed-light"><i class="zmdi zmdi-border-color"></i> &nbsp;
CATEGORIA</legend><br>
315 <div class="mdl-cell mdl-cell--12-col">

```

Figura. 280 Codificación y llamado de registros en los campos del formulario productos. Fuente. Elaboración propia.

Se coloca un botón permitiendo la redirección al archivo “action_editar” continuando con el proceso de actualización de datos del formulario, así como se observa en la figura 281.

```

352 </div>
353 </div>
354 <div class="text-center">
355 <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
mdl-button--colored bg-primary" id="btn-addProduct">
356 <i class="zmdi zmdi-plus"></i>
357 </button>
358 <div class="mdl-tooltip" for="btn-addProduct">Agregar Producto</div>
359 </div>
360 </form>
361 </div>
362 </div>
363 </div>
364 </div>
365 </div>
366 <div class="mdl-tabs_panel" id="tablistClient">

```

Figura. 281 Creación del botón agregar producto, Fuente. Elaboración propia.

Al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad, como se observa en la figura 282.

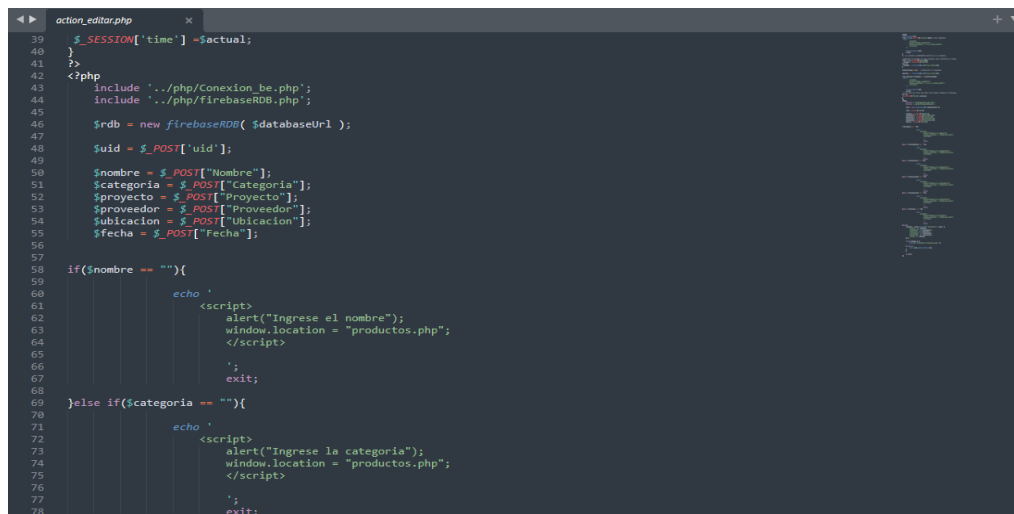
```

1 <?php
2 session_start();
3 if(!isset($_SESSION['usuario'])){// Esta logeado?.
4 echo "
5 <script>
6 alert("Inicie session");
7 window.location = "../index.html";
8 </script>
9 ";
10
11 session_destroy();
12 die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18 $tiempo = $_SESSION['time'];
19 }else{
20 $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Expresion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26

```

Figura. 282 Codificación de tiempo de inactividad del archivo action_editar.php. Fuente. Elaboración propia.

Se hace el llamado a la conexión de la base de datos y creación de variables que almacenaran los cambios indicados en los input y listas, como se muestra en la figura 283, a su vez se verifica que dichos inputs contengan información ya que al no contenerla se coloca una alerta y redireccionamiento al archivo “productos.php”.



```

39  $_SESSION['time'] = $actual;
40  }
41  }
42  <?php
43  include '../php/Conexion_be.php';
44  include '../php/firebaseRDB.php';
45
46  $rdb = new firebaseRDB( $databaseUrl );
47
48  $uid = $_POST['uid'];
49
50  $nombre = $_POST["Nombre"];
51  $categoria = $_POST["Categoria"];
52  $proyecto = $_POST["Proyecto"];
53  $proveedor = $_POST["Proveedor"];
54  $ubicacion = $_POST["Ubicacion"];
55  $fecha = $_POST["Fecha"];
56
57
58  if($nombre == ""){
59
60      echo '
61
62      <script>
63          alert("Ingrese el nombre");
64          window.location = "productos.php";
65      </script>
66      ';
67      exit;
68  }
69  }else if($categoria == ""){
70
71      echo '
72
73      <script>
74          alert("Ingrese la categoria");
75          window.location = "productos.php";
76      </script>
77      ';
78      exit;

```

Figura. 283 Conexión a la base de datos y verificación de campos. Fuente. Elaboración propia.

En cambio, al proceso de ingreso de información como se observa en la figura 284, se anula la evaluación de datos repetidos y se comienza el proceso de actualización de datos, el cual una vez finalizado dará redireccionamiento al archivo productos.php.



```

110
111  echo '
112  <script>
113      alert("Ingrese la fecha");
114      window.location = "productos.php";
115  </script>
116  ';
117  exit;
118  }else{
119
120      $update = $rdb->update( "productos", $uid, [
121          "Nombre" => $nombre,
122          "Categoria" => $categoria,
123          "Proyecto" => $proyecto,
124          "Proveedor" => $proveedor,
125          "Ubicacion" => $ubicacion,
126          "Fecha" => $fecha,
127      ] );
128
129      if ( $update ) {
130          header( "Location: productos.php" );
131      }
132      } else {
133          echo $rdb->getLastError();
134      }
135  }
136
137  exit();
138  ?>

```

Figura. 284 Codificación de actualización de registros de la tabla productos, Fuente. Elaboración propia.

Lo que se muestra en la figura 285 es el llenado de la tabla de los registros se ocupa el archivo “productos.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros que fueron actualizados y mostrarlos en el orden indicado con anterioridad.

```

364 </div>
365 </div>
366 <div class="mdl-tabs_panel" id="tablistClient">
367 <div class="mdl-grid">
368 <div class="mdl-cell mdl-cell--12-col">
369 <div class="full-width panel mdl-shadow--2dp">
370 <div class="full-width panel-title bg-success text-center titles">
371 Lista de Productos
372 </div>
373 <div class="full-width panel-content">
374 <div class="table-responsive text-center">
375 <table class="table table-bordered table-hover" id="datatablesSimple">
376 <thead>
377 <tr>
378 <th scope="col"><strong>ID</strong></th>
379 <th scope="col"><strong>Nombre del producto</strong></th>
380 <th scope="col"><strong>Fecha</strong></th>
381 <th scope="col"><strong>Categoría</strong></th>
382 <th scope="col"><strong>Proyecto</strong></th>
383 <th scope="col"><strong>Proveedor</strong></th>
384 <th scope="col"><strong>Ubicación</strong></th>
385 <th scope="col"><strong>Acción</strong></th>
386 </tr>
387 </thead>
388 <tbody id="tr_productos">
389 </tbody>
390 </table>
391 </div>
392 </div>
393 </div>
394 </div>
395 </div>
396 </div>
397 </div>
398 </div>
399 </div>
400 <script type="module" src="productos.js"></script>
401 </body>
402 </html>
403 </html>
404 </html>
405

```

Figura. 285 Configuración para la visualización de los registros de la tabla de productos. Fuente. Elaboración propia.

Por otra parte, como se observa en la figura 286, se encuentra el archivo “action_eliminar” ubicado dentro de la misma carpeta de administración, Al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad.

```

1 <?php
2 session_start();
3 if(!isset($_SESSION['usuario'])){// Esta logeado?.
4 echo '
5 <script>
6 alert("Inicie session");
7 window.location = "../index.html";
8 </script>
9 '
10 session_destroy();
11 die();
12 }
13 // La variable $_SESSION['usuario'] es un ejemplo.
14
15 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
16 if(isset($_SESSION['time'])){
17 $tiempo = $_SESSION['time'];
18 }else{
19 $tiempo = strtotime(date("Y-m-d H:i:s"));
20 }
21
22 $inactividad =300; //Exprecion en segundos.
23
24 $actual = strtotime(date("Y-m-d H:i:s"));
25
26 if( ($actual-$tiempo) >= $inactividad){
27 echo '
28 <script>
29 alert("Tiempo excedido");
30 window.location = "../index.html";
31 </script>
32 '
33 session_destroy();
34 die();
35 // En caso que este sea mayor del tiempo seteado lo desloga.
36 }else{
37 $_SESSION['time'] = $actual;
38 }
39 <?php
40

```

Figura. 286 Codificación de tiempo de inactividad del archivo action_eliminar.php, Fuente, Elaboración propia.

Se hace el llamado a la conexión de la base de datos y se comienza el borrado de registro mediante una sentencia de evaluación que hace uso del identificador del registro y la variable delete, una vez finalizada la eliminación se coloca una alerta sobre la eliminación y se redirecciona al archivo “productos.php”, así como se observa en la figura 287.

```

35 session_destroy();
36 die();
37 // En caso que este sea mayor del tiempo seteado lo desloga.
38 }else{
39 $_SESSION['time'] = $actual;
40 }
41 }
42 }
43 <?php
44 include '../php/Conexion_be.php';
45 include '../php/firebaseRDB.php';
46 $db = new firebaseRDB( $databaseUrl );
47 $id = $_GET['id'];
48
49 if ( $id != "" ) {
50     $delete = $db->delete( "productos", $id );
51
52     echo '
53     <script>
54         alert("Datos eliminados");
55         window.location = "productos.php";
56     </script>
57     ';
58 }
59 }

```

Figura. 287 Codificación y eliminación de registros de la tabla usuario, Fuente. Elaboración propia.

Se muestra en la figura 288, las funcionalidades de edición y eliminación empiecen su proceso, dentro de la tabla de visualización de registros se visualizan 2 botones de acción para cada proceso correspondiente, dichos botones son ubicados en el archivo “productos.js”.

```

44 <td>{{usuario.Contrasena}}</td>
45 <td>
46 <div>
47 <a href="editar.php?id={{key}}" class="editar"><i class="zmdi zmdi-edit"></i></a>
48 <a href="action_eliminar.php?id={{key}}" class="eliminar"><i class="zmdi zmdi-delete"></i></a>
49 </div>
50 </td>
51 </tr>
52 `;
53
54 rowNumber++;
55 usuariosTr.appendChild(usuarioTable);
56 }
57 }
58

```

Figura. 288 Codificación de vínculos para los botones eliminar y editar, Fuente. Elaboración propia.

3.10.10. Proveedores:

Por parte del archivo “proveedores.php” se especifica en lenguaje de php mediante sentencias de decisión donde se evalúa si una vez en la página de home.php se autentico de manera correcta, como se observa en la figura 289, en caso de no ser así se coloca una alerta y retornara a index.html.

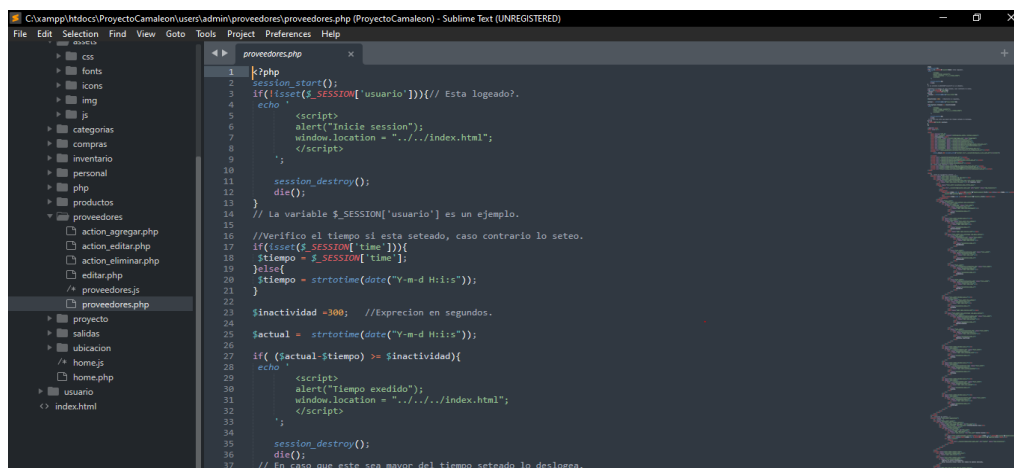


Figura. 289 Configuración de lenguaje y tiempo de inactividad del archivo proveedores.php. Fuente. Elaboración propia.

Posteriormente se coloca la estructura de !DOCTYPE html> la cual informara al navegador el tipo de versión HTML que fue implementado en el archivo, a su vez se indica con una etiqueta de tipo lang el tipo de idioma con el que será escrita la página web en este caso se hace uso del idioma español. A su vez se muestra la apertura de un head en el cual se proveerá la información como lo es el título y lo enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assets” y de las subcarpetas de “img”, “css” y “js”, así como se observa en la figura 290.



Figura. 290 Estructura de lenguaje HTML e Idioma del archivo proveedores.php. Fuente. Elaboración propia.

Una vez definido el encabezado de la página mediante la etiqueta head, así como se observa en la figura 291, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web. Posteriormente se coloca un elemento que ayudara a representar el contenido principal de la página web que es el “body”. Posteriormente se abre una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, se colocan contenedores para aplicar estilos correspondientes a las sentencias, se agregan imágenes que harán la función del avatar del administrador, así como el nombre del usuario administrador que se autentico con anterioridad los datos son llamados desde la base de datos de Firebase.

```

73 <body>
74 <!-- barra de navegacion lateral -->
75 <section class="full-width navlateral">
76 <div class="full-width navlateral-bg btn-menu"></div>
77 <div class="full-width navlateral-body">
78 <div class="full-width navlateral-body-logo text-center tittles">
79 <i class="zmdi zmdi-close btn-menu"></i> CC Camaleon Color
80 </div>
81 <figure class="full-width navlateral-body-tittle-menu">
82 <div>
83 
84 </div>
85 <figcaption>
86 <span><?php echo $_SESSION['usuario']['Nombrecompleto'];></span>&nbsp;<span><?php echo $_SESSION['
87 usuario']['Apellido'];></span></span></div>
88 <span><small><?php echo $_SESSION['usuario']['Usuario a.1'];></small></span>

```

Figura. 291 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia

Posteriormente se coloca una etiqueta nav para proporcionar una navegación estructurada en la página, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta assets/css, así como se observa en las figuras 292, 293 y 294,295.

```

91 <nav class="full-width">
92 <ul class="full-width list-unstyled menu-principal">
93 <li class="full-width">
94 <a href="../../home.php" class="full-width">
95 <div class="navlateral-body-cl">
96 <i class="zmdi zmdi-view-dashboard"></i>
97 </div>
98 <div class="navlateral-body-cr">
99 INICIO
100 </div>
101 </a>
102 </li>
103 <li class="full-width divider-menu-h"></li>
104 <li class="full-width">
105 <a href="#" class="full-width btn-subMenu">
106 <div class="navlateral-body-cl">
107 <i class="zmdi zmdi-case"></i>
108 </div>
109 <div class="navlateral-body-cr">
110 ADMINISTRACION
111 </div>
112 <span class="zmdi zmdi-chevron-left"></span>
113 </a>
114 <ul class="full-width menu-principal sub-menu-options">
115 <li class="full-width">
116 <a href="../../proveedores/proveedores.php" class="full-width">
117 <div class="navlateral-body-cl">
118 <i class="zmdi zmdi-truck"></i>
119 </div>
120 <div class="navlateral-body-cr">
121 PROVEEDORES
122 </div>
123 </a>
124 </li>
125 <li class="full-width">
126 <a href="../../ubicacion/ubicacion.php" class="full-width">
127 <div class="navlateral-body-cl">
128 <i class="zmdi zmdi-card"></i>
129 </div>
130 <div class="navlateral-body-cr">
131 UBICACIONES
132 </div>

```

Figura. 292 Configuración de la barra de navegación lateral pt1. Fuente. Elaboración propia.

```

129         </div>
130         <div class="navLateral-body-cr">
131             UBICACIONES
132         </div>
133     </a>
134 </li>
135 <li class="full-width">
136     <a href="../categorias/categorias.php" class="full-width">
137         <div class="navLateral-body-cl">
138             <i class="zmdi zmdi-label"></i>
139         </div>
140         <div class="navLateral-body-cr">
141             CATEGORIAS
142         </div>
143     </a>
144 </li>
145 <li class="full-width">
146     <a href="../proyecto/proyectos.php" class="full-width">
147         <div class="navLateral-body-cl">
148             <i class="zmdi zmdi-case"></i>
149         </div>
150         <div class="navLateral-body-cr">
151             PROYECTOS
152         </div>
153     </a>
154 </li>
155 </ul>
156 </li>
157 <li class="full-width divider-menu-h"></li>
158 <li class="full-width">
159     <a href="#" class="full-width btn-subMenu">
160         <div class="navLateral-body-cl">
161             <i class="zmdi zmdi-face"></i>
162         </div>
163         <div class="navLateral-body-cr">
164             USUARIOS
165         </div>
166     </a>
167     <span class="zmdi zmdi-chevron-left"></span>
168     <ul class="full-width menu-principal sub-menu-options">
169         <li class="full-width">
170             <a href="admin.php" class="full-width">

```

Figura. 293 Configuración de la barra de navegación lateral pt2. Fuente. Elaboración propia.

```

176         </li>
177     </ul>
178 </li>
179 <li class="full-width">
180     <a href="../personal/personal.php" class="full-width">
181         <div class="navLateral-body-cl">
182             <i class="zmdi zmdi-accounts"></i>
183         </div>
184         <div class="navLateral-body-cr">
185             PERSONAL CALIFICADO
186         </div>
187     </a>
188 </li>
189 </ul>
190 </li>
191 <li class="full-width divider-menu-h"></li>
192 <li class="full-width">
193     <a href="../compras/compras.php" class="full-width">
194         <div class="navLateral-body-cl">
195             <i class="zmdi zmdi-shopping-cart"></i>
196         </div>
197         <div class="navLateral-body-cr">
198             COMPRAS
199         </div>
200     </a>
201 </li>
202 <li class="full-width divider-menu-h"></li>
203 <li class="full-width">
204     <a href="../productos/productos.php" class="full-width">

```

Figura. 294 Configuración de la barra de navegación lateral pt3. Fuente. Elaboración propia.

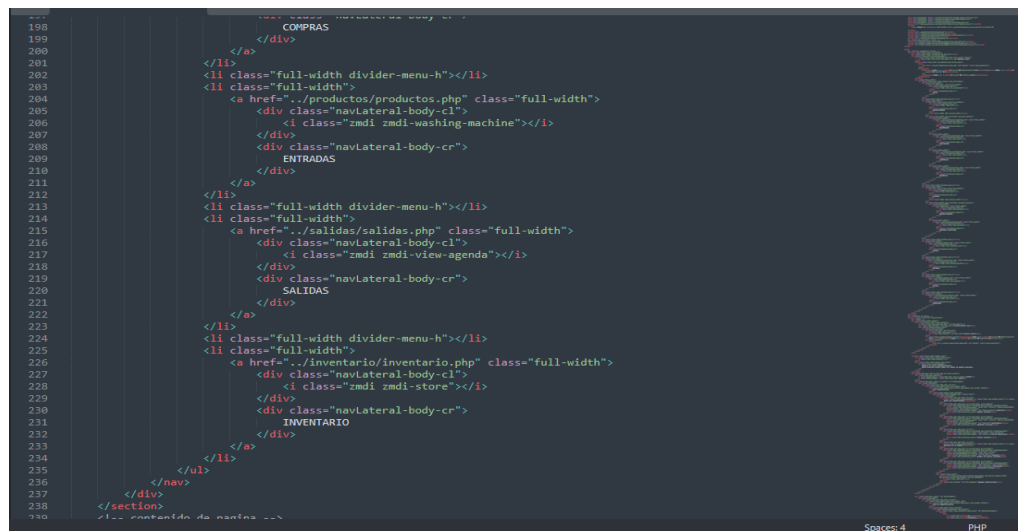


Figura. 295 Configuración de la barra de navegación lateral pt4. Fuente. Elaboración propia.

Por parte del contenido de la página web se coloca una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, la cual contendrá contenedores para aplicar estilos con css, además de continuar con las animaciones de la barra de navegación, se presenta la configuración para ocultar y mostrar la barra de navegación. Se programa la interfaz de cierre de sesión mediante la implementación de un botón con identificador, como se observa en la figura 296. Posteriormente se coloca una imagen con el avatar de identificación del administrador y se manda a llamar los datos propios de este, desde la base de datos Firebase.

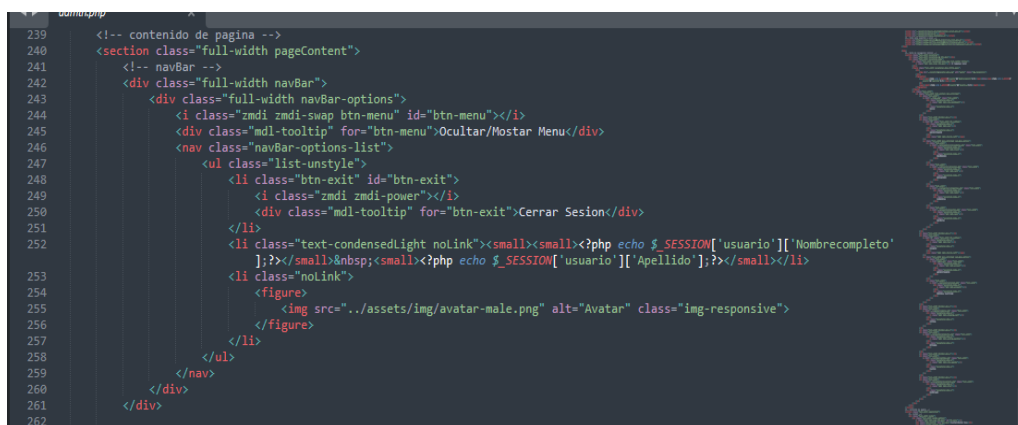


Figura. 296 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Las animaciones correspondientes al cierre de sesión y ocultar/mostrar la barra de navegación son estructuradas en el archivo “marco.js” ubicado en la carpeta assets/js, así como se muestra en la figura 297.

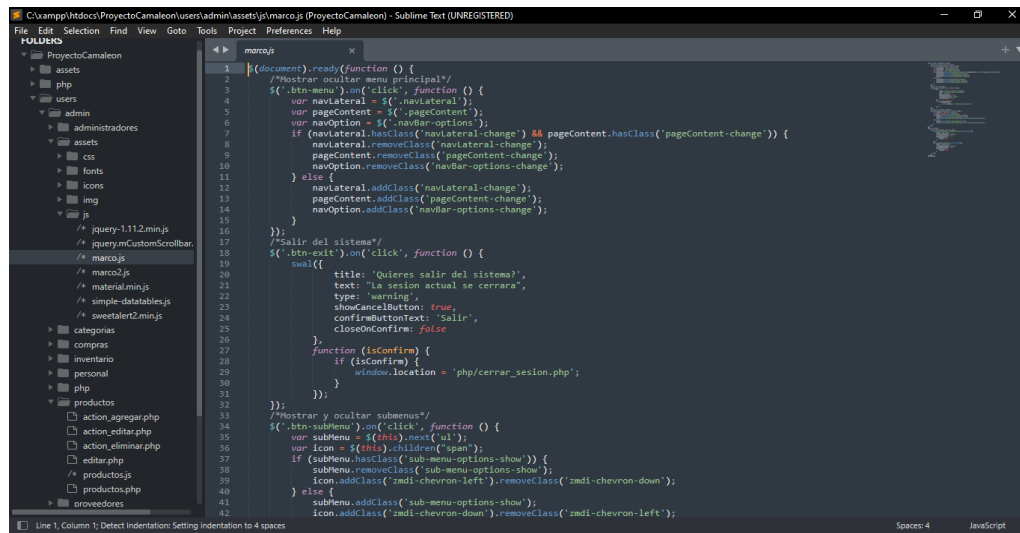


Figura. 297 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.

Se coloca una etiqueta de section con sus respectivos contenedores para dar formato a la descripción del formulario, como se muestra en la figura 28.

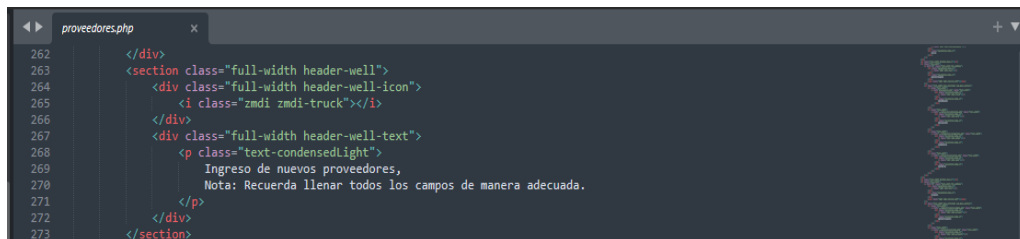


Figura. 298 Creación del contenedor de indicaciones para el formulario, Fuente. Elaboración propia.

A su vez como se observa en la figura 299, se coloca un nuevo contenedor el cual contendrá dos enlaces que permitirán navegar dentro de la misma página, uno para visualizar el contenido de los registros del formulario en una tabla y el segundo que muestra el formulario a ser completado.

```

272     </div>
273 </section>
274 <div class="mdl-tabs mdl-js-tabs mdl-js-ripple-effect">
275   <div class="mdl-tabs__tab-bar">
276     <a href="#tabNewAdmin" class="mdl-tabs__tab is-active">NUEVO</a>
277     <a href="#tabListAdmin" class="mdl-tabs__tab">LISTA</a>
278   </div>
279   <div class="mdl-tabs__panel is-active" id="tabNewAdmin">
280     <div class="mdl-grid">
281       <div class="mdl-cell mdl-cell--12-col">
282         <div class="full-width panel mdl-shadow--2dp">
283           <div class="full-width panel-title bg-primary text-center titles">

```

Figura. 299 Creación de enlaces navegadores del archivo proveedores.php. Fuente. Elaboración propia.

Por un lado, como se observa en la figura 300, una vez colocados los enlaces se crea un contenedor donde se define el título del formulario y se expresa en texto los Datos que deberán ser llenados a continuación.

```

280 <div class="mdl-grid">
281   <div class="mdl-cell mdl-cell--12-col">
282     <div class="full-width panel mdl-shadow--2dp">
283       <div class="full-width panel-title bg-primary text-center titles">
284         Nuevo Proveedor
285       </div>
286       <div class="full-width panel-content">
287         <form action="action_agregar.php" method="POST">
288           <div class="mdl-grid">
289             <div class="mdl-cell mdl-cell--12-col">
290               <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp;
291                 DATOS DEL PROVEEDOR</legend><br>
292             </div>
293             <div class="mdl-cell mdl-cell--12-col">
294               <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
295                 <input class="mdl-textfield__input" type="text" pattern="^[A-Za-z0-9
296                 ]*(\.[0-9]+)" name="Nombre">
297                 <label class="mdl-textfield__label" for="Nombre">Nombre del proveedor </label>
298                 <span class="mdl-textfield__error">Nombre Invalido</span>
299               </div>
300             <div class="mdl-cell mdl-cell--12-col">
301               <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp;
302                 DIRECCION</legend><br>

```

Figura. 300 Creación de título del formulario e indicaciones, Fuente. Elaboración propia.

La creación de nuevos contenedores que almacenaran los inputs necesarios para el registro de información, así como se observa en la figura 1.

```

299 <div class="mdl-cell mdl-cell--12-col">
300   <legend class="text-condensedLight"><i class="zmdi zmdi-border-color"></i> &nbsp;
301     DIRECCION</legend><br>
302 </div>
303 <div class="mdl-cell mdl-cell--12-col">
304   <div class="mdl-textfield mdl-js-textfield">
305     <select class="mdl-textfield__input" name="Países">
306       <option value="">Selecciona el País</option>
307       <option value="Alemania">Alemania</option>
308       <option value="Arabia Saudita">Arabia Saudita</option>
309       <option value="Argentina">Argentina</option>
310       <option value="Australia">Australia</option>
311       <option value="Brasil">Brasil</option>
312       <option value="Canadá">Canadá</option>
313       <option value="China">China</option>
314       <option value="Corea del Sur">Corea del Sur</option>
315       <option value="Estados Unidos">Estados Unidos</option>
316       <option value="Francia">Francia</option>
317       <option value="India">India</option>
318       <option value="Indonesia">Indonesia</option>
319       <option value="Italia">Italia</option>
320       <option value="Japón">Japón</option>
321       <option value="México">México</option>
322       <option value="Reino Unido">Reino Unido</option>
323       <option value="Rusia">Rusia</option>
324       <option value="Sudáfrica">Sudáfrica</option>
325       <option value="Turquía">Turquía</option>
326       <option value="Unión Europea">Unión Europea</option>

```

Figura. 301 Codificación de la estructura de listas de selección. Fuente. Elaboración propia.

```
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
  <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
    <input class="mdl-textfield__input" type="text" pattern="^[A-Za-zéíóúÁÉÍÓÚ
    ]*(\.[0-9]+)?" name="Nombrecompleto">
    <label class="mdl-textfield__label" for="Nombrecompleto">Nombre(s)</label>
    <span class="mdl-textfield__error">Nombre invalido</span>
  </div>
</div>
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
```

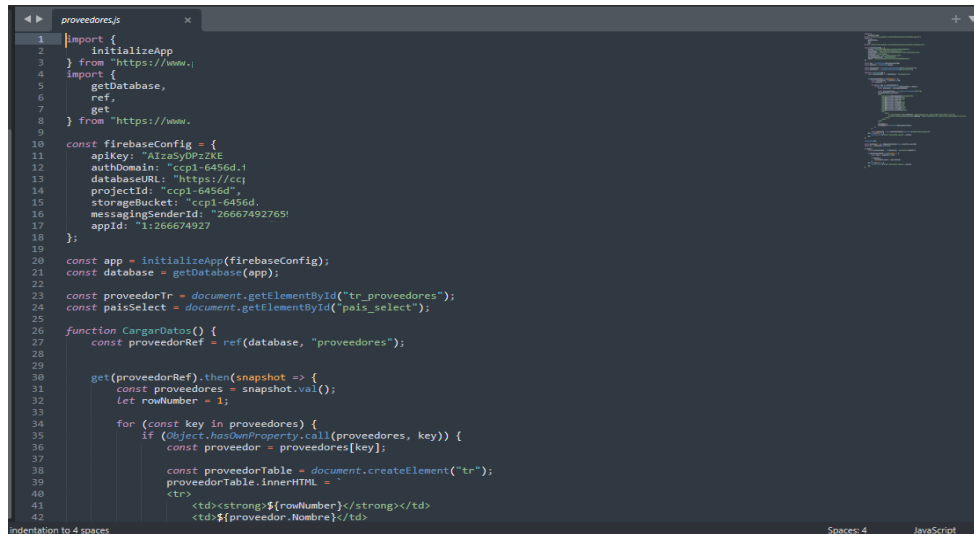
$$\begin{array}{ccccccc} \cdot & 1\cdot & 1 & \cdot & \cdot & 1\cdot 1\cdot & \cdot \\ & & & & & & 1\cdot / & 1\cdot & 1 & & \cdot & 1 \end{array}$$

```

366         </tbody>
367     </table>
368     <button id="btnExportar" class="btn btn-success">
369         <i class="fas fa-file-excel"></i> Exportar datos a Excel
370     </button>
371 </div>
372 </div>
373 </div>
374 </div>
375 </div>
376 </div>
377 </div>
378 </section>
379

```


Dentro del archivo proveedores.js se hace la conexión a la base de datos y se mandan llamar a las tablas necesarias para cargar y almacenar la información en las listas mediante el uso de variables, como se muestra en la figura 305.



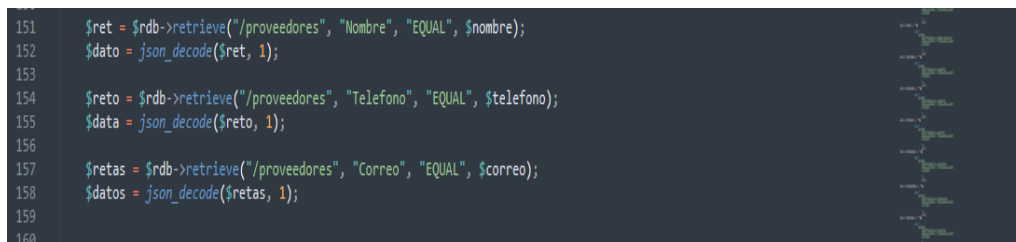
```

1 import {
2   initializeApp
3 } from "https://www.
4 import {
5   getDatabase,
6   ref,
7   get
8 } from "https://www.
9
10 const firebaseConfig = {
11   apiKey: "AIzaSyDPzZKE
12   authDomain: "ccpl-6456d.1
13   databaseURL: "https://ccpl
14   projectId: "ccpl-6456d",
15   storageBucket: "ccpl-6456d.
16   messagingSenderId: "266674927651
17   appId: "1:266674927
18 };
19
20 const app = initializeApp(firebaseConfig);
21 const database = getDatabase(app);
22
23 const proveedorTr = document.getElementById("tr_proveedores");
24 const paisSelect = document.getElementById("pais_select");
25
26 function CargarDatos() {
27   const proveedorRef = ref(database, "proveedores");
28
29   get(proveedorRef).then(snapshot => {
30     const proveedores = snapshot.val();
31     let rowNumber = 1;
32
33     for (const key in proveedores) {
34       if (Object.hasOwnProperty.call(proveedores, key)) {
35         const proveedor = proveedores[key];
36
37         const proveedorTable = document.createElement("tr");
38         proveedorTable.innerHTML =
39           <tr>
40             <td><strong>${rowNumber}</strong></td>
41             <td>${proveedor.Nombre}</td>

```

Figura. 305 Conexión a la base de datos y almacenamiento de información de listas, Fuente. Elaboración propia.

Por lo tanto, si la información es correcta se mandará llamar a las tablas de indicadas, como se muestra en la figura 306.



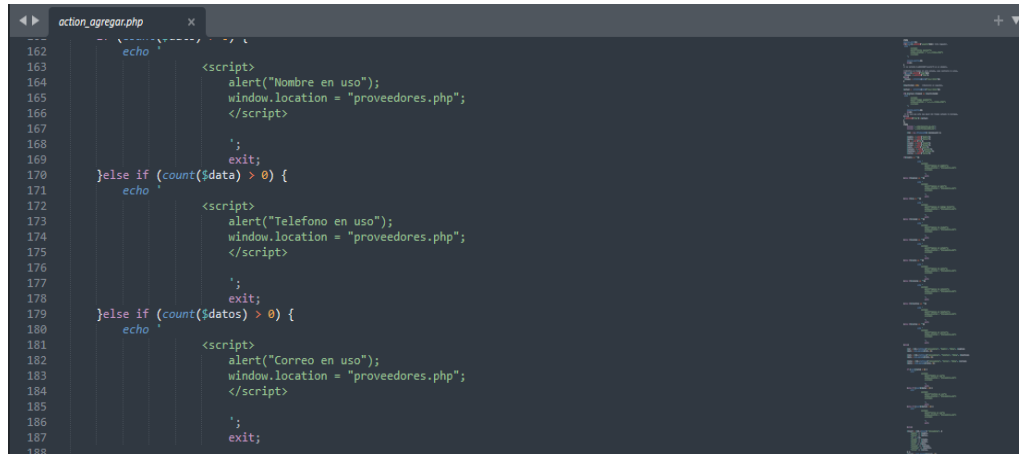
```

151 $ret = $rdb->retrieve("/proveedores", "Nombre", "EQUAL", $nombre);
152 $dato = json_decode($ret, 1);
153
154 $reto = $rdb->retrieve("/proveedores", "Telefono", "EQUAL", $telefono);
155 $data = json_decode($reto, 1);
156
157 $retas = $rdb->retrieve("/proveedores", "Correo", "EQUAL", $correo);
158 $datos = json_decode($retas, 1);
159
160

```

Figura. 306 Codificación para el llamado de tablas en Firebase, Fuente. Elaboración propia.

Por consiguiente, como se muestra en la figura 307, si la información es correcta se verificará por medio de sentencias de decisión si es que los datos proporcionados no están almacenados dentro de las tablas de la base de datos, si están almacenados por medio de un script regresara una alerta y redirigía al archivo proveedores.php.



```

162     echo '
163
164         <script>
165             alert("Nombre en uso");
166             window.location = "proveedores.php";
167         </script>
168
169         '
170     }else if (count($data) > 0) {
171         echo '
172
173         <script>
174             alert("Telefono en uso");
175             window.location = "proveedores.php";
176         </script>
177
178         '
179     }else if (count($datos) > 0) {
180         echo '
181
182         <script>
183             alert("Correo en uso");
184             window.location = "proveedores.php";
185         </script>
186
187         '
188     }

```

Figura. 307 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia.

Por otro lado, como se muestra en la figura 308, en caso de no estar almacenados, se insertará por medio de una variable que almacenará los datos correctos y devolverá al archivo productos.php.



```

190
191     $insert = $rdb->insert( "proveedores", [
192         "Nombre" => $nombre,
193         "Paises" => $paises,
194         "Cp" => $cp,
195         "Ciudad" => $ciudad,
196         "Estado" => $estado,
197         "Calle" => $calle,
198         "Colonia" => $colonia,
199         "Telefono" => $telefono,
200         "Correo" => $correo,
201     ] );
202     $result = json_decode($insert, 1);
203
204     if ( $insert ) {
205         header( "Location: proveedores.php" );
206     } else {
207         echo $rdb->getLastError();
208     }
209
210
211
212     exit();
213
214 }>
215

```

Figura. 308 Codificación de ingreso de datos a la tabla proveedores, Fuente. Elaboración propia.

Una vez realizado el ingreso de información se coloca un contenedor el cual almacenara el título de la lista de registros y se da formato a la tabla de registros con los campos acorde a los inputs y registros de la base de datos, se asignan identificadores para dar formato mediante css y hacer uso de Java Script, así como se observa en la figura 309.

```

397 <div class="mdl-tabs__panel" id="tabListProvider">
398 <div class="mdl-grid">
399 <div class="mdl-cell mdl-cell--12-col">
400 <div class="full-width panel mdl-shadow--2dp">
401 <div class="full-width panel-title bg-success text-center titles">
402 Lista de Proveedores
403 </div>
404 <div class="full-width panel-content">
405 <div class="table-responsive text-center">
406 <table class="table table-bordered table-hover" id="datatablesSimple">
407 <thead>
408 <tr>
409 <th scope="col"><strong>ID</strong></th>
410 <th scope="col"><strong>Nombre</strong></th>
411 <th scope="col"><strong>País</strong></th>
412 <th scope="col"><strong>CP</strong></th>
413 <th scope="col"><strong>Ciudad</strong></th>
414 <th scope="col"><strong>Estado</strong></th>
415 <th scope="col"><strong>Colles</strong></th>
416 <th scope="col"><strong>Colonia</strong></th>
417 <th scope="col"><strong>Correo</strong></th>
418 <th scope="col"><strong>Telefono</strong></th>
419 <th scope="col"><strong>Acción</strong></th>
420 </tr>
421 </thead>
422 <tbody id="tr_proveedores">
423 </tbody>
424 </table>
425 <button id="btnExportar" class="btn btn-success">
426 <i class="fas fa-file-excel"></i> Exportar datos a Excel
427 </button>
428 </div>
429 </div>
430 </div>
431 </div>
432 </div>
433 </div>
434 </div>
435 </section>

```

Figura. 309 Creación de la tabla de registros para proveedores, Fuente. Elaboración propia.

Para llenar la tabla de los registros se ocupa el archivo “proveedores.js”, como se muestra en la figura 310, el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros de cada insert para almacenarlos y mostrarlos en el orden indicado con anterioridad.

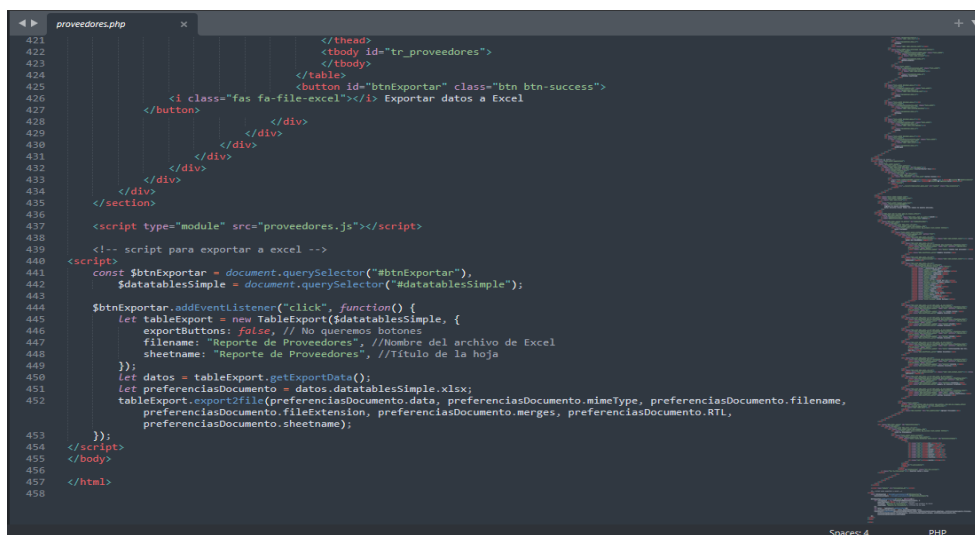
```

1 import {
2   initializeApp
3 } from "https://www.
4 import {
5   getDatabase,
6   ref,
7   get
8 } from "https://www.gstatic.
9
10 const firebaseConfig = {
11   apiKey: "AIzaSyDPz2KE3d",
12   authDomain: "ccpl-6456d",
13   databaseURL: "https://c",
14   projectId: "ccpl-6456d",
15   storageBucket: "ccpl-6456d",
16   messagingSenderId: "2666749276",
17   appId: "1:2666749276"
18 };
19
20 const app = initializeApp(firebaseConfig);
21 const database = getDatabase(app);
22
23 const proveedorTr = document.getElementById("tr_proveedores");
24 const paisSelect = document.getElementById("pais_select");
25
26 function CargarDatos() {
27   const proveedorRef = ref(database, "proveedores");
28
29   get(proveedorRef).then(snapshot => {
30     const proveedores = snapshot.val();
31     let rowNumber = 1;
32
33     for (const key in proveedores) {
34       if (Object.hasOwnProperty.call(proveedores, key)) {
35         const proveedor = proveedores[key];
36
37         const proveedorTable = document.createElement("tr");
38         proveedorTable.innerHTML = `
39 <tr>
40 <td><strong>${rowNumber}</strong></td>
41 <td>${proveedor.Nombre}</td>

```

Figura. 310 Conexión a la base de datos e incorporación de registros de la tabla proveedores, Fuente. Elaboración propia.

Como se muestra en la figura 311, dentro de la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, mediante el uso de un script colocado posteriormente del llamado “proveedores.js”.



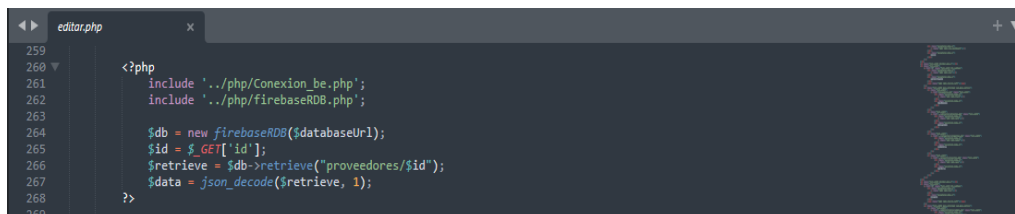
```

421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458

```

Figura. 311 Codificación y conversión de registros a formato Excel, Fuente. Elaboración propia.

En relación con el archivo proveedores.php se encuentra “editar.php” perteneciente a la misma carpeta, se encuentran las configuraciones iniciales de la barra de navegación, no obstante, se hace un llamado a la conexión de la base de datos dentro de la interpretación de php, se especifica la tabla a la cual se tendrá acceso de información, así como se observa en la figura 312.



```

259
260
261
262
263
264
265
266
267
268
269

```

Figura. 312 Creación y configuración del archivo editar.php. Fuente. Elaboración propia.

Posteriormente los inputs reciben una alteración de código, permitiendo que los datos almacenados con anterioridad en los campos designados sean llamados y visualizados dentro del mismo input y listas de selección, así como se observa en la figura 313, se conservan las reglas de patrones indicados por el archivo antecesor proveedores.php.

```

300 <input type="text" name="usuario" value="<?php echo $usuario?>" />
301 </div>
302 <div class="mdl-cell mdl-cell--12-col">
303 <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
304 <input class="mdl-textfield__input" type="text" pattern="[A-Za-z0-9
305 ]+(\.[0-9]+)?" name="Nombre" value="<?php echo $data['Nombre']?>" />
306 <label class="mdl-textfield__label" for="Nombre">Nombre del proveedor </label>
307 <span class="mdl-textfield__error">Nombre Invalido</span>
308 </div>
309 </div>
310 <div class="mdl-cell mdl-cell--12-col">
311 <legend class="text-condensed light"><i class="zmdi zmdi-border-color"></i> &nbsp; DIRECCION</legend>
312 </div>
313 <div class="mdl-cell mdl-cell--12-col">
314 <div class="mdl-textfield mdl-js-textfield">
315 <select class="mdl-textfield__input" name="Países" id="pais_select">
316 <option value="">Selecciona el País</option>
317 <option value="Alemania">Alemania</option>
318 <option value="Arabia Saudita">Arabia Saudita</option>
319 <option value="Argentina">Argentina</option>
320 <option value="Australia">Australia</option>
321 <option value="Brasil">Brasil</option>
322 <option value="Canadá">Canadá</option>
323 <option value="China">China</option>
324 <option value="Corea del Sur">Corea del Sur</option>
325 <option value="Estados Unidos">Estados Unidos</option>
326 <option value="Francia">Francia</option>
327 <option value="India">India</option>
328 <option value="Indonesia">Indonesia</option>
329 <option value="Italia">Italia</option>
330 <option value="Japón">Japón</option>
331 <option value="México">México</option>
332 <option value="Reino Unido">Reino Unido</option>
333 <option value="Rusia">Rusia</option>
334 <option value="Sudáfrica">Sudáfrica</option>
335 <option value="Turquía">Turquía</option>
336 <option value="Unión Europea">Unión Europea</option>
337 </select>
338 </div>

```

Figura. 313 Codificación y llamado de registros en los campos del formulario proveedor, Fuente. Elaboración propia.

Se coloca un botón permitiendo la redirección al archivo “action_editar” continuando con el proceso de actualización de datos del formulario, así como se muestra en la figura 314.

```

395 <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
396 mdl-button--colored bg-primary" id="btn-addProvider">
397 <i class="zmdi zmdi-plus"></i>
398 </button>
399 <div class="mdl-tooltip" for="btn-addProvider">Guardar Cambio</div>
400 </div>
401 </div>
402 </div>
403 </div>
404 </div>
405 </div>
406 <div class="mdl-tabs__panel" id="tabListProvider">
407

```

Figura. 314 Creación del botón guardar cambio, Fuente. Elaboración propia.

Al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad, como se muestra en la figura 315.

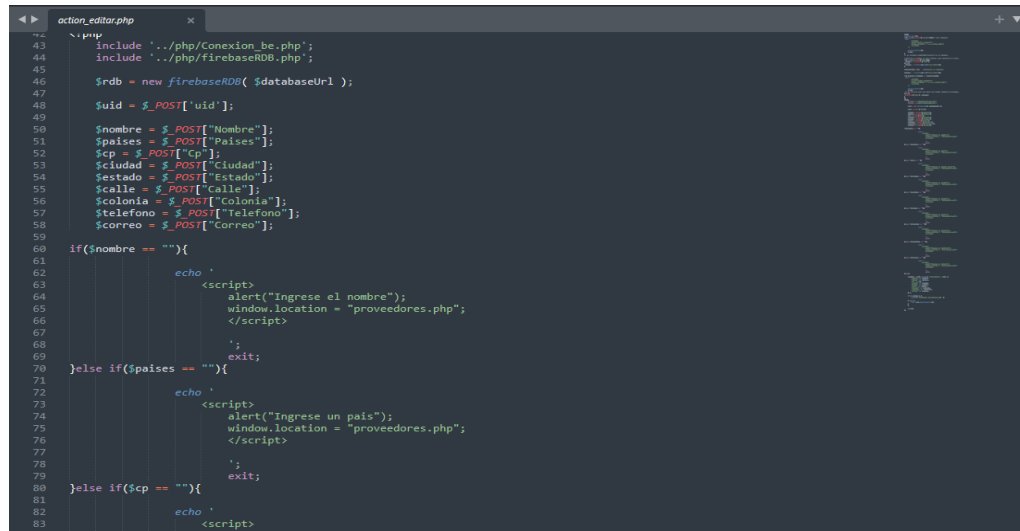
```

1 <?php
2 session_start();
3 if(isset($_SESSION['usuario'])){ // Esta logeado.
4 echo
5 <script>
6 alert("Inicio session");
7 window.location = "../index.html";
8 </script>
9
10 session_destroy();
11 die();
12
13 // La variable $_SESSION['usuario'] es un ejemplo.
14
15 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
16 if(isset($_SESSION['time'])){
17 $tiempo = $_SESSION['time'];
18 }else{
19 $tiempo = strtotime(date("Y-m-d H:i:s"));
20 }
21
22 $inactividad = 300; //Expresion en segundos.
23 $actual = strtotime(date("Y-m-d H:i:s"));
24

```

Figura. 315 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente. Elaboración propia.

Se hace el llamado a la conexión de la base de datos y creación de variables que almacenaran los cambios indicados en los input y listas, así como se muestra en la figura 316, a su vez se verifica que dichos inputs contengan información ya que al no contenerla se coloca una alerta y redireccionamiento al archivo “proveedores.php”.



```

43 include '../php/Conexion_be.php';
44 include '../php/firebaseRDB.php';
45
46 $rdb = new firebaseRDB( $databaseUrl );
47
48 $uid = $_POST['uid'];
49
50 $nombre = $_POST['Nombre'];
51 $países = $_POST['Países'];
52 $cp = $_POST['cp'];
53 $ciudad = $_POST['Ciudad'];
54 $estado = $_POST['Estado'];
55 $calle = $_POST['Calle'];
56 $colonia = $_POST['Colonia'];
57 $telefono = $_POST['Telefono'];
58 $correo = $_POST['Correo'];
59
60 if($nombre == ""){
61     echo '
62     <script>
63         alert("Ingrese el nombre");
64         window.location = "proveedores.php";
65     </script>
66     ';
67     exit;
68 }
69
70 }else if($países == ""){
71     echo '
72     <script>
73         alert("Ingrese un país");
74         window.location = "proveedores.php";
75     </script>
76     ';
77     exit;
78 }
79
80 }else if($cp == ""){
81     echo '
82     <script>
83         alert("Ingrese el código postal");
84         window.location = "proveedores.php";
85     </script>
86     ';
87     exit;
88 }
89
90 }else if($ciudad == ""){
91     echo '
92     <script>
93         alert("Ingrese la ciudad");
94         window.location = "proveedores.php";
95     </script>
96     ';
97     exit;
98 }
99
100 }else if($estado == ""){
101     echo '
102     <script>
103         alert("Ingrese el estado");
104         window.location = "proveedores.php";
105     </script>
106     ';
107     exit;
108 }
109
110 }else if($calle == ""){
111     echo '
112     <script>
113         alert("Ingrese la calle");
114         window.location = "proveedores.php";
115     </script>
116     ';
117     exit;
118 }
119
120 }else if($colonia == ""){
121     echo '
122     <script>
123         alert("Ingrese la colonia");
124         window.location = "proveedores.php";
125     </script>
126     ';
127     exit;
128 }
129
130 }else if($telefono == ""){
131     echo '
132     <script>
133         alert("Ingrese el teléfono");
134         window.location = "proveedores.php";
135     </script>
136     ';
137     exit;
138 }
139
140 }else if($correo == ""){
141     echo '
142     <script>
143         alert("Ingrese el correo");
144         window.location = "proveedores.php";
145     </script>
146     ';
147     exit;
148 }
149
150 }else{
151     $update = $rdb->update( "productos", $uid, [
152         "Nombre" => $nombre,
153         "Categoría" => $categoría,
154         "Proyecto" => $proyecto,
155         "Proveedor" => $proveedor,
156         "Ubicación" => $ubicación,
157         "Fecha" => $fecha,
158     ] );
159
160     if ( $update ) {
161         header( "location: productos.php" );
162     } else {
163         echo $rdb->getLastError();
164     }
165
166     exit();
167 }
168
169 ?>
    
```

Figura. 316 Conexión a la base de datos y verificación de campos, Fuente. Elaboración propia.

En cambio, como se muestra en la figura 31

7, el proceso de ingreso de información, se anula la evaluación de datos repetidos y se comienza el proceso de actualización de datos, el cual una vez finalizado dará redireccionamiento al archivo proveedores.php.



```

110
111 echo '
112 <script>
113     alert("Ingrese la fecha");
114     window.location = "productos.php";
115 </script>
116 ';
117 exit;
118 }
119
120 }else{
121     $update = $rdb->update( "productos", $uid, [
122         "Nombre" => $nombre,
123         "Categoría" => $categoría,
124         "Proyecto" => $proyecto,
125         "Proveedor" => $proveedor,
126         "Ubicación" => $ubicación,
127         "Fecha" => $fecha,
128     ] );
129
130     if ( $update ) {
131         header( "location: productos.php" );
132     } else {
133         echo $rdb->getLastError();
134     }
135
136     exit();
137 }
138
139 ?>
    
```

Figura. 317 Codificación de ingreso de datos a la tabla productos, Fuente. Elaboración propia.

El llenado de la tabla de los registros se ocupa el archivo “proveedores.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros que fueron actualizados y mostrarlos en el orden indicado con anterioridad, como se observa en la figura 318.

```

397 <div class="mdl-tabs_panel" id="tabListProvider">
398 <div class="mdl-grid">
399 <div class="mdl-cell mdl-cell--12-col">
400 <div class="full-width panel mdl-shadow--2dp">
401 <div class="full-width panel-title bg-success text-center titles">
402 Lista de Proveedores
403 </div>
404 <div class="full-width panel-content">
405 <div class="table-responsive text-center">
406 <table class="table table-bordered table-hover" id="datatablesSimple">
407 <thead>
408 <tr>
409 <th scope="col"><strong>ID</strong></th>
410 <th scope="col"><strong>Nombre</strong></th>
411 <th scope="col"><strong>País</strong></th>
412 <th scope="col"><strong>CP</strong></th>
413 <th scope="col"><strong>Ciudad</strong></th>
414 <th scope="col"><strong>Estado</strong></th>
415 <th scope="col"><strong>Calle</strong></th>
416 <th scope="col"><strong>Colonia</strong></th>
417 <th scope="col"><strong>Correo</strong></th>
418 <th scope="col"><strong>Telefono</strong></th>
419 <th scope="col"><strong>Acción</strong></th>
420 </tr>
421 </thead>
422 <tbody id="tr_proveedores">
423 </tbody>
424 </table>
425 <button id="btnExportar" class="btn btn-success">
426 <i class="fas fa-file-excel"></i> Exportar datos a Excel
427 </button>

```

Figura. 318 Creación de la tabla de registros actualizados para proveedores. Fuente. Elaboración propia.

Por otra parte, como se muestra en la figura 319, se encuentra el archivo “action_eliminar” ubicado dentro de la misma carpeta de administración, Al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad.

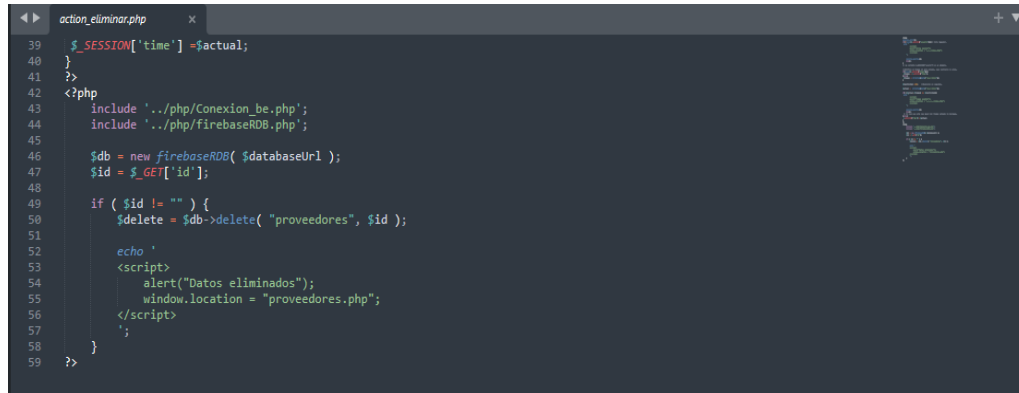
```

1 <?php
2 session_start();
3 if(isset($_SESSION['usuario'])){// Esta logeado?.
4 echo '
5 <script>
6 alert("Inicie session");
7 window.location = ".../index.html";
8 </script>
9 '
10
11 session_destroy();
12 die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18 $tiempo = $_SESSION['time'];
19 }else{
20 $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Exprecion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28 echo '
29 <script>
30 alert("Tiempo excedido");
31 window.location = ".../index.html";
32 </script>
33 '
34
35 session_destroy();
36 die();
37 // En caso que este sea mayor del tiempo seteado lo desloga.
38 }else{
39 $_SESSION['time'] = $actual;
40 }
41 }
42 <?php

```

Figura. 319 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente, Elaboración propia.

Se hace el llamado a la conexión de la base de datos y se comienza el borrado de registro mediante una sentencia de evaluación que hace uso del identificador del registro y la variable delete, como se observa en la figura 320, una vez finalizada la eliminación se coloca una alerta sobre la eliminación y se redirecciona al archivo “proveedores.php”.



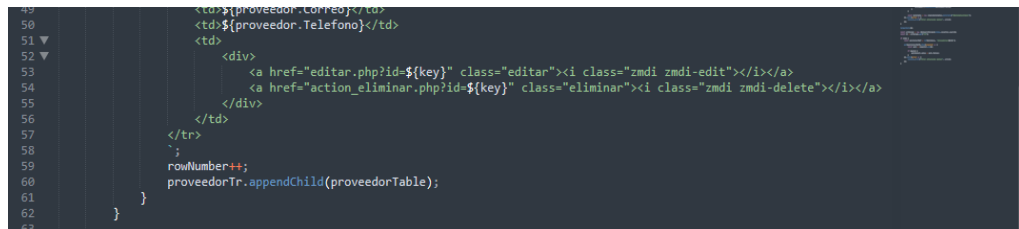
```

39  $_SESSION['time'] = $actual;
40  }
41  }
42  }
43  <?php
44  include '../php/Conexion_be.php';
45  include '../php/firebaseRDB.php';
46
47  $db = new firebaseRDB( $databaseUrl );
48  $id = $_GET['id'];
49
50  if ( $id != "" ) {
51      $delete = $db->delete( "proveedores", $id );
52
53      echo '
54      <script>
55          alert("Datos eliminados");
56          window.location = "proveedores.php";
57      </script>
58      ';
59  }
60  }

```

Figura. 320 Codificación y eliminación de registros de la tabla proveedores, Fuente. Elaboración propia.

Se muestra en la figura 321 que, para que las funciones de editar y eliminar empiecen su proceso, dentro de la tabla de visualización de registros se visualizan 2 botones de acción para cada proceso correspondiente, dichos botones son ubicados en el archivo “proveedores.js”.



```

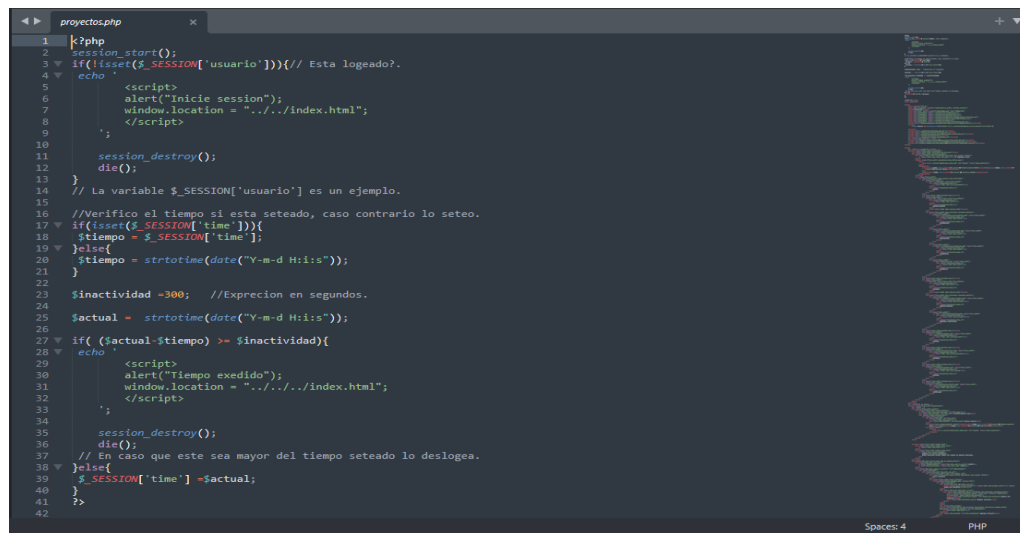
49  <td>{proveedor.Nombre}</td>
50  <td>{proveedor.Telefono}</td>
51  <td>
52  <div>
53  <a href="editar.php?id={key}" class="editar"><i class="zmdi zmdi-edit"></i></a>
54  <a href="action_eliminar.php?id={key}" class="eliminar"><i class="zmdi zmdi-delete"></i></a>
55  </div>
56  </td>
57  </tr>
58  ';
59  rowNumber++;
60  proveedorTr.appendChild(proveedorTable);
61  }
62  }
63  }

```

Figura. 321 Codificación de vínculos para los botones eliminar y editar, Fuente. Elaboración propia.

3.10.11. Proyectos:

Por parte del archivo “proyectos.php” se especifica en lenguaje de php mediante sentencias de decisión donde se evalúa si una vez en la página de home.php se autentico de manera correcta, en caso de no ser así se coloca una alerta y retornara a index.html, como se observa en la figura 322. A si mismo evalúa el tiempo de inactividad entre navegación de archivos teniendo un conteo de 300 segundos de inactividad. Si el conteo es completado se retornará a index.html de manera forzada, en caso de navegar entre archivos el conteo se reiniciará de manera automática, culminando con el cierre del lenguaje de PHP.



```

1 <?php
2 session_start();
3 if(!isset($_SESSION['usuario'])){// Esta logeado?.
4     echo '
5     <script>
6     alert("Inicie session");
7     window.location = "../index.html";
8     </script>
9     '
10    session_destroy();
11    die();
12 }
13 // La variable $_SESSION['usuario'] es un ejemplo.
14
15 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
16 if(isset($_SESSION['time'])){
17     $tiempo = $_SESSION['time'];
18 }else{
19     $tiempo = strtotime(date("Y-m-d H:i:s"));
20 }
21
22 $inactividad =300; //Expresion en segundos.
23
24 $actual = strtotime(date("Y-m-d H:i:s"));
25
26 if( ($actual-$tiempo) >= $inactividad){
27     echo '
28     <script>
29     alert("Tiempo excedido");
30     window.location = "../index.html";
31     </script>
32     '
33     session_destroy();
34     die();
35 // En caso que este sea mayor del tiempo seteado lo deslogea.
36 }else{
37     $_SESSION['time'] =$actual;
38 }
39 }
40 }
41 }
42
  
```

Figura. 322 Configuración de lenguaje y tiempo de inactividad del archivo proyectos.php. Fuente. Elaboración propia.

Posteriormente como se observa en la figura 323, se coloca la estructura de !DOCTYPE html> la cual informara al navegador el tipo de versión HTML que fue implementado en el archivo, a su vez se indica con una etiqueta de tipo lang el tipo de idioma con el que será escrita la página web en este caso se hace uso del idioma español. A su vez se muestra la apertura de un head en el cual se proveerá la información como lo es el título y lo enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assets” y de las subcarpetas de “img”, “css” y “js”.

```

43 <!DOCTYPE html>
44 <html lang="es">
45
46 <head>
47   <meta charset="UTF-8">
48   <meta name="viewport" content="width=device-width, initial-scale=1">
49   <title>Personal</title>
50   <link rel="icon" href="../../assets/img/logo.png" type="image/png">
51   <link rel="stylesheet" href="../../assets/css/normalize.css">
52   <link rel="stylesheet" href="../../assets/css/sweetalert2.css">
53   <link rel="stylesheet" href="../../assets/css/material.min.css">
54   <link rel="stylesheet" href="../../assets/css/material-design-iconic-font.min.css">
55   <link rel="stylesheet" href="../../assets/css/jquery.mCustomScrollbar.css">
56   <link rel="stylesheet" href="../../assets/css/main.css">
57   <link rel="stylesheet" href="../../assets/css/simple-datatables.min.css">
58   <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
59   <script>
60     window.jQuery || document.write('<script src="../../assets/js/jquery-1.11.2.min.js"></script>')
61   </script>
62   <script src="../../assets/js/material.min.js"></script>
63   <script src="../../assets/js/sweetalert2.min.js"></script>
64   <script src="../../assets/js/jquery.mCustomScrollbar.concat.min.js"></script>
65   <script src="../../assets/js/marco2.js"></script>
66   <script src="../../assets/js/simple-datatables.js"></script>
67   <!-- links para exportar a excel -->
68   <script src="https://unpkg.com/xlsx@0.16.9/dist/xlsx.full.min.js"></script>
69   <script src="https://unpkg.com/file-saverjs@latest/FileSaver.min.js"></script>
70   <script src="https://unpkg.com/tableexport@latest/dist/js/tableexport.min.js"></script>
71 </head>

```

Figura. 323 Estructura de lenguaje HTML e Idioma del archivo proyectos.php, Fuente. Elaboración propia.

Una vez definido el encabezado de la página mediante la etiqueta head, como se observa en la figura 324, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web. Posteriormente se coloca un elemento que ayudara a representar el contenido principal de la página web que es el “body”. Posteriormente se abre una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, se colocan contenedores para aplicar estilos correspondientes a las sentencias, se agregan imágenes que harán la función del avatar del administrador, así como el nombre del usuario administrador que se autentico con anterioridad los datos son llamados desde la base de datos de Firebase.

```

72 <body>
73   <!-- barra de navegacion lateral -->
74   <section class="full-width navLateral">
75     <div class="full-width navLateral-bg btn-menu"></div>
76     <div class="full-width navLateral-body">
77       <div class="full-width navLateral-body-logo text-center tittles">
78         <i class="zmdi zmdi-close btn-menu"></i> CC Camaleon Color
79       </div>
80       <figure class="full-width navLateral-body-tittle-menu">
81         <div>
82           
83         </div>
84         <figcaption>
85           <span><span><?php echo $_SESSION['usuario'];></span></span><?php echo $_SESSION['NombreCompleto'];></span><?php echo $_SESSION['Apellido'];></span></span>
86           <br>
87           <span><small><?php echo $_SESSION['usuario'];></small></span>
88         </figcaption>
89       </figure>
90     </div>

```

Figura. 324 Codificación del nombre y avatar del usuario, Fuente. Elaboración propia.

Se coloca una etiqueta nav para proporcionar una navegación estructurada en la página, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta assets/css, así como se observa en las figuras 325, 326 y 327,328.

```

91 <nav class="full-width">
92 <ul class="full-width list-unstyled menu-principal">
93 <li class="full-width">
94 <a href=".."home.php" class="full-width">
95 <div class="navLateral-body-cl">
96 <i class="zmdi zmdi-view-dashboard"></i>
97 </div>
98 <div class="navLateral-body-cr">
99 INICIO
100 </div>
101 </a>
102 </li>
103 <li class="full-width divider-menu-h"></li>
104 <li class="full-width">
105 <a href="#" class="full-width btn-subMenu">
106 <div class="navLateral-body-cl">
107 <i class="zmdi zmdi-case"></i>
108 </div>
109 <div class="navLateral-body-cr">
110 ADMINISTRACION
111 </div>
112 <span class="zmdi zmdi-chevron-left"></span>
113 </a>
114 <ul class="full-width menu-principal sub-menu-options">
115 <li class="full-width">
116 <a href=".."proveedores/proveedores.php" class="full-width">
117 <div class="navLateral-body-cl">
118 <i class="zmdi zmdi-truck"></i>
119 </div>
120 <div class="navLateral-body-cr">
121 PROVEEDORES
122 </div>
123 </a>
124 </li>
125 <li class="full-width">
126 <a href=".."ubicacion/ubicacion.php" class="full-width">
127 <div class="navLateral-body-cl">
128 <i class="zmdi zmdi-card"></i>
129 </div>
130 <div class="navLateral-body-cr">
131 UBICACIONES
132 </div>

```

Figura. 325 Configuración de la barra de navegación lateral pt1, Fuente. Elaboración propia.

```

129 </div>
130 <div class="navLateral-body-cr">
131 UBICACIONES
132 </div>
133 </a>
134 </li>
135 <li class="full-width">
136 <a href=".."categorias/categorias.php" class="full-width">
137 <div class="navLateral-body-cl">
138 <i class="zmdi zmdi-label"></i>
139 </div>
140 <div class="navLateral-body-cr">
141 CATEGORIAS
142 </div>
143 </a>
144 </li>
145 <li class="full-width">
146 <a href=".."proyecto/proyectos.php" class="full-width">
147 <div class="navLateral-body-cl">
148 <i class="zmdi zmdi-case"></i>
149 </div>
150 <div class="navLateral-body-cr">
151 PROYECTOS
152 </div>
153 </a>
154 </li>
155 </ul>
156 </li>
157 <li class="full-width divider-menu-h"></li>
158 <li class="full-width">
159 <a href="#" class="full-width btn-subMenu">
160 <div class="navLateral-body-cl">
161 <i class="zmdi zmdi-face"></i>
162 </div>
163 <div class="navLateral-body-cr">
164 USUARIOS
165 </div>
166 <span class="zmdi zmdi-chevron-left"></span>
167 </a>
168 <ul class="full-width menu-principal sub-menu-options">
169 <li class="full-width">
170 <a href="admin.php" class="full-width">

```

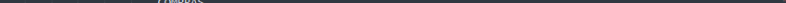
Figura. 326 Configuración de la barra de navegación lateral pt2, Fuente. Elaboración propia.

```

178 </li>
179 <li class="full-width">
180 <a href="../personal/personal.php" class="full-width">
181 <div class="navLateral-body-cl">
182 <i class="zmdi zmdi-accounts"></i>
183 </div>
184 <div class="navLateral-body-cr">
185 PERSONAL CALIFICADO
186 </div>
187 </a>
188 </li>
189 </ul>
190 </li>
191 <li class="full-width divider-menu-h"></li>
192 <li class="full-width">
193 <a href="../compras/compras.php" class="full-width">
194 <div class="navLateral-body-cl">
195 <i class="zmdi zmdi-shopping-cart"></i>
196 </div>
197 <div class="navLateral-body-cr">
198 COMPRAS
199 </div>
200 </a>
201 </li>
202 <li class="full-width divider-menu-h"></li>
203 <li class="full-width">
204 <a href="../productos/productos.php" class="full-width">

```

Spaces: 4 PHP

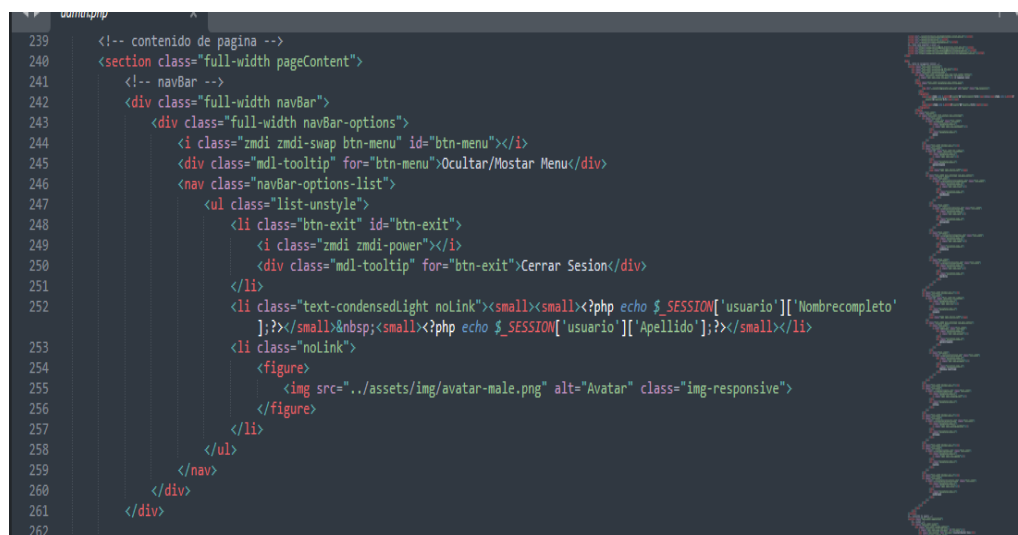


```

207     <li class="full-width divider-menu-h"></li>
208     </li>
209     </div>
210     </a>
211     </li>
212     <li class="full-width divider-menu-h"></li>
213     </li>
214     <li class="full-width">
215         <a href=".."salidas/salidas.php" class="full-width">
216             <div class="navLateral-body-cl">
217                 <i class="zmdi zmdi-view-agenda"></i>
218             </div>
219             <div class="navLateral-body-cr">
220                 SALIDAS
221             </div>
222         </a>
223     </li>
224     <li class="full-width divider-menu-h"></li>
225     </li>
226     <li class="full-width">
227         <a href=".."inventario/inventario.php" class="full-width">
228             <div class="navLateral-body-cl">
229                 <i class="zmdi zmdi-store"></i>
230             </div>
231             <div class="navLateral-body-cr">
232                 INVENTARIO
233             </div>
234         </a>
235     </li>
236 </ul>
237 </nav>
238 </div>
239 </section>
240 <!-- container da pagina -->

```

Por parte del contenido de la página web como se observa en la figura 329, se coloca una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, la cual contendrá contenedores para aplicar estilos con css, además de continuar con las animaciones de la barra de navegación, se presenta la configuración para ocultar y mostrar la barra de navegación. Se programa la interfaz de cierre de sesión mediante la implementación de un botón con identificador. Posteriormente se coloca una imagen con el avatar de identificación del administrador y se manda a llamar los datos propios de este, desde la base de datos Firebase.



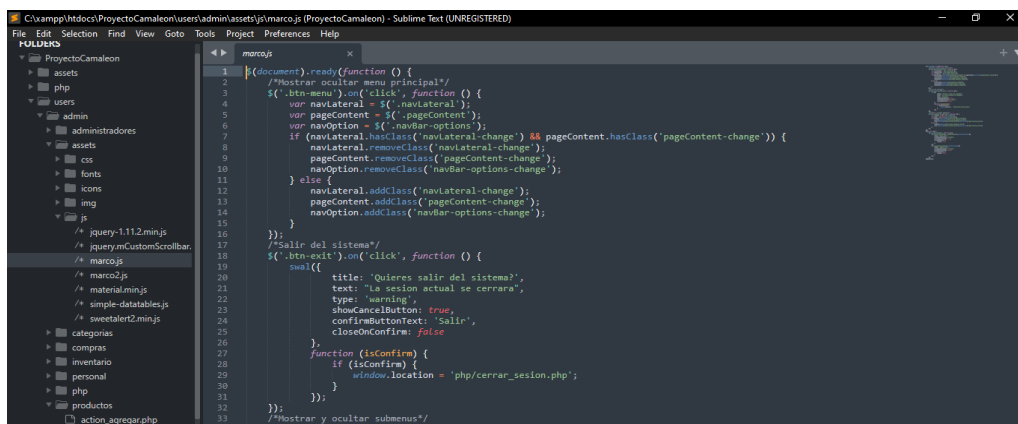
```

239 <!-- contenido de pagina -->
240 <section class="full-width pageContent">
241 <!-- navbar -->
242 <div class="full-width navbar">
243 <div class="full-width navbar-options">
244 <i class="zmdi zmdi-swap btn-menu" id="btn-menu"></i>
245 <div class="mdl-tooltip" for="btn-menu">Ocultar/Mostrar Menu</div>
246 <nav class="navbar-options-list">
247 <ul class="list-unstyled">
248 <li class="btn-exit" id="btn-exit">
249 <i class="zmdi zmdi-power"></i>
250 <div class="mdl-tooltip" for="btn-exit">Cerrar Sesión</div>
251 </li>
252 <li class="text-condensedLight noLink"><small><?php echo $_SESSION['usuario']['NombreCompleto']></small><small><?php echo $_SESSION['usuario']['Apellido']></small></li>
253 <li class="noLink">
254 <figure>
255 
256 </figure>
257 </li>
258 </ul>
259 </nav>
260 </div>
261 </div>
262

```

Figura. 329 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Las animaciones correspondientes al cierre de sesión y ocultar/mostrar la barra de navegación son estructuradas en el archivo “marco.js” ubicado en la carpeta assets/js, como se muestran en la figura 330.



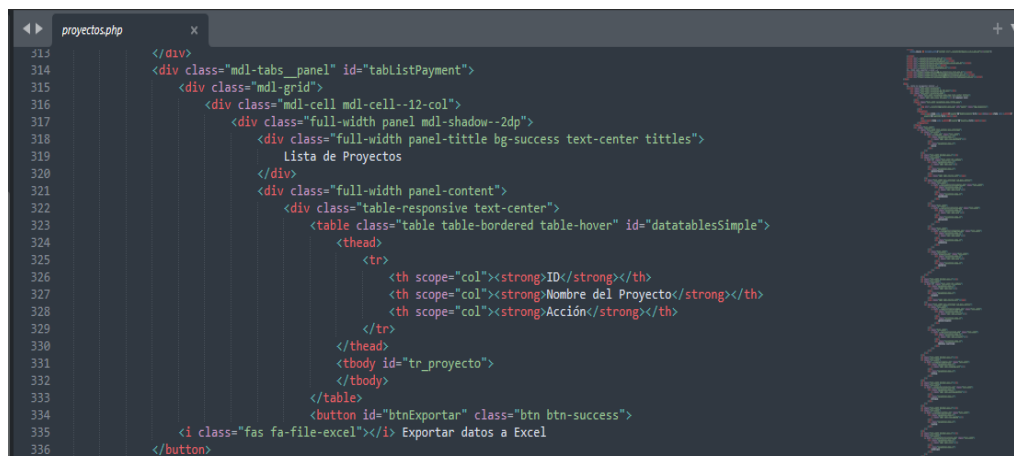
```

1 (document).ready(function () {
2 //Mostrar ocultar menu principal/
3 $('btn-menu').on('click', function () {
4 var navLateral = $('#navLateral');
5 var pageContent = $('#pageContent');
6 var navOption = $('#navBar-options');
7 if (navLateral.hasClass('navLateral-change') || pageContent.hasClass('pageContent-change')) {
8 navLateral.removeClass('navLateral-change');
9 pageContent.removeClass('pageContent-change');
10 navOption.removeClass('navBar-options-change');
11 } else {
12 navLateral.addClass('navLateral-change');
13 pageContent.addClass('pageContent-change');
14 navOption.addClass('navBar-options-change');
15 }
16 });
17 //Salir del sistema/
18 $('btn-exit').on('click', function () {
19 swal({
20 title: 'Quieres salir del sistema?',
21 text: 'La sesion actual se cerrara',
22 type: 'warning',
23 showCancelButton: true,
24 confirmButtonText: 'Salir',
25 closeOnConfirm: false
26 });
27 function (isConfirm) {
28 if (isConfirm) {
29 window.location = 'php/cerrar_sesion.php';
30 }
31 });
32 });
33 //Mostrar y ocultar submenus/
34

```

Figura. 330 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Así mismo como se observa en la figura 334, la creación de nuevos contenedores que almacenaran los inputs necesarios para el registro de información.



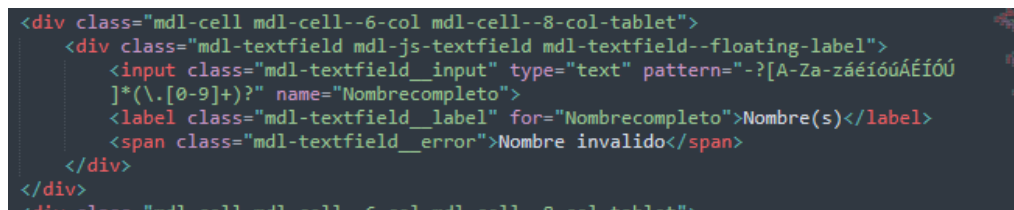
```

313 </div>
314 <div class="mdl-tabs_panel" id="tabListPayment">
315   <div class="mdl-grid">
316     <div class="mdl-cell mdl-cell--12-col">
317       <div class="full-width panel mdl-shadow--2dp">
318         <div class="full-width panel-title bg-success text-center titles">
319           Lista de Proyectos
320         </div>
321         <div class="full-width panel-content">
322           <div class="table-responsive text-center">
323             <table class="table table-bordered table-hover" id="datatablesSimple">
324               <thead>
325                 <tr>
326                   <th scope="col"><strong>ID</strong></th>
327                   <th scope="col"><strong>Nombre del Proyecto</strong></th>
328                   <th scope="col"><strong>Acción</strong></th>
329                 </tr>
330               </thead>
331               <tbody id="tr_proyecto">
332               </tbody>
333             </table>
334             <button id="btnExportar" class="btn btn-success">
335               <i class="fas fa-file-excel"></i> Exportar datos a Excel
336             </button>

```

Figura. 334 Creación de los campos para el formulario de proyectos, Fuente. Elaboración propia.

Siendo de esta manera que la estructura de cada input se definirá por el tipo de dato a almacenar, ya sea numérico, texto, correo o contraseña. Se definen con la expresión regular de pattern para indicarle el tipo de simbología que deberá de contener, como se observa en la figura 335, esto no aplica para los campos de correo y contraseña ya que poseen una estructura específica, a su vez se coloca el formato las listas de selección las cuales pueden contener información predestinada mediante etiquetas “option” o en su caso incluir un archivo js implementando información desde la base de datos. En caso de incumplir dicha estructura se colocará abajo del input una alerta de texto erróneo.



```

<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
  <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
    <input class="mdl-textfield__input" type="text" pattern="-?[A-Za-záéíóúÁÉÍÓÚ]*(\.[0-9]+)?" name="Nombrecompleto">
    <label class="mdl-textfield__label" for="Nombrecompleto">Nombre(s)</label>
    <span class="mdl-textfield__error">Nombre invalido</span>
  </div>
</div>
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">

```

Figura. 335 Codificación de la estructura de los campos para el formulario. Fuente. Elaboración propia.

Tal y como se muestra en la figura 336, al finalizar las inserciones de los input se crea un botón dentro de un contenedor que comenzara el procesamiento de información del formulario.

```

333 </table>
334 <button id="btnExportar" class="btn btn-success">
335 <i class="fas fa-file-excel"></i> Exportar datos a Excel
336 </button>
337 </div>
338 </div>
339 </div>
340 </div>
341 </div>
342 </div>
343 </div>
344 </section>
345
346 <script type="module" src="proyecto.js"></script>
347
348 <!-- script para exportar a excel -->
349 <script>
350 const $btnExportar = document.querySelector("#btnExportar"),
351 $datatablesSimple = document.querySelector("#datatablesSimple");
352
353 $btnExportar.addEventListener("click", function() {
354 let tableExport = new TableExport($datatablesSimple, {
355 exportButtons: false, // No queremos botones
356 filename: "Reporte de Proyectos", //Nombre del archivo de Excel
357 sheetname: "Reporte de Proyectos", //Título de la hoja
358 });
359 let datos = tableExport.getData();
360 let preferenciasDocumento = datos.datatablesSimple.xlsx;
361 tableExport.export2file(preferenciasDocumento.data, preferenciasDocumento.mimeType, preferenciasDocumento.filename, preferenciasDocumento.fileExtension, preferenciasDocumento.merges, preferenciasDocumento.RTL, preferenciasDocumento.sheetname);
362 });
363 </script>

```

Figura. 336 Creación del botón agregar proyecto, Fuente. Elaboración propia.

Se observa en la figura 337 que, para que el procesamiento de datos este completo el uso del archivo “action_agregar.php” cumple con la función de verificar el tiempo de inactividad dentro de la página y almacenar los datos obtenidos de los inputs y listas de selección en variables, las cuales serán evaluadas mediante sentencias condicionales identificando si la variable recibió información o esta es nula, si a información obtenida resulta ser nula se regresara al archivo “proyectos.php”.

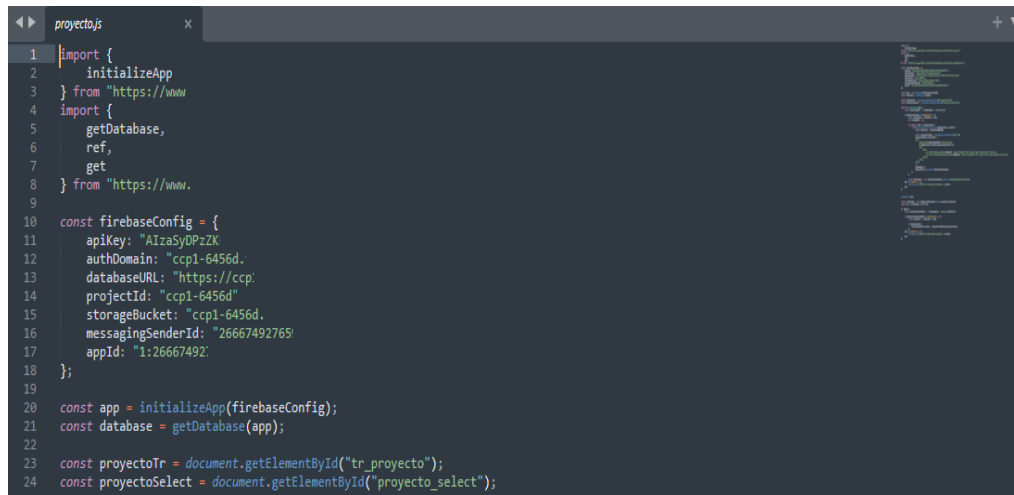
```

File Edit Selection Find View Goto Tools Project Preferences Help
1 <?php
2 session_start();
3 if(isset($_SESSION['usuario'])){// Esta logeado?.
4 echo '
5 <script>
6 alert("Inicio session");
7 window.location = "../index.html";
8 </script>
9 '
10
11 session_destroy();
12 die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18 $tiempo = $_SESSION['time'];
19 }else{
20 $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Expresion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28 echo '
29 <script>
30 alert("Tiempo excedido");
31 window.location = "../index.html";
32 </script>
33 '
34
35 session_destroy();
36 die();
37 // En caso que este sea mayor del tiempo seteado lo deslogea.
38 }else{
39 $_SESSION['time'] = $actual;
40 }
41 }
42 <?php

```

Figura. 337 Codificación de tiempo de inactividad del archivo action_agregar.php. Fuente, Elaboración propia.

Reflejado en la figura 338 muestra que dentro del archivo proveedores.js se hace la conexión a la base de datos y se mandan llamar a las tablas necesarias para cargar y almacenar la información en las listas mediante el uso de variables. Por lo tanto, si la información es correcta se mandará a llamar a las tablas de proyectos.



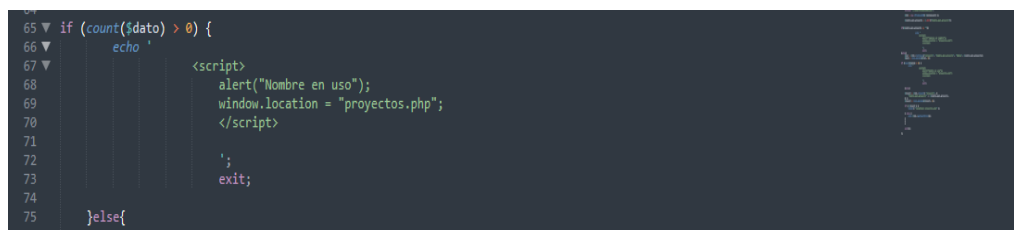
```

1  import {
2    initializeApp
3  } from "https://www
4  import {
5    getDatabase,
6    ref,
7    get
8  } from "https://www
9
10 const firebaseConfig = {
11   apiKey: "AIzaSyUPzZK
12   authDomain: "ccp1-6456d
13   databaseURL: "https://ccp
14   projectId: "ccp1-6456d"
15   storageBucket: "ccp1-6456d
16   messagingSenderId: "26667492765
17   appId: "1:26667492
18 };
19
20 const app = initializeApp(firebaseConfig);
21 const database = getDatabase(app);
22
23 const proyectoTr = document.getElementById("tr_proyecto");
24 const proyectoSelect = document.getElementById("proyecto_select");

```

Figura. 338 Conexión a la base de datos e incorporación de registros en listas del formulario, Fuente. Elaboración propia.

Por consiguiente, como se observa en la figura 339, si la información es correcta se verificará por medio de sentencias de decisión si es que los datos proporcionados no están almacenados dentro de las tablas de la base de datos, si están almacenados por medio de un script regresará una alerta y redirigirá al archivo proyectos.php.



```

65 if (count($dato) > 0) {
66   echo '
67   <script>
68     alert("Nombre en uso");
69     window.location = "proyectos.php";
70   </script>
71   '
72   exit;
73 }else{
74
75

```

Figura. 339 Codificación de verificación para datos repetidos de los campos. Fuente. Elaboración propia.

Por otro lado, en caso de no estar almacenados, se insertará por medio de una variable que almacenará los datos correctos y devolverá al archivo proyectos.php, así como se muestra en la figura 340.

```

77 $insert = $rdb->insert( "proyecto", [
78     "Nombre_del_proyecto" => $nombre_del_proyecto,
79 ] );
80 $result = json_decode($insert, 1);
81
82 if ( $insert ) {
83     header( "Location: proyectos.php" );
84 } else {
85     echo $rdb->getLastError();
86 }
87 }
88 }
89 }
90
91 exit();

```

Figura. 340 Codificación de ingreso de datos a la tabla proyectos, Fuente. Elaboración propia.

Una vez realizado el ingreso de información se coloca un contenedor el cual almacenara el título de la lista de registros y se da formato a la tabla de registros con los campos acorde a los inputs y registros de la base de datos, como se muestra en la figura 341, se asignan identificadores para dar formato mediante css y hacer uso de Java Script.

```

321 <div class="full-width panel-content">
322 <div class="table-responsive text-center">
323 <table class="table table-bordered table-hover" id="datatablesSimple">
324 <thead>
325 <tr>
326 <th scope="col"><strong>ID</strong></th>
327 <th scope="col"><strong>Nombre del Proyecto</strong></th>
328 <th scope="col"><strong>Acción</strong></th>
329 </tr>
330 </thead>
331 <tbody id="tr_proyecto">
332 </tbody>
333 </table>
334 <button id="btnExportar" class="btn btn-success">
335 <i class="fas fa-file-excel"></i> Exportar datos a Excel
336 </button>

```

Figura. 341 Creación de la tabla de registros para proyectos. Fuente. Elaboración propia.

Para llenar la tabla de los registros se ocupa el archivo “proyectos.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros de cada insert para almacenarlos y mostrarlos en el orden indicado con anterioridad, así como se observa en la figura 32.

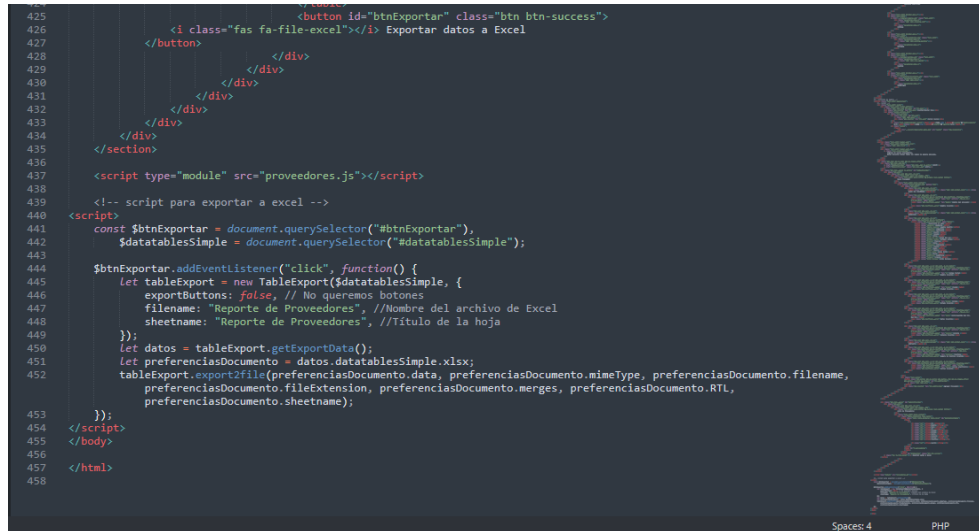
```

1 import {
2   initializeApp
3 } from "https://www
4 import {
5   getDatabase,
6   ref,
7   get
8 } from "https://www
9
10 const firebaseConfig = {
11   apiKey: "AIzaSy0Pz2KE
12   authDomain: "ccpi-6456d
13   databaseURL: "https://cc
14   projectId: "ccpi-6456d",
15   storageBucket: "ccpi-6456d
16   messagingSenderId: "2666749270",
17   appId: "1:266674927
18 };
19
20 const app = initializeApp(firebaseConfig);
21 const database = getDatabase(app);
22
23 const proveedorRef = document.getElementById("tr_proveedores");
24 const paisSelect = document.getElementById("pais_select");
25
26 function CargarDatos() {
27   const proveedorRef = ref(database, "proveedores");
28
29   get(proveedorRef).then(snapshot => {
30     const proveedores = snapshot.val();
31     let rowNumber = 1;
32
33     for (const key in proveedores) {
34       if (Object.hasOwnProperty.call(proveedores, key)) {
35         const proveedor = proveedores[key];
36
37         const proveedorTable = document.createElement("tr");
38         proveedorTable.innerHTML = "
39         <td><strong>${rowNumber}</strong></td>
40         <td>${proveedor.Nombre}</td>

```

Figura. 342 Conexión a la base de datos e incorporación de registros en la tabla proyectos, Fuente. Elaboración propia.

De ahí que como se muestra en la figura 343, dentro de la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, mediante el uso de un script colocado posteriormente del llamado “proyectos.js”.

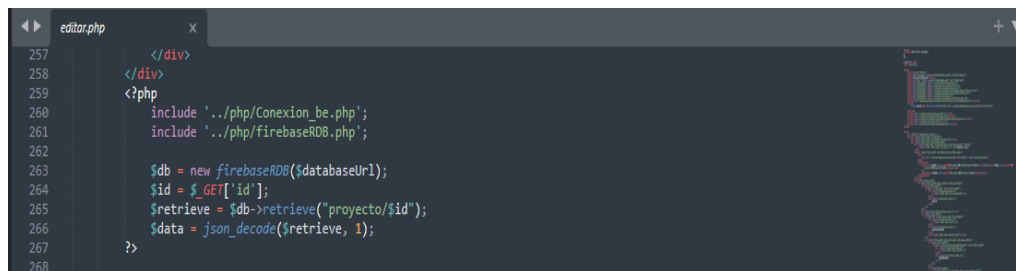


```

425         <button id="btnExportar" class="btn btn-success">
426             <i class="fas fa-file-excel"></i> Exportar datos a Excel
427         </button>
428     </div>
429 </div>
430 </div>
431 </div>
432 </div>
433 </div>
434 </div>
435 </section>
436
437 <script type="module" src="proveedores.js"></script>
438
439 <!-- script para exportar a excel -->
440 <script>
441     const $btnExportar = document.querySelector("#btnExportar");
442     $datatablesSimple = document.querySelector("#datatablesSimple");
443
444     $btnExportar.addEventListener("click", function() {
445         let tableExport = new TableExport($datatablesSimple, {
446             exportButtons: false, // No queremos botones
447             filename: "Reporte de Proveedores", //Nombre del archivo de Excel
448             sheetname: "Reporte de Proveedores", //Título de la hoja
449         });
450         let datos = tableExport.getExportData();
451         let preferenciasDocumento = datos.datatablesSimple.xlsx;
452         tableExport.export2File(preferenciasDocumento.data, preferenciasDocumento.mimeType, preferenciasDocumento.filename,
453             preferenciasDocumento.fileExtension, preferenciasDocumento.merges, preferenciasDocumento.RTL,
454             preferenciasDocumento.sheetname);
455     });
456 </script>
457 </body>
458 </html>
    
```

Figura. 343 Codificación y conversión de registros a formato Excel, Fuente. Elaboración propia.

En relación con el archivo proyectos.php se encuentra “editar.php” perteneciente a la misma carpeta, se encuentran las configuraciones iniciales de la barra de navegación, y como se muestra en la figura 344, se hace un llamado a la conexión de la base de datos dentro de la interpretación de php, se especifica la tabla a la cual se tendrá acceso de información.



```

257     </div>
258 </div>
259 <?php
260     include '../php/Conexion_be.php';
261     include '../php/firebaseRDB.php';
262
263     $db = new firebaseRDB($databaseUrl);
264     $id = $_GET['id'];
265     $retrieve = $db->retrieve("proyecto/$id");
266     $data = json_decode($retrieve, 1);
267 ?>
268
    
```

Figura. 344 Creación y configuración del archivo editar.php. Fuente. Elaboración propia.

Posteriormente como se observa en la figura 345, los inputs reciben una alteración de código, permitiendo que los datos almacenados con anterioridad en los campos designados sean llamados y visualizados dentro del mismo input y listas de selección, se conservan las reglas de patrones indicados por el archivo antecesor proveedores.php.

```

297     </div>
298     <div style="display: none;">
299     <input type="text" name="uid" value="{<?php echo $id}">
300     </div>
301     <div class="mdl-cell mdl-cell--12-col">
302     <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
303     <input class="mdl-textfield__input" type="text" pattern="-?[A-Za-z0-9
    ]*(\.[0-9]+)?" name="Nombre_del_proyecto" value="{<?php echo $data[
    Nombre_del_proyecto]}">
304     <label class="mdl-textfield__label" for="Nombre_del_proyecto">Nombre del
    Proyecto</label>
305     <span class="mdl-textfield__error">Nombre Invalida</span>
306     </div>
307     </div>
308     </div>
309     <div class="text-center">
310     <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
    mdl-button--colored bg-primary" id="btn-addPayment">
311     <i class="zmdi zmdi-plus"></i>
312     </button>
313     <div class="mdl-tooltip" for="btn-addPayment">Agregar Proyecto</div>
314     </div>
315     </form>

```

Figura. 345 Codificación y llamado de registros en los campos del formulario proyecto, Fuente. Elaboración propia

Se coloca un botón permitiendo la redirección al archivo “action_editar” continuando con el proceso de actualización de datos del formulario, así como se muestra en la figura 346.

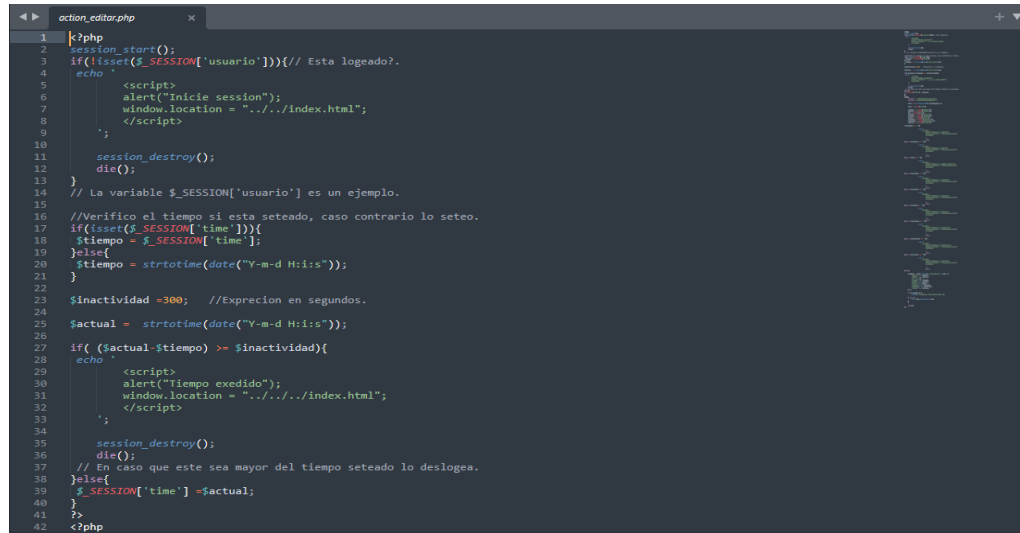
```

306     </div>
307     </div>
308     </div>
309     <div class="text-center">
310     <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
    mdl-button--colored bg-primary" id="btn-addPayment">
311     <i class="zmdi zmdi-plus"></i>
312     </button>
313     <div class="mdl-tooltip" for="btn-addPayment">Agregar Proyecto</div>
314     </div>
315     </form>
316     </div>
317     </div>
318     </div>
319     </div>
320     </div>
321     <div class="mdl-take--panel" id="tab-listPayment">

```

Figura. 346 Creación del botón agregar proyecto, Fuente. Elaboración propia.

Como se observa en la figura 347, al igual que el archivo `action_agregar` se conservan las instrucciones sobre inactividad.



```

1  <?php
2  session_start();
3  if(!isset($_SESSION['usuario'])){// Esta logeado?.
4      echo '
5          <script>
6              alert("Inicie session");
7              window.location = "../index.html";
8          </script>
9      ';
10     session_destroy();
11     die();
12 }
13 // La variable $_SESSION['usuario'] es un ejemplo.
14
15 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
16 if(isset($_SESSION['time'])){
17     $tiempo = $_SESSION['time'];
18 }else{
19     $tiempo = strtotime(date("Y-m-d H:i:s"));
20 }
21
22 $inactividad = 300; //Expreccion en segundos.
23
24 $actual = strtotime(date("Y-m-d H:i:s"));
25
26 if( ($actual-$tiempo) >= $inactividad){
27     echo '
28         <script>
29             alert("Tiempo excedido");
30             window.location = "../index.html";
31         </script>
32     ';
33     session_destroy();
34     die();
35 // En caso que este sea mayor del tiempo seteado lo deslogea.
36 }else{
37     $_SESSION['time'] = $actual;
38 }
39 }
40 }
41 }
42 <?php
    
```

Figura. 347 Codificación de tiempo de inactividad del archivo `action_editar.php`. Fuente, Elaboración propia.

Se hace el llamado a la conexión de la base de datos y creación de variables que almacenaran los cambios indicados en los input y listas, a su vez se verifica que dichos inputs contengan información ya que al no contenerla se coloca una alerta y redireccionamiento al archivo “`proyectos.php`”, así como se observa en la figura 348.

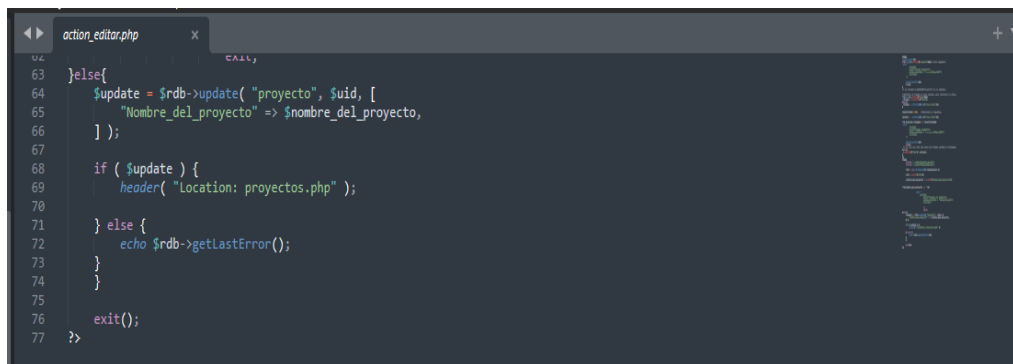


```

40 }
41 }
42 <?php
43 include '../php/Conexion_be.php';
44 include '../php/firebaseRDB.php';
45
46 $rdb = new firebaseRDB( $databaseUrl );
47
48 $uid = $_POST['uid'];
49
50 $nombre_del_proyecto = $_POST["Nombre_del_proyecto"];
51
52
53 if($nombre_del_proyecto == ""){
54     echo '
55         <script>
56             alert("Ingresa el nombre");
57             window.location = "proyectos.php";
58         </script>
59     ';
60     exit;
61 }
62 }
63 }
    
```

Figura. 348 Codificación de verificación para datos repetidos de los campos, Fuente, Elaboración propia.

En cambio, como se muestra en la figura 349, el proceso de ingreso de información, se anula la evaluación de datos repetidos y se comienza el proceso de actualización de datos, el cual una vez finalizado dará redireccionamiento al archivo proyectos.php.



```

63 }else{
64     $update = $rdb->update( "proyecto", $uid, [
65         "Nombre_del_proyecto" => $nombre_del_proyecto,
66     ] );
67
68     if ( $update ) {
69         header( "Location: proyectos.php" );
70     } else {
71         echo $rdb->getLastError();
72     }
73 }
74
75 exit();
76 }>
    
```

Figura. 349 Codificación de ingreso de datos a la tabla proyecto, Fuente. Elaboración propia.

Se muestra en la figura 350 el llenado de la tabla de los registros se ocupa el archivo “proveedores.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros que fueron actualizados y mostrarlos en el orden indicado con anterioridad.

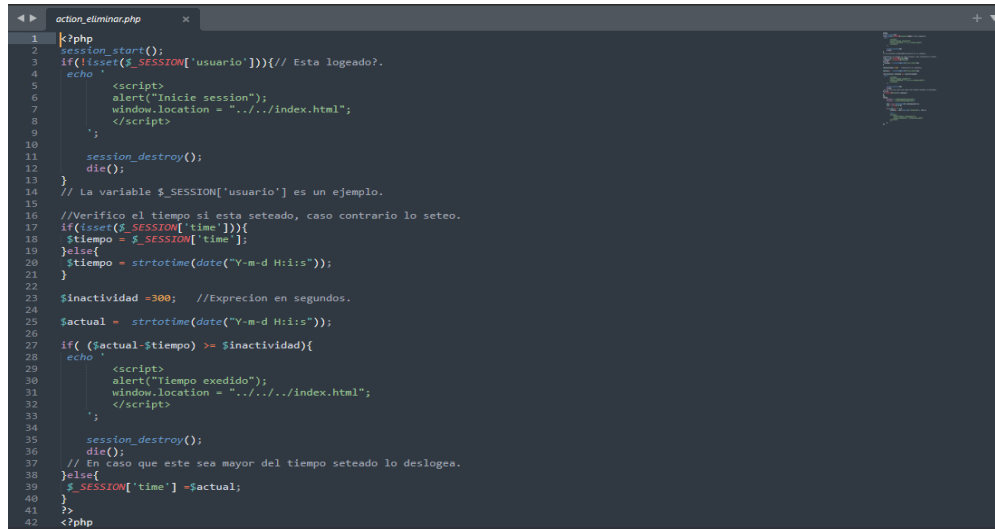


```

327 </div>
328 <div class="full-width panel-content">
329     <div class="table-responsive text-center">
330         <table class="table table-bordered table-hover" id="datatablesSimple">
331             <thead>
332                 <tr>
333                     <th scope="col"><strong>ID</strong></th>
334                     <th scope="col"><strong>Nombre del Proyecto</strong></th>
335                     <th scope="col"><strong>Acción</strong></th>
336                 </tr>
337             </thead>
338             <tbody id="tr_proyecto">
339             </tbody>
340         </table>
    
```

Figura. 350 Creación de la tabla de registros actualizados para proyectos, Fuente. Elaboración propia.

Por otra parte, como se muestra en la figura 351, se encuentra el archivo “action_eliminar” ubicado dentro de la misma carpeta de administración, Al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad.



```

1 <?php
2 session_start();
3 if(isset($_SESSION['usuario'])){// Esta logeado?.
4     echo '
5         <script>
6             alert("Inicio session");
7             window.location = ".../index.html";
8         </script>
9     ';
10
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Expreccion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28     echo '
29         <script>
30             alert("Tiempo excedido");
31             window.location = ".../index.html";
32         </script>
33     ';
34
35     session_destroy();
36     die();
37 // En caso que este sea mayor del tiempo seteado lo deslogea.
38 }else{
39     $_SESSION['time'] = $actual;
40 }
41 }
42 <?php
    
```

Figura. 351 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente, Elaboración propia.

Se hace el llamado a la conexión de la base de datos y se comienza el borrado de registro mediante una sentencia de evaluación que hace uso del identificador del registro y la variable delete, como se muestra en la figura 352, una vez finalizada la eliminación se coloca una alerta sobre la eliminación y se redirecciona al archivo “proyectos.php”.



```

41 <?php
42
43 include '../php/Conexion_be.php';
44 include '../php/firebaseRDB.php';
45
46 $db = new firebaseRDB( $databaseUrl );
47 $id = $_GET['id'];
48
49 if ( $id != "" ) {
50     $delete = $db->delete( "proyecto", $id );
51
52     echo '
53         <script>
54             alert("Datos eliminados");
55             window.location = "proyectos.php";
56         </script>
57     ';
58 }
59 >
    
```

Figura. 352 Codificación y eliminación de registros de la tabla proyecto, Fuente. Elaboración propia.

Se muestra en la figura 353 que, para que las funcionalidades de edición y eliminación empiecen su proceso, dentro de la tabla de visualización de registros se visualizan 2 botones de acción para cada proceso correspondiente, dichos botones son ubicados en el archivo “proyectos.js”.

```

35     const proyecto = proyectos[key];
36
37     const proyectoTable = document.createElement("tr");
38     proyectoTable.innerHTML = `
39     <tr>
40         <td><strong>${rowNumber}</strong></td>
41         <td>${proyecto.Nombre_del_proyecto}</td>
42         <td>
43             <div>
44                 <a href="editar.php?id=${key}" class="editar"><i class="zmdi zmdi-edit"></i></a>
45                 <a href="action_eliminar.php?id=${key}" class="eliminar"><i class="zmdi zmdi-delete"></i></a>
46             </div>
47         </td>
48     </tr>
49     `;
50     rowNumber++;
51     proyectoTr.appendChild(proyectoTable);
52 }

```

Figura. 353 Codificación de vínculos para los botones eliminar y editar, Fuente. Elaboración propia.

3.10.12. Salidas:

Por parte del archivo “salidas.php” se especifica en lenguaje de php mediante sentencias de decisión donde se evalúa si una vez en la página de home.php se autentico de manera correcta, como se observa en la figura 354, en caso de no ser así se coloca una alerta y retornara a index.html.

```

1  <?php
2  session_start();
3  if(!isset($_SESSION['usuario'])){// Esta logeado?.
4  echo '
5      <script>
6          alert("Inicie session");
7          window.location = "../index.html";
8      </script>
9  '
10
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad = 300; //Exprecion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28     echo '
29         <script>
30             alert("Tiempo excedido");
31             window.location = "../index.html";
32         </script>
33     '
34 }

```

Figura. 354 Configuración de lenguaje y tiempo de inactividad del archivo salidas.php. Fuente. Elaboración propia.

Posteriormente como se muestra en la figura 355, se coloca la estructura de !DOCTYPE html> la cual informara al navegador el tipo de versión HTML que fue implementado en el archivo, a su vez se indica con una etiqueta de tipo lang el tipo de idioma con el que será escrita la página web en este caso se hace uso del idioma español. A su vez se muestra la apertura de un head en el cual se proveerá la información como lo es el título y lo enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes son llamadas de la carpeta “assets” y de las sub carpetas de “img”, “css” y “js”.

```

43 <!DOCTYPE html>
44 <html lang="es">
45
46 <head>
47   <meta charset="UTF-8">
48   <meta name="viewport" content="width=device-width, initial-scale=1">
49   <title>Personal</title>
50   <link rel="icon" href="../../assets/img/logpa.png" type="image/png">
51   <link rel="stylesheet" href="../../assets/css/normalize.css">
52   <link rel="stylesheet" href="../../assets/css/sweetalert2.css">
53   <link rel="stylesheet" href="../../assets/css/material.min.css">
54   <link rel="stylesheet" href="../../assets/css/material-design-iconic-font.min.css">
55   <link rel="stylesheet" href="../../assets/css/jquery.mCustomScrollbar.css">
56   <link rel="stylesheet" href="../../assets/css/main.css">
57   <link rel="stylesheet" href="../../assets/css/simple-datatables.min.css">
58   <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
59   <script>
60     window.jQuery || document.write('<script src="../../assets/js/jquery-1.11.2.min.js"></script>')
61   </script>
62   <script src="../../assets/js/material.min.js"></script>
63   <script src="../../assets/js/sweetalert2.min.js"></script>
64   <script src="../../assets/js/jquery.mCustomScrollbar.concat.min.js"></script>
65   <script src="../../assets/js/marco2.js"></script>
66   <script src="../../assets/js/simple-datatables.js"></script>
67   <!-- links para exportar a excel -->
68   <script src="https://unpkg.com/xlsx@0.16.9/dist/xlsx.full.min.js"></script>
69   <script src="https://unpkg.com/file-saverjs@latest/FileSaver.min.js"></script>
70   <script src="https://unpkg.com/tableexport@latest/dist/js/tableexport.min.js"></script>
71 </head>

```

Figura. 355 Estructura de lenguaje HTML e Idioma del archivo salidas.php, Fuente. Elaboración propia.

Una vez definido el encabezado de la página mediante la etiqueta head como se observa en la figura 356, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web. Posteriormente se coloca un elemento que ayudará a representar el contenido principal de la página web que es el “body”. Posteriormente se abre una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, se colocan contenedores para aplicar estilos correspondientes a las sentencias, se agregan imágenes que harán la función del avatar del administrador, así como el nombre del usuario administrador que se autentico con anterioridad los datos son llamados desde la base de datos de Firebase.

```

72
73 <body>
74 <!-- barra de navegacion lateral -->
75 <section class="full-width navLateral">
76 <div class="full-width navLateral-bg btn-menu"></div>
77 <div class="full-width navLateral-body">
78 <div class="full-width navLateral-body-logo text-center tittles">
79 <i class="zmdi zmdi-close btn-menu"></i> CC Camaleon Color
80 </div>
81 <figure class="full-width navLateral-body-tittle-menu">
82 <div>
83 
84 </div>
85 <figcaption>
86 <span><span><?php echo $_SESSION['usuario']["NombreCompleto"];></span>&nbsp;<span><?php echo $_SESSION['
87 usuario']["Apellido"];></span></span>
88 <br>
89 <span><small><?php echo $_SESSION['usuario']["Usuario_a"];></small></span>
90 </figcaption>
91 </figure>

```

Figura. 356 Codificación del nombre y avatar del usuario. Fuente. Elaboración propia.

Posteriormente se coloca una etiqueta nav para proporcionar una navegación estructurada en la página, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta assets/css, así como se muestra en las figuras 357, 358, 359 y 360.

```

91 <nav class="full-width">
92 <ul class="full-width list-unstyled menu-principal">
93 <li class="full-width">
94 <a href="../../../home.php" class="full-width">
95 <div class="navLateral-body-cl">
96 <i class="zmdi zmdi-view-dashboard"></i>
97 </div>
98 <div class="navLateral-body-cr">
99 INICIO
100 </div>
101 </a>
102 </li>
103 <li class="full-width divider-menu-h"></li>
104 <li class="full-width">
105 <a href="#" class="full-width btn-subMenu">
106 <div class="navLateral-body-cl">
107 <i class="zmdi zmdi-case"></i>
108 </div>
109 <div class="navLateral-body-cr">
110 ADMINISTRACION
111 </div>
112 <span class="zmdi zmdi-chevron-left"></span>
113 </a>
114 <ul class="full-width menu-principal sub-menu-options">
115 <li class="full-width">
116 <a href="../../../proveedores/proveedores.php" class="full-width">
117 <div class="navLateral-body-cl">
118 <i class="zmdi zmdi-truck"></i>
119 </div>
120 <div class="navLateral-body-cr">
121 PROVEEDORES
122 </div>
123 </a>
124 </li>
125 <li class="full-width">
126 <a href="../../../ubicacion/ubicacion.php" class="full-width">
127 <div class="navLateral-body-cl">
128 <i class="zmdi zmdi-card"></i>
129 </div>
130 <div class="navLateral-body-cr">
131 UBICACIONES
132 </div>

```

Figura. 357 Configuración de la barra de navegación lateral pt1, Fuente. Elaboración propia.

```

129         </div>
130         <div class="navLateral-body-cr">
131             UBICACIONES
132         </div>
133     </a>
134 </li>
135 <li class="full-width">
136     <a href=".."categorias/categorias.php" class="full-width">
137         <div class="navLateral-body-cl">
138             <i class="zmdi zmdi-label"></i>
139         </div>
140         <div class="navLateral-body-cr">
141             CATEGORIAS
142         </div>
143     </a>
144 </li>
145 <li class="full-width">
146     <a href=".."proyecto/proyectos.php" class="full-width">
147         <div class="navLateral-body-cl">
148             <i class="zmdi zmdi-case"></i>
149         </div>
150         <div class="navLateral-body-cr">
151             PROYECTOS
152         </div>
153     </a>
154 </li>
155 </ul>
156 </li>
157 <li class="full-width divider-menu-h"></li>
158 <li class="full-width">
159     <a href="#" class="full-width btn-subMenu">
160         <div class="navLateral-body-cl">
161             <i class="zmdi zmdi-face"></i>
162         </div>
163         <div class="navLateral-body-cr">
164             USUARIOS
165         </div>
166     </a>
167     <span class="zmdi zmdi-chevron-left"></span>
168     <ul class="full-width menu-principal sub-menu-options">
169         <li class="full-width">
170             <a href="admin.php" class="full-width">

```

Figura. 358 Configuración de la barra de navegación lateral pt2, Fuente. Elaboración propia.

```

176         </li>
177     </ul>
178 </li>
179 <li class="full-width">
180     <a href=".."personal/personal.php" class="full-width">
181         <div class="navLateral-body-cl">
182             <i class="zmdi zmdi-accounts"></i>
183         </div>
184         <div class="navLateral-body-cr">
185             PERSONAL CALIFICADO
186         </div>
187     </a>
188 </li>
189 </ul>
190 </li>
191 <li class="full-width divider-menu-h"></li>
192 <li class="full-width">
193     <a href=".."compras/compras.php" class="full-width">
194         <div class="navLateral-body-cl">
195             <i class="zmdi zmdi-shopping-cart"></i>
196         </div>
197         <div class="navLateral-body-cr">
198             COMPRAS
199         </div>
200     </a>
201 </li>
202 <li class="full-width divider-menu-h"></li>
203 <li class="full-width">
204     <a href=".."productos/productos.php" class="full-width">

```

Figura. 359 Configuración de la barra de navegación lateral pt3, Fuente. Elaboración propia.

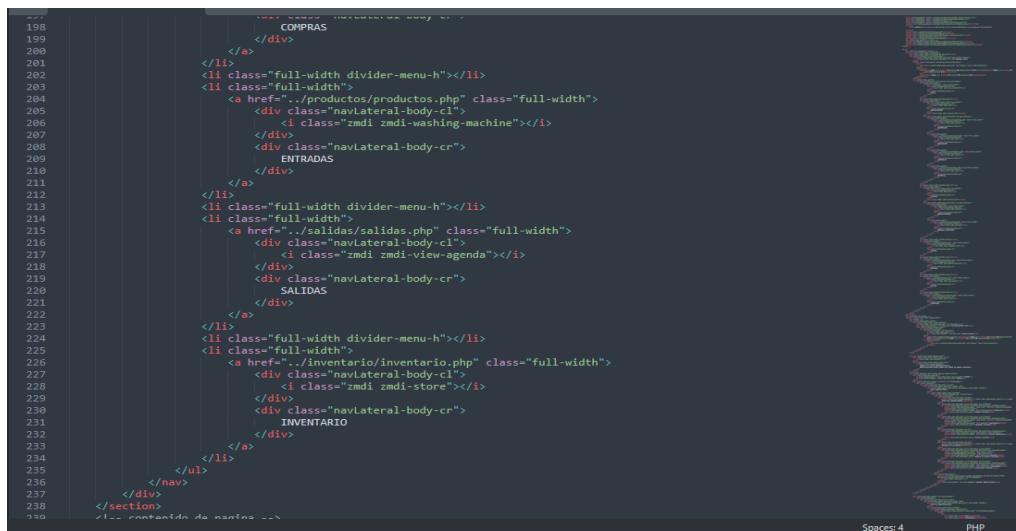


Figura. 360 Configuración de la barra de navegación lateral pt4. Fuente. Elaboración propia.

Por parte del contenido de la página web se coloca una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, la cual contendrá contenedores para aplicar estilos con css, además de continuar con las animaciones de la barra de navegación, se presenta la configuración para ocultar y mostrar la barra de navegación. Se programa la interfaz de cierre de sesión mediante la implementación de un botón con identificador, como se muestra en la figura 361. Posteriormente se coloca una imagen con el avatar de identificación del administrador y se manda a llamar los datos propios de este, desde la base de datos Firebase.

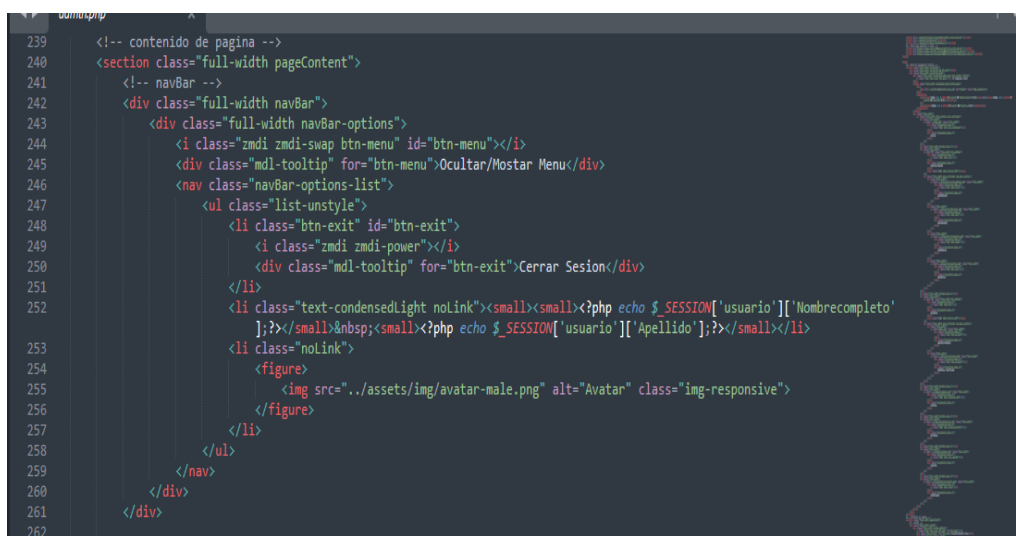


Figura. 361 Creación de los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.

Las animaciones correspondientes al cierre de sesión y ocultar/mostrar la barra de navegación son estructuradas en el archivo “marco.js” ubicado en la carpeta assets/js, así como se muestra en la figura 362.

```

1  (document).ready(function () {
2      //Mostrar ocultar menu principal*/
3      $('.btn-menu').on('click', function () {
4          var navLateral = $('.navLateral');
5          var pageContent = $('.pageContent');
6          var navOption = $('.navBar-options');
7          if (navLateral.hasClass('navLateral-change') && pageContent.hasClass('pageContent-change')) {
8              navLateral.removeClass('navLateral-change');
9              pageContent.removeClass('pageContent-change');
10             navOption.removeClass('navBar-options-change');
11         } else {
12             navLateral.addClass('navLateral-change');
13             pageContent.addClass('pageContent-change');
14             navOption.addClass('navBar-options-change');
15         }
16     });
17     //Salir del sistema*/
18     $('.btn-exit').on('click', function () {
19         swal({
20             title: 'Quieres salir del sistema?',
21             text: 'La sesion actual se cerrara',
22             type: 'warning',
23             showCancelButton: true,
24             confirmButtonText: 'Salir',
25             closeOnConfirm: false
26         },
27         function (isConfirm) {
28             if (isConfirm) {
29                 window.location = 'php/cerrar_sesion.php';
30             }
31         });
32     });
33     //Mostrar y ocultar submenus*/
34     $('.btn-subMenu').on('click', function () {
35         var subMenu = $(this).next('ul');
36         var icon = $(this).children('span');
37         if (subMenu.hasClass('sub-menu-options-show')) {
38             subMenu.removeClass('sub-menu-options-show');
39             icon.addClass('zmdi-chevron-left').removeClass('zmdi-chevron-down');
40         } else {
41             subMenu.addClass('sub-menu-options-show');
42             icon.addClass('zmdi-chevron-down').removeClass('zmdi-chevron-left');
43         }
44     });
45 });

```

Figura. 362 Codificación de animaciones para los botones ocultar/mostrar y cerrar sesión, Fuente. Elaboración propia.

Tal y como se observa en la figura 363, la creación de nuevos contenedores que almacenaran los inputs necesarios para el registro de información. Al finalizar las inserciones de los input se crea un botón dentro de un contenedor que comenzara el procesamiento de información del formulario.

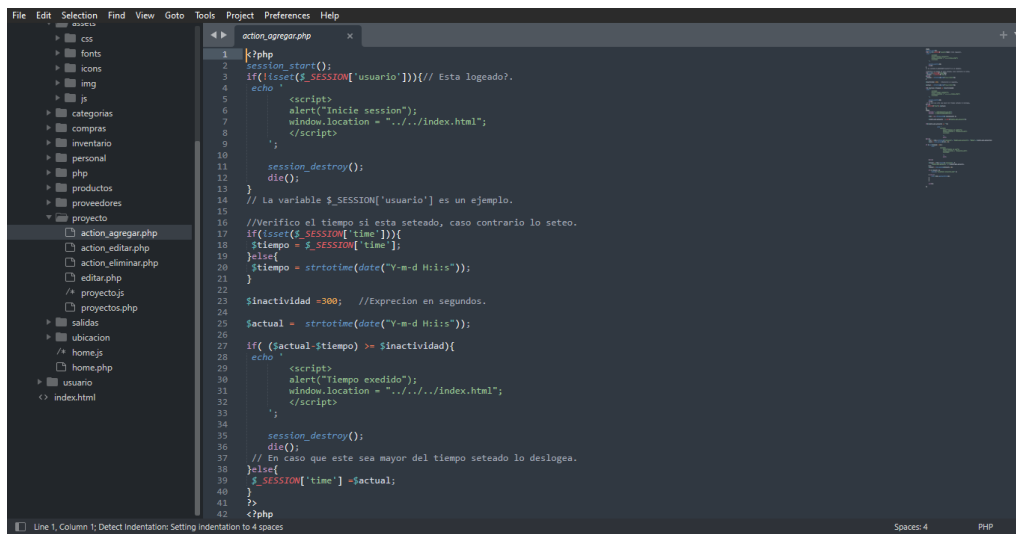
```

274 </section>
275 <div class="full-width divider-menu-h"></div>
276 <div class="mdl-grid">
277   <div class="mdl-cell mdl-cell--4-col-phone mdl-cell--8-col-tablet mdl-cell--12-col-desktop">
278
279     <div class="table-responsive text-center">
280       <table class="table table-bordered table-hover" id="datatablesSimple">
281         <thead>
282           <tr>
283             <th scope="col"><strong>ID</strong></th>
284             <th scope="col"><strong>Usuario</strong></th>
285             <th scope="col"><strong>Producto</strong></th>
286             <th scope="col"><strong>Proveedor</strong></th>
287             <th scope="col"><strong>Cantidad Retirada</strong></th>
288             <th scope="col"><strong>Fecha de Salida</strong></th>
289           </tr>
290         </thead>
291         <tbody id="tr_salidas">
292           </tbody>
293       </table>
294       <button id="btnExportar" class="btn btn-success">
295         <i class="fas fa-file-excel"></i> Exportar datos a Excel
296       </button>
297       <button id="dl-pdf">Exportar datos a PDF

```

Figura. 363 Creación de la tabla de registros para salidas. Fuente. Elaboración propia.

No obstante como se observa en la figura 364, para que el procesamiento de datos este completo el uso del archivo “action_agregar_editar.php” cumple con la función de verificar el tiempo de inactividad dentro de la página y almacenar los datos obtenidos de los inputs y listas de selección en variables, las cuales serán evaluadas mediante sentencias condicionales identificando si la variable recibió información o esta es nula, a su vez actualiza la información recibida por medio de los input, si la información obtenida resulta ser nula se regresara al archivo “salidas.php”.

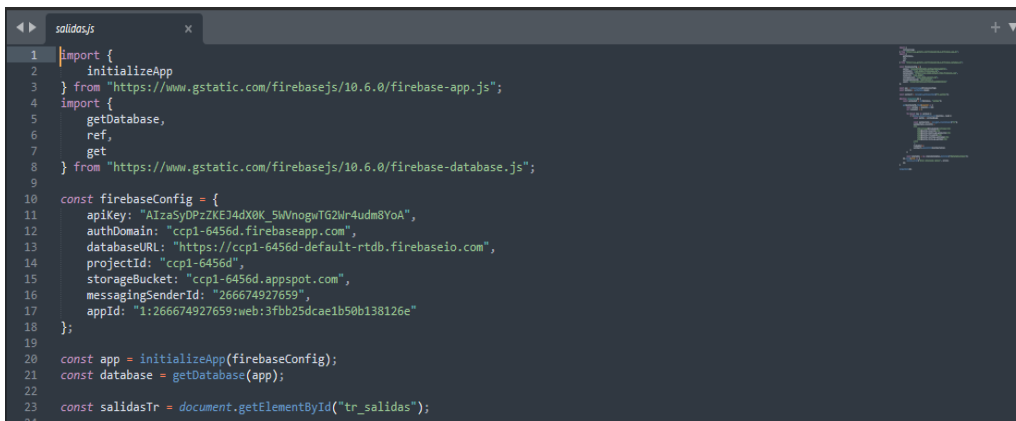


```

1 <?php
2 session_start();
3 if(!isset($_SESSION['usuario'])){// Esta logeado?.
4     echo "
5     <script>
6         alert("Inicie session");
7         window.location = "../index.html";
8     </script>
9 ";
10
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Expresion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if(($actual-$tiempo) >= $inactividad){
28     echo "
29     <script>
30         alert("Tiempo excedido");
31         window.location = "../index.html";
32     </script>
33 ";
34
35     session_destroy();
36     die();
37 // En caso que este sea mayor del tiempo seteado lo deslogea.
38 }else{
39     $_SESSION['time'] = $actual;
40 }
41 }
42 <?php
  
```

Figura. 364 Codificación de tiempo de inactividad del archivo action_editar.php, Fuente, Elaboración propia.

A sí que dentro del archivo salidas.js y salida.js, se hace la conexión a la base de datos y se mandan llamar a las tablas necesarias para cargar y almacenar la información en las listas mediante el uso de variables, como se muestra en la figura 365.



```

1 import {
2     initializeApp
3 } from "https://www.gstatic.com/firebasejs/10.6.0/firebase-app.js";
4 import {
5     getDatabase,
6     ref,
7     get
8 } from "https://www.gstatic.com/firebasejs/10.6.0/firebase-database.js";
9
10 const firebaseConfig = {
11     apiKey: "AIzaSyDPzZKE14dX9K_5WVn0gwT62Mr4udm8YoA",
12     authDomain: "ccpl-6456d.firebaseio.com",
13     databaseURL: "https://ccpl-6456d-default-rtdb.firebaseio.com",
14     projectId: "ccpl-6456d",
15     storageBucket: "ccpl-6456d.appspot.com",
16     messagingSenderId: "266674927659",
17     appId: "1:266674927659:web:3fbb25dcae1b50b138126e"
18 };
19
20 const app = initializeApp(firebaseConfig);
21 const database = getDatabase(app);
22
23 const salidasTr = document.getElementById("tr_salidas");
  
```

Figura. 365 Conexión a la base de datos e incorporación de registros en listas del formulario. Fuente. Elaboración propia.

Por otro lado, como se muestra en la figura 366, se inserta por medio de una variable que almacenará los datos correctos y devolverá al archivo salidas.php.

```

69  $cantidadrestante = $stock - $cantidad;
70
71  $update = $rdb->update( "productos", $uid, [
72    "CantidadDinamica" => $cantidadrestante,
73  ] );
74
75  if ( $insert || $update ) {
76    header( "Location: salidas.php" );
77  } else {
78    echo $rdb->getLastError();
79  }
80
81  exit();
82
83  ?>

```

Figura. 366 Codificación de ingreso de datos a la tabla productos, Fuente. Elaboración propia.

Una vez realizado el ingreso de información se coloca un contenedor el cual almacenara el título de la lista de registros y se da formato a la tabla de registros con los campos acorde a los inputs y registros de la base de datos, se asignan identificadores para dar formato mediante css y hacer uso de Java Script, así como se observa en la figura 367.

```

273  </div>
274  </section>
275  <div class="full-width divider-menu-h"></div>
276  <div class="mdl-grid">
277    <div class="mdl-cell mdl-cell--4-col-phone mdl-cell--8-col-tablet mdl-cell--12-col-desktop">
278      <div class="table-responsive text-center">
279        <table class="table table-bordered table-hover" id="datatablesSimple">
280          <thead>
281            <tr>
282              <th scope="col"><strong>ID</strong></th>
283              <th scope="col"><strong>Usuario</strong></th>
284              <th scope="col"><strong>Producto</strong></th>
285              <th scope="col"><strong>Proveedor</strong></th>
286              <th scope="col"><strong>Cantidad Retirada</strong></th>
287              <th scope="col"><strong>Fecha de Salida</strong></th>
288            </tr>
289          </thead>
290          <tbody id="tr_salidas">
291            <tbody>
292          </tbody>
293        </table>
294        <button id="btnExportar" class="btn btn-success">

```

Figura. 367 Creación de la tabla de registros actualizados para las salidas, Fuente. Elaboración propia.

Para llenar la tabla de los registros se ocupan de 2 archivos, “salidas.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros de cada insert para almacenarlos y mostrarlos en el orden indicado con anterioridad, como se observa en la figura 368.

```

1  import {
2    initializeApp
3  } from "https://www.
4  import {
5    getDatabase,
6    ref,
7    get
8  } from "https://www.
9
10 const firebaseConfig = {
11   apiKey: "AIzaSyDPzZ
12   authDomain: "ccp1-6456d.
13   databaseURL: "https://c
14   projectId: "ccp1-6456d",
15   storageBucket: "ccp1-6456d.
16   messagingSenderId: "266674927659
17   appId: "1:266674927
18 };
19
20 const app = initializeApp(firebaseConfig);
21 const database = getDatabase(app);
22
23 const proveedorTr = document.getElementById("tr_proveedores");
24 const paisSelect = document.getElementById("pais_select");
25
26 function CargarDatos() {
27   const proveedorRef = ref(database, "proveedores");
28
29   get(proveedorRef).then(snapshot => {
30     const proveedores = snapshot.val();
31     let rowNumber = 1;
32
33     for (const key in proveedores) {
34       if (Object.hasOwn(proveedores, key)) {
35         const proveedor = proveedores[key];
36
37         const proveedorTable = document.createElement("tr");
38         proveedorTable.innerHTML = `
39           <tr>
40             <td><strong>${rowNumber}</strong></td>
41             <td>${proveedor.Nombre}</td>

```

Figura. 368 Codificación de ingreso de datos a la tabla de salidas, Fuente. Elaboración propia.

Lo que se muestra en la figura 39 es que, dentro de la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, mediante el uso de un script colocado posteriormente del llamado “salidas.js”.

```

425   <button id="btnExportar" class="btn btn-success">
426     <i class="fas fa-file-excel"></i> Exportar datos a Excel
427   </button>
428 </div>
429 </div>
430 </div>
431 </div>
432 </div>
433 </div>
434 </div>
435 </section>
436
437 <script type="module" src="proveedores.js"></script>
438
439 <!-- script para exportar a excel -->
440 <script>
441   const $btnExportar = document.querySelector("#btnExportar"),
442     $datatablesSimple = document.querySelector("#datatablesSimple");
443
444   $btnExportar.addEventListener("click", function() {
445     let tableExport = new TableExport($datatablesSimple, {
446       exportButtons: false, // No queremos botones
447       filename: "Reporte de Proveedores", //Nombre del archivo de Excel
448       sheetname: "Reporte de Proveedores", //Título de la hoja
449     });
450     let datos = tableExport.getExportData();
451     let preferenciasDocumento = datos.datatablesSimple.xlsx;
452     tableExport.export2file(preferenciasDocumento.data, preferenciasDocumento.mimeType, preferenciasDocumento.filename,
453       preferenciasDocumento.fileExtension, preferenciasDocumento.merges, preferenciasDocumento.RTL,
454       preferenciasDocumento.sheetname);
455   });
456 </script>
457 </body>
458 </html>

```

Figura. 369 Codificación y conversión de registros a formato Excel, Fuente. Elaboración propia.

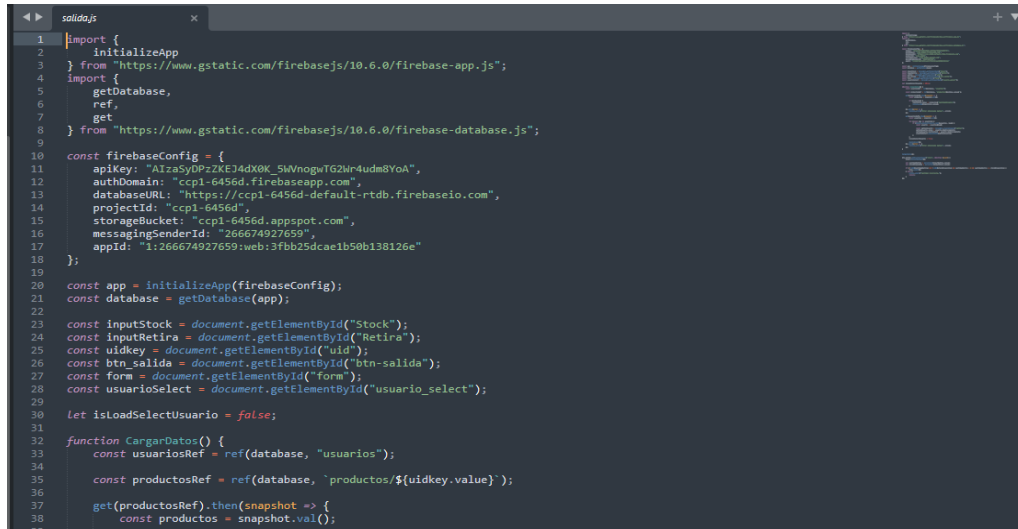
solid454n

370 Codificación y llamado de registros en los campos del formulario de salidas. Fuente: Elab.

See Project Hierarchies Page

2371 Codificación de enlaces de navegación y conexión a la base de datos. Fuente: Elaboración

Por lo tanto, como se observa en la figura 372, el archivo hace uso de salida.js el cual almacena las animaciones y cálculos de conexión, así como el llamado de información para su actualización.



```

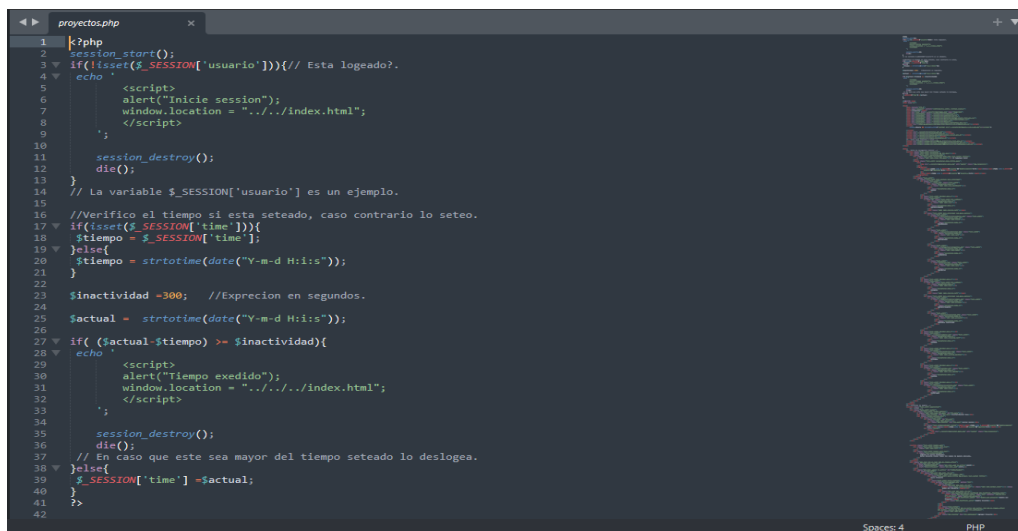
1 import {
2   initializeApp
3 } from "https://www.gstatic.com/firebasejs/10.6.0/firebase-app.js";
4 import {
5   getDatabase,
6   ref,
7   get
8 } from "https://www.gstatic.com/firebasejs/10.6.0/firebase-database.js";
9
10 const firebaseConfig = {
11   apiKey: "AIzaSyDPz2KEj4dX0K_5Wn0gwT62W4udm8Y0A",
12   authDomain: "ccp1-6456d.firebaseio.com",
13   databaseURL: "https://ccp1-6456d-default-rtdb.firebaseio.com",
14   projectId: "ccp1-6456d",
15   storageBucket: "ccp1-6456d.appspot.com",
16   messagingSenderId: "266674927659",
17   appId: "1:266674927659:web:3fbb25dcae1b50b138126e"
18 };
19
20 const app = initializeApp(firebaseConfig);
21 const database = getDatabase(app);
22
23 const inputStock = document.getElementById("Stock");
24 const inputRetira = document.getElementById("Retira");
25 const uidkey = document.getElementById("uid");
26 const btn_salida = document.getElementById("btn-salida");
27 const form = document.getElementById("form");
28 const usuarioSelect = document.getElementById("usuario_select");
29
30 let isLoadingSelectUsuario = false;
31
32 function CargarDatos() {
33   const usuariosRef = ref(database, "usuarios");
34   const productosRef = ref(database, `productos/${uidkey.value}`);
35
36   get(productosRef).then(snapshot => {
37     const productos = snapshot.val();
38   });
39 }

```

Figura. 372 Conexión a la base de datos y codificación de consulta de datos, Fuente. Elaboración propia.

3.10.13. Ubicaciones:

Por parte del archivo “Ubicaciones.php” se especifica en lenguaje de php mediante sentencias de decisión mostrado en figura 373.



```

1 <?php
2 session_start();
3 if(isset($_SESSION['usuario'])){// Esta logeado?.
4   echo
5     <script>
6       alert("Inicie session");
7       window.location = "../index.html";
8     </script>
9   ;
10   session_destroy();
11   die();
12 }
13 // La variable $_SESSION['usuario'] es un ejemplo.
14
15 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
16 if(isset($_SESSION['time'])){
17   $tiempo = $_SESSION['time'];
18 }else{
19   $tiempo = strtotime(date("Y-m-d H:i:s"));
20 }
21
22 $inactividad = 300; //Exprecion en segundos.
23 $actual = strtotime(date("Y-m-d H:i:s"));
24
25 if( ($actual-$tiempo) >= $inactividad){
26   echo
27     <script>
28       alert("Tiempo exedido");
29       window.location = "../index.html";
30     </script>
31   ;
32   session_destroy();
33   die();
34 }
35 // En caso que este sea mayor del tiempo seteado lo deslogea.
36 if( ($actual-$tiempo) >= $inactividad){
37   $_SESSION['time'] = $actual;
38 }
39 }
40 }
41 }
42

```

Figura. 373 Configuración de lenguaje y tiempo de inactividad del archivo ubicaciones.php, Fuente. Elaboración propia.

Posteriormente como se muestra en la figura 374, da inicio a la apertura de un head en el cual se proveerá la información como lo es el título y lo enlaces de los scripts, css e imágenes, los css y las imágenes correspondientes a ubicaciones, son llamadas de la carpeta “assets” y de las sub carpetas de “img”, “css” y “js”.

```

43 <!DOCTYPE html>
44 <html lang="es">
45
46 <head>
47   <meta charset="UTF-8">
48   <meta name="viewport" content="width=device-width, initial-scale=1">
49   <title>Personal</title>
50   <link rel="icon" href="../../assets/img/logo.png" type="image/png">
51   <link rel="stylesheet" href="../../assets/css/normalize.css">
52   <link rel="stylesheet" href="../../assets/css/sweetalert2.css">
53   <link rel="stylesheet" href="../../assets/css/material.min.css">
54   <link rel="stylesheet" href="../../assets/css/material-design-iconic-font.min.css">
55   <link rel="stylesheet" href="../../assets/css/jquery.mCustomScrollbar.css">
56   <link rel="stylesheet" href="../../assets/css/main.css">
57   <link rel="stylesheet" href="../../assets/css/simple-datatables.min.css">
58   <script src="//ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
59   <script>
60     window.jQuery || document.write('<script src="../../assets/js/jquery-1.11.2.min.js"></script>')
61   </script>
62   <script src="../../assets/js/material.min.js"></script>
63   <script src="../../assets/js/sweetalert2.min.js"></script>
64   <script src="../../assets/js/jquery.mCustomScrollbar.concat.min.js"></script>
65   <script src="../../assets/js/marco2.js"></script>
66   <script src="../../assets/js/simple-datatables.js"></script>
67   <!-- links para exportar a excel -->
68   <script src="https://unpkg.com/xlsx@0.16.9/dist/xlsx.full.min.js"></script>
69   <script src="https://unpkg.com/file-saverjs@latest/FileSaver.min.js"></script>
70   <script src="https://unpkg.com/tableexport@latest/dist/js/tableexport.min.js"></script>
71 </head>

```

Figura. 374 Estructura de lenguaje HTML e Idioma del archivo ubicaciones.php, Fuente. Elaboración propia.

Una vez definido el encabezado de la página mediante la etiqueta head, se abre una nueva etiqueta “body” la cual definirá el área donde se colocará el contenido visible de la página web, como se observa en la figura 375. Posteriormente se coloca un elemento que ayudara a representar el contenido principal de la página web que es el “body”. Posteriormente se abre una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, se colocan contenedores para aplicar estilos correspondientes a las sentencias, se agregan imágenes que harán la función del avatar del administrador, así como el nombre del usuario administrador que se autentico con anterioridad los datos son llamados desde la base de datos de Firebase

```

72
73 <body>
74   <!-- barra de navegacion lateral -->
75   <section class="full-width navlateral">
76     <div class="full-width navlateral-bg btn-menu"></div>
77     <div class="full-width navlateral-body">
78       <div class="full-width navlateral-body-logo text-center titles">
79         <i class="zmdi zmdi-close btn-menu"></i> CC Camaleon Color
80       </div>
81       <figure class="full-width navlateral-body-title-menu">
82         <div>
83           
84         </div>
85         <figcaption>
86           <span><span><?php echo $_SESSION['usuario'];></span><span>&nbsp;<span><?php echo $_SESSION['
87             usuario'];></span></span>
88           <br>
89           <span><small><?php echo $_SESSION['usuario'];></small></span>
90         </figcaption>

```

Figura. 375 Codificación del nombre y avatar del usuario, Fuente. Elaboración propia.

Posteriormente como se muestra en las figuras, 376, 377, 378 y 379 se coloca una etiqueta nav para proporcionar una navegación estructurada en la página, se agregan vínculos de navegación entre archivos dependiendo de la necesidad del usuario, los cuales son colocados en contenedores con identificadores para dar formato con los archivos CSS ubicados en la carpeta assets/css.

```

91 <nav class="full-width">
92 <ul class="full-width list-unstyled menu-principal">
93 <li class="full-width">
94 <a href=".."home.php" class="full-width">
95 <div class="navLateral-body-cl">
96 <i class="zmdi zmdi-view-dashboard"></i>
97 </div>
98 <div class="navLateral-body-cr">
99 INICIO
100 </div>
101 </a>
102 </li>
103 <li class="full-width divider-menu-h"></li>
104 <li class="full-width">
105 <a href="#" class="full-width btn-subMenu">
106 <div class="navLateral-body-cl">
107 <i class="zmdi zmdi-case"></i>
108 </div>
109 <div class="navLateral-body-cr">
110 ADMINISTRACION
111 </div>
112 <span class="zmdi zmdi-chevron-left"></span>
113 </a>
114 <ul class="full-width menu-principal sub-menu-options">
115 <li class="full-width">
116 <a href=".."proveedores/proveedores.php" class="full-width">
117 <div class="navLateral-body-cl">
118 <i class="zmdi zmdi-truck"></i>
119 </div>
120 <div class="navLateral-body-cr">
121 PROVEEDORES
122 </div>
123 </li>
124 </ul>
125 </li>
126 <li class="full-width">
127 <a href=".."ubicacion/ubicacion.php" class="full-width">
128 <div class="navLateral-body-cl">
129 <i class="zmdi zmdi-card"></i>
130 </div>
131 <div class="navLateral-body-cr">
132 UBICACIONES
133 </div>
134 </a>
135 </li>
136 <li class="full-width">
137 <a href=".."categorias/categorias.php" class="full-width">
138 <div class="navLateral-body-cl">
139 <i class="zmdi zmdi-label"></i>
140 </div>
141 <div class="navLateral-body-cr">
142 CATEGORIAS
143 </div>
144 </a>
145 </li>
146 <li class="full-width">
147 <a href=".."proyecto/proyectos.php" class="full-width">
148 <div class="navLateral-body-cl">
149 <i class="zmdi zmdi-case"></i>
150 </div>
151 <div class="navLateral-body-cr">
152 PROYECTOS
153 </div>
154 </a>
155 </li>
156 </ul>
157 </li>
158 <li class="full-width divider-menu-h"></li>
159 <li class="full-width">
160 <a href="#" class="full-width btn-subMenu">
161 <div class="navLateral-body-cl">
162 <i class="zmdi zmdi-face"></i>
163 </div>
164 <div class="navLateral-body-cr">
165 USUARIOS
166 </div>
167 <span class="zmdi zmdi-chevron-left"></span>
168 </a>
169 <ul class="full-width menu-principal sub-menu-options">
170 <li class="full-width">
171 <a href="admin.php" class="full-width">

```

Figura. 376 Configuración de la barra de navegación lateral pt1, Fuente. Elaboración propia.

```

129 </div>
130 <div class="navLateral-body-cr">
131 UBICACIONES
132 </div>
133 </a>
134 </li>
135 <li class="full-width">
136 <a href=".."categorias/categorias.php" class="full-width">
137 <div class="navLateral-body-cl">
138 <i class="zmdi zmdi-label"></i>
139 </div>
140 <div class="navLateral-body-cr">
141 CATEGORIAS
142 </div>
143 </a>
144 </li>
145 <li class="full-width">
146 <a href=".."proyecto/proyectos.php" class="full-width">
147 <div class="navLateral-body-cl">
148 <i class="zmdi zmdi-case"></i>
149 </div>
150 <div class="navLateral-body-cr">
151 PROYECTOS
152 </div>
153 </a>
154 </li>
155 </ul>
156 </li>
157 <li class="full-width divider-menu-h"></li>
158 <li class="full-width">
159 <a href="#" class="full-width btn-subMenu">
160 <div class="navLateral-body-cl">
161 <i class="zmdi zmdi-face"></i>
162 </div>
163 <div class="navLateral-body-cr">
164 USUARIOS
165 </div>
166 <span class="zmdi zmdi-chevron-left"></span>
167 </a>
168 <ul class="full-width menu-principal sub-menu-options">
169 <li class="full-width">
170 <a href="admin.php" class="full-width">

```

Figura. 377 Configuración de la barra de navegación lateral pt2, Fuente. Elaboración propia.

```

179      </li>
180      <li class="full-width">
181        <a href="../personal/personal.php" class="full-width">
182          <div class="navlateral-body-cl">
183            <i class="zmdi zmdi-accounts"></i>
184            </div>
185            <div class="navlateral-body-cr">
186              PERSONAL CALIFICADO
187            </div>
188          </li>
189        </ul>
190      </li>
191      <li class="full-width divider-menu-h"></li>
192      <li class="full-width">
193        <a href="../compras/compras.php" class="full-width">
194          <div class="navlateral-body-cl">
195            <i class="zmdi zmdi-shopping-cart"></i>
196            </div>
197            <div class="navlateral-body-cr">
198              COMPRAS
199            </div>
200          </li>
201        </ul>
202      <li class="full-width divider-menu-h"></li>
203      <li class="full-width">
204        <a href="../productos/productos.php" class="full-width">
205          ..
206          ..
207          ..
208          ..
209          ..
210          ..
211          ..
212          ..
213          ..
214          ..
215          ..
216          ..
217          ..
218          ..
219          ..
220          ..
221          ..
222          ..
223          ..
224          ..
225          ..
226          ..
227          ..
228          ..
229          ..
230          ..
231          ..
232          ..
233          ..
234          ..
235          ..
236          ..
237          ..
238          ..
239          ..
240          ..
241          ..
242          ..
243          ..
244          ..
245          ..
246          ..
247          ..
248          ..
249          ..
250          ..
251          ..
252          ..
253          ..
254          ..
255          ..
256          ..
257          ..
258          ..
259          ..
260          ..
261          ..
262          ..
263          ..
264          ..
265          ..
266          ..
267          ..
268          ..
269          ..
270          ..
271          ..
272          ..
273          ..
274          ..
275          ..
276          ..
277          ..
278          ..
279          ..
280          ..
281          ..
282          ..
283          ..
284          ..
285          ..
286          ..
287          ..
288          ..
289          ..
290          ..
291          ..
292          ..
293          ..
294          ..
295          ..
296          ..
297          ..
298          ..
299          ..
300          ..
301          ..
302          ..
303          ..
304          ..
305          ..
306          ..
307          ..
308          ..
309          ..
310          ..
311          ..
312          ..
313          ..
314          ..
315          ..
316          ..
317          ..
318          ..
319          ..
320          ..
321          ..
322          ..
323          ..
324          ..
325          ..
326          ..
327          ..
328          ..
329          ..
330          ..
331          ..
332          ..
333          ..
334          ..
335          ..
336          ..
337          ..
338          ..
339          ..
340          ..
341          ..
342          ..
343          ..
344          ..
345          ..
346          ..
347          ..
348          ..
349          ..
350          ..
351          ..
352          ..
353          ..
354          ..
355          ..
356          ..
357          ..
358          ..
359          ..
360          ..
361          ..
362          ..
363          ..
364          ..
365          ..
366          ..
367          ..
368          ..
369          ..
370          ..
371          ..
372          ..
373          ..
374          ..
375          ..
376          ..
377          ..
378          ..
379          ..
380          ..
381          ..
382          ..
383          ..
384          ..
385          ..
386          ..
387          ..
388          ..
389          ..
390          ..
391          ..
392          ..
393          ..
394          ..
395          ..
396          ..
397          ..
398          ..
399          ..
400          ..
401          ..
402          ..
403          ..
404          ..
405          ..
406          ..
407          ..
408          ..
409          ..
410          ..
411          ..
412          ..
413          ..
414          ..
415          ..
416          ..
417          ..
418          ..
419          ..
420          ..
421          ..
422          ..
423          ..
424          ..
425          ..
426          ..
427          ..
428          ..
429          ..
430          ..
431          ..
432          ..
433          ..
434          ..
435          ..
436          ..
437          ..
438          ..
439          ..
440          ..
441          ..
442          ..
443          ..
444          ..
445          ..
446          ..
447          ..
448          ..
449          ..
450          ..
451          ..
452          ..
453          ..
454          ..
455          ..
456          ..
457          ..
458          ..
459          ..
460          ..
461          ..
462          ..
463          ..
464          ..
465          ..
466          ..
467          ..
468          ..
469          ..
470          ..
471          ..
472          ..
473          ..
474          ..
475          ..
476          ..
477          ..
478          ..
479          ..
480          ..
481          ..
482          ..
483          ..
484          ..
485          ..
486          ..
487          ..
488          ..
489          ..
490          ..
491          ..
492          ..
493          ..
494          ..
495          ..
496          ..
497          ..
498          ..
499          ..
500          ..
501          ..
502          ..
503          ..
504          ..
505          ..
506          ..
507          ..
508          ..
509          ..
510          ..
511          ..
512          ..
513          ..
514          ..
515          ..
516          ..
517          ..
518          ..
519          ..
520          ..
521          ..
522          ..
523          ..
524          ..
525          ..
526          ..
527          ..
528          ..
529          ..
530          ..
531          ..
532          ..
533          ..
534          ..
535          ..
536          ..
537          ..
538          ..
539          ..
540          ..
541          ..
542          ..
543          ..
544          ..
545          ..
546          ..
547          ..
548          ..
549          ..
550          ..
551          ..
552          ..
553          ..
554          ..
555          ..
556          ..
557          ..
558          ..
559          ..
560          ..
561          ..
562          ..
563          ..
564          ..
565          ..
566          ..
567          ..
568          ..
569          ..
570          ..
571          ..
572          ..
573          ..
574          ..
575          ..
576          ..
577          ..
578          ..
579          ..
580          ..
581          ..
582          ..
583          ..
584          ..
585          ..
586          ..
587          ..
588          ..
589          ..
590          ..
591          ..
592          ..
593          ..
594          ..
595          ..
596          ..
597          ..
598          ..
599          ..
600          ..
601          ..
602          ..
603          ..
604          ..
605          ..
606          ..
607          ..
608          ..
609          ..
610          ..
611          ..
612          ..
613          ..
614          ..
615          ..
616          ..
617          ..
618          ..
619          ..
620          ..
621          ..
622          ..
623          ..
624          ..
625          ..
626          ..
627          ..
628          ..
629          ..
630          ..
631          ..
632          ..
633          ..
634          ..
635          ..
636          ..
637          ..
638          ..
639          ..
640          ..
641          ..
642          ..
643          ..
644          ..
645          ..
646          ..
647          ..
648          ..
649          ..
650          ..
651          ..
652          ..
653          ..
654          ..
655          ..
656          ..
657          ..
658          ..
659          ..
660          ..
661          ..
662          ..
663          ..
664          ..
665          ..
666          ..
667          ..
668          ..
669          ..
670          ..
671          ..
672          ..
673          ..
674          ..
675          ..
676          ..
677          ..
678          ..
679          ..
680          ..
681          ..
682          ..
683          ..
684          ..
685          ..
686          ..
687          ..
688          ..
689          ..
690          ..
691          ..
692          ..
693          ..
694          ..
695          ..
696          ..
697          ..
698          ..
699          ..
700          ..
701          ..
702          ..
703          ..
704          ..
705          ..
706          ..
707          ..
708          ..
709          ..
710          ..
711          ..
712          ..
713          ..
714          ..
715          ..
716          ..
717          ..
718          ..
719          ..
720          ..
721          ..
722          ..
723          ..
724          ..
725          ..
726          ..
727          ..
728          ..
729          ..
730          ..
731          ..
732          ..
733          ..
734          ..
735          ..
736          ..
737          ..
738          ..
739          ..
740          ..
741          ..
742          ..
743          ..
744          ..
745          ..
746          ..
747          ..
748          ..
749          ..
750          ..
751          ..
752          ..
753          ..
754          ..
755          ..
756          ..
757          ..
758          ..
759          ..
760          ..
761          ..
762          ..
763          ..
764          ..
765          ..
766          ..
767          ..
768          ..
769          ..
770          ..
771          ..
772          ..
773          ..
774          ..
775          ..
776          ..
777          ..
778          ..
779          ..
780          ..
781          ..
782          ..
783          ..
784          ..
785          ..
786          ..
787          ..
788          ..
789          ..
790          ..
791          ..
792          ..
793          ..
794          ..
795          ..
796          ..
797          ..
798          ..
799          ..
800          ..
801          ..
802          ..
803          ..
804          ..
805          ..
806          ..
807          ..
808          ..
809          ..
810          ..
811          ..
812          ..
813          ..
814          ..
815          ..
816          ..
817          ..
8
```

Figura. 378 Configuración de la barra de navegación lateral pt3, Fuente. Elaboración propia.

```

198                                     COMPRAS
199             </div>
200         </a>
201     </li>
202     <li class="full-width divider-menu-h"></li>
203     <li class="full-width">
204         <a href="~/productos/productos.php" class="full-width">
205             <div class="navLateral-body-cl">
206                 <i class="zmdi zmdi-washing-machine"></i>
207             </div>
208             <div class="navLateral-body-cr">
209                 ENTRADAS
210             </div>
211         </a>
212     </li>
213     <li class="full-width divider-menu-h"></li>
214     <li class="full-width">
215         <a href="~/salidas/salidas.php" class="full-width">
216             <div class="navLateral-body-cl">
217                 <i class="zmdi zmdi-view-agenda"></i>
218             </div>
219             <div class="navLateral-body-cr">
220                 SALIDAS
221             </div>
222         </a>
223     </li>
224     <li class="full-width divider-menu-h"></li>
225     <li class="full-width">
226         <a href="~/inventario/inventario.php" class="full-width">
227             <div class="navLateral-body-cl">
228                 <i class="zmdi zmdi-store"></i>
229             </div>
230             <div class="navLateral-body-cr">
231                 INVENTARIO
232             </div>
233         </a>
234     </li>
235 </ul>
236 </nav>
237 </div>
238 </section>
239 <!-- contenido de pagina -->

```

Figura. 379 Configuración de la barra de navegación lateral pt4, Fuente. Elaboración propia.

Por parte del contenido de la página web se coloca una etiqueta “section” para agrupar de manera lógica y estructura las sentencias de código, la cual contendrá contenedores para aplicar estilos con css, además de continuar con las animaciones de la barra de navegación, se presenta la configuración para ocultar y mostrar la barra de navegación. Se programa la interfaz de cierre de sesión mediante la implementación de un botón con identificador. Posteriormente como se muestra en la figura 380, se coloca una imagen con el avatar de identificación del administrador y se manda a llamar los datos propios de este, desde la base de datos Firebase.

```

239 <!-- contenido de pagina -->
240 <section class="full-width pageContent">
241 <!-- navBar -->
242 <div class="full-width navBar">
243 <div class="full-width navBar-options">
244 <i class="zmdi zmdi-swap btn-menu" id="btn-menu"></i>
245 <div class="mdl-tooltip" for="btn-menu">Ocultar/Mostar Menu</div>
246 <nav class="navBar-options-list">
247 <ul class="list-unstyled">
248 <li class="btn-exit" id="btn-exit">
249 <i class="zmdi zmdi-power"></i>
250 <div class="mdl-tooltip" for="btn-exit">Cerrar Sesión</div>
251 </li>
252 <li class="text-condensedLight noLink"><small><small><?php echo $_SESSION['usuario']["NombreCompleto"]></small></small><?php echo $_SESSION['usuario']["Apellido"]></small></li>
253 <li class="noLink">
254 <figure>
255 
256 </figure>
257 </li>
258 </ul>
259 </nav>
260 </div>
261 </div>
262

```

Figura. 380 Creación de los botones ocultar/mostrar y cerrar sesión. Fuente. Elaboración propia.

Las animaciones correspondientes al cierre de sesión y ocultar/mostrar la barra de navegación son estructuradas en el archivo “marco.js” ubicado en la carpeta assets/js, así como se observa en la figura 381.

```

1 (document).ready(function () {
2   /*Mostrar ocultar menu principal*/
3   $('btn-menu').on('click', function () {
4     var navLateral = $('navLateral');
5     var pageContent = $('pageContent');
6     var navOption = $('navBar-options');
7     if (navLateral.hasClass('navLateral-change') && pageContent.hasClass('pageContent-change')) {
8       navLateral.removeClass('navLateral-change');
9       pageContent.removeClass('pageContent-change');
10      navOption.removeClass('navBar-options-change');
11    } else {
12      navLateral.addClass('navLateral-change');
13      pageContent.addClass('pageContent-change');
14      navOption.addClass('navBar-options-change');
15    }
16  });
17  /*Salir del sistema*/
18  $('btn-exit').on('click', function () {
19    swal({
20      title: "Quieres salir del sistema?",
21      text: "La sesión actual se cerrará",
22      type: "warning",
23      showCancelButton: true,
24      confirmButtonText: "Salir",
25      closeOnConfirm: false
26    },
27    function (isConfirm) {
28      if (isConfirm) {
29        window.location = "php/cerrar_sesion.php";
30      }
31    });
32  });
33  /*Mostrar y ocultar submenus*/
34  $('btn-submenu').on('click', function () {
35    var subMenu = $(this).next('ul');
36    var icon = $(this).children('span');
37    if (subMenu.hasClass('sub-menu-options-show')) {
38      subMenu.removeClass('sub-menu-options-show');
39      icon.addClass('zmdi-chevron-left').removeClass('zmdi-chevron-down');
40    } else {
41      subMenu.addClass('sub-menu-options-show');
42      icon.addClass('zmdi-chevron-down').removeClass('zmdi-chevron-left');
43    }
44  });
45

```

Figura. 381 Creación del contenedor de indicaciones para el formulario. Fuente. Elaboración propia.

Posteriormente como se muestra en la figura 382, se coloca una etiqueta de section con sus respectivos contenedores para dar formato a la descripción del formulario.

```

264 <section class="full-width header-well">
265 <div class="full-width header-well-icon">
266 <i class="zmdi zmdi-card"></i>
267 </div>
268 <div class="full-width header-well-text">
269 <p class="text-condensedLight">
270 Ingreso de nuevas Ubicaciones,
271 Nota: Recuerda llenar todos los campos de manera adecuada.
272 </p>
273 </div>
274 </section>
275

```

Figura. 382 Creación del contenedor de indicaciones para el formulario. Fuente. Elaboración propia.

A su vez como se muestra en la figura 383, se coloca un nuevo contenedor el cual contendrá dos enlaces que permitirán navegar dentro de la misma página, uno para visualizar el contenido de los registros del formulario en una tabla y el segundo que muestra el formulario a ser completado.

```

272      Nota: Recuerda llenar todos los campos de manera adecuada.
273    </p>
274  </div>
275 </section>
276 <div class="mdl-tabs mdl-js-tabs mdl-js-ripple-effect">
277   <div class="mdl-tabs__tab-bar">
278     <a href="#tabNewPayment" class="mdl-tabs__tab is-active">NUEVO</a>
279     <a href="#tabListPayment" class="mdl-tabs__tab">LISTA</a>
280   </div>
281   <div class="mdl-tabs__panel is-active" id="tabNewPayment">
282     <div class="mdl-grid">

```

Figura. 383 Creación de enlaces navegadores del archivo ubicaciones.php. Fuente. Elaboración propia.

Por un lado, como se muestra en la figura 384, una vez colocados los enlaces se crea un contenedor donde se define el título del formulario y se expresa en texto los datos que deberán ser llenados a continuación.

```

281   <div class="mdl-tabs__panel is-active" id="tabNewPayment">
282     <div class="mdl-grid">
283       <div class="mdl-cell mdl-cell--12-col">
284         <div class="full-width panel mdl-shadow--2dp">
285           <div class="full-width panel-title bg-primary text-center titles">
286             Nuevo Proyecto
287           </div>
288           <div class="full-width panel-content">
289             <form action="action_agregar.php" method="POST">
290               <div class="mdl-grid">
291                 <div class="mdl-cell mdl-cell--12-col">
292                   <legend class="text-condensed-light"><i class="zmdi zmdi-border-color"></i> &nbsp; DATOS DEL PROYECTO</legend><br>
293                   </div>

```

Figura. 384 Creación de título del formulario e indicaciones, Fuente. Elaboración propia.

Así mismo como se muestra en la figura 35, se crean nuevos contenedores que almacenaran los inputs necesarios para el registro de información

```

296       <input class="mdl-textfield__input" type="text" pattern="
297       -[A-Za-z0-9áéíóúÁÉÍÓÚ ]*(\.[0-9]+)?" name="Nombre_del_anaquel">
298       <label class="mdl-textfield__label" for="Nombre_del_anaquel">Nombre del
299       Anaquel </label>
300       <span class="mdl-textfield__error">Correo Invalido</span>
301     </div>
302     <div class="mdl-cell mdl-cell--12-col">
303       <div class="mdl-textfield mdl-js-textfield">
304         <select class="mdl-textfield__input" id="nivel select" name="Niveles">
305           <option value="">Selecciona el Nivel</option>
306         </select>
307       </div>
308     </div>
309     <div class="text-center">
310       <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
311       mdl-button--colored bg-primary" id="btn-addPayment">
312         <i class="zmdi zmdi-plus"></i>
313       </button>
314       <div class="mdl-tooltip" for="btn-addPayment">Agregar Ubicacion</div>
315     </div>

```

Figura. 385 Creación de los campos para el formulario de ubicaciones, Fuente. Elaboración propia.

Siendo de esta manera que la estructura de cada input se definirá por el tipo de dato a almacenar, ya sea numérico, texto, correo o contraseña. Se definen con la expresión regular de pattern para indicarle el tipo de simbología que deberá de contener, esto no aplica para los campos de correo y contraseña ya que poseen una estructura específica, a su vez se coloca el formato las listas de selección las cuales pueden contener información predestinada mediante etiquetas “option” o en su caso incluir un archivo js implementando información desde la base de datos. En caso de incumplir dicha estructura se colocará abajo del input una alerta de texto erróneo, así como se muestra en la figura 386.

```
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
  <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label">
    <input class="mdl-textfield__input" type="text" pattern="-?[A-Za-záéíóúÁÉÍÓÚ]*(\.[0-9]+)?" name="Nombrecompleto">
    <label class="mdl-textfield__label" for="Nombrecompleto">Nombre(s)</label>
    <span class="mdl-textfield__error">Nombre invalido</span>
  </div>
</div>
<div class="mdl-cell mdl-cell--6-col mdl-cell--8-col-tablet">
```

Figura. 386 Codificación de la estructura de los campos para el formulario, Fuente. Elaboración propia.

Se muestra en la figura 387, al finalizar las inserciones de los input se crea un botón dentro de un contenedor que comenzará el procesamiento de información del formulario.

```
305     </select>
306   </div>
307 </div>
308 </div>
309 <div class="text-center">
310   <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
mdl-button--colored bg-primary" id="btn-addPayment">
    <i class="zmdi zmdi-plus"></i>
  </button>
313   <div class="mdl-tooltip" for="btn-addPayment">Agregar Ubicación</div>
314 </div>
315 </form>
316 </div>
317 </div>
318 </div>
319 </div>
320 </div>
```

Figura. 387 Creación del botón agregar ubicación, Fuente. Elaboración propia.

Se observa en la figura 388, para que el procesamiento de datos este completo el uso del archivo “action_agregar.php” cumple con la función de verificar el tiempo de inactividad dentro de la página y almacenar los datos obtenidos de los inputs y listas de selección en variables, las cuales serán evaluadas mediante sentencias condicionales identificando si la variable recibió información o esta es nula, si a información obtenida resulta ser nula se regresará al archivo “ubicaciones.php”.

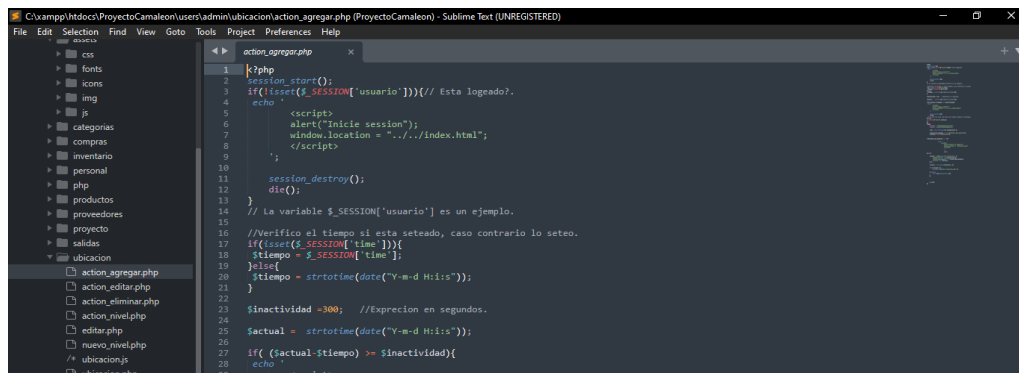


Figura. 388 Codificación de tiempo de inactividad del archivo `action_agregar.php`, Fuente. Elaboración propia.

Dentro del archivo `ubicaciones.js` se hace la conexión a la base de datos y se mandan llamar a las tablas necesarias para cargar y almacenar la información en las listas mediante el uso de variables, como se muestra en la figura 389. Por lo tanto, si la información es correcta se mandará a llamar a la tabla de ubicación.

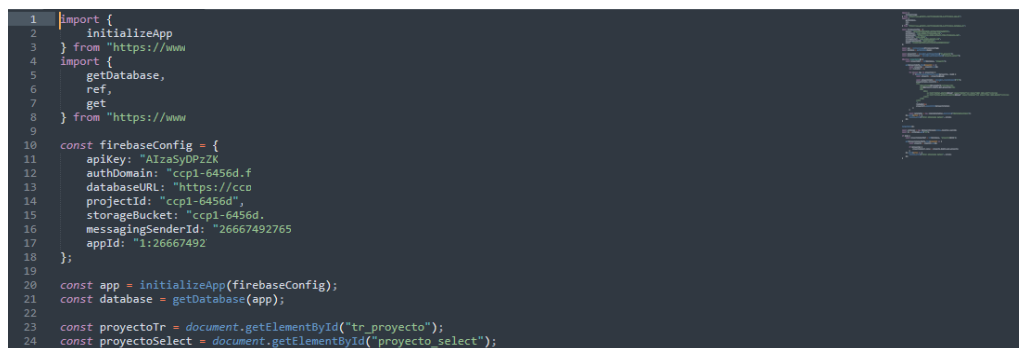


Figura. 389 Conexión a la base de datos e incorporación de registros en listas del formulario, Fuente. Elaboración propia.

Como se muestra en la figura 390, si la información es correcta se verificará por medio de sentencias de decisión si es que los datos proporcionados no están almacenados dentro de las tablas de la base de datos, si están almacenados por medio de un script regresara una alerta y redirigirá al archivo `ubicaciones.php`

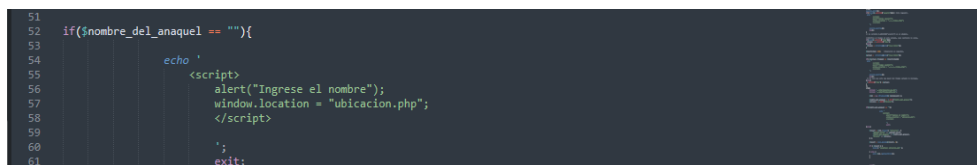


Figura. 390 Conexión a la base de datos y verificación de campos, Fuente. Elaboración propia.

Por otro lado, en caso de no estar almacenados, se insertará por medio de una variable que almacenará los datos correctos y devolverá al archivo ubicaciones.php, como se observa en la figura 391.

```

64 $insert = $rdb->insert( "ubicacion", [
65     "Agregar_nivel" => $agregar_nivel,
66     "Nombre_del_anaquel" => $nombre_del_anaquel,
67     "Niveles" => $niveles,
68 ] );
69
70 $result = json_decode($insert, 1);
71
72 if ( $insert ) {
73     header( "Location: ubicacion.php" );
74 } else {
75     echo $rdb->getLastError();
76 }
77
78

```

Figura. 391 Codificación de ingreso de datos a la tabla ubicación. Fuente. Elaboración propia.

Una vez realizado el ingreso de información se coloca un contenedor el cual almacenara el título de la lista de registros y se da formato a la tabla de registros con los campos acorde a los inputs y registros de la base de datos, se asignan identificadores para dar formato mediante css y hacer uso de Java Script, como se muestra en la figura 392.

```

321 <div class="full-width panel-content">
322 <div class="table-responsive text-center">
323 <table class="table table-bordered table-hover" id="datatablesSimple">
324 <thead>
325 <tr>
326 <th scope="col"><strong>ID</strong></th>
327 <th scope="col"><strong>Nombre del Proyecto</strong></th>
328 <th scope="col"><strong>Acción</strong></th>
329 </tr>
330 </thead>
331 <tbody id="tr_proyecto">
332 </tbody>
333 </table>
334 <button id="btnExportar" class="btn btn-success">
335 <i class="fas fa-file-excel"></i> Exportar datos a Excel
336 </button>

```

Figura. 392 Creación de la tabla de registros actualizados para ubicaciones, Fuente. Elaboración propia.

Para llenar la tabla de los registros se ocupa el archivo “ubicacion.js”, como se muestra en la figura 393, el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros de cada insert para almacenarlos y mostrarlos en el orden indicado con anterioridad.

```

1  import {
2    initializeApp
3  } from "https://www.
4  import {
5    getDatabase,
6    ref,
7    get
8  } from "https://www.
9
10 const firebaseConfig = {
11   apiKey: "AIzaSyDPz7K
12   authDomain: "ccpl-6456d.1
13   databaseURL: "https://c
14   projectId: "ccpl-6456d.
15   storageBucket: "ccpl-6456d.1
16   messagingSenderId: "26667492765
17   appId: "1:266674927
18 };
19
20 const app = initializeApp(firebaseConfig);
21 const database = getDatabase(app);
22
23 const proveedorTr = document.getElementById("tr_proveedores");
24 const paisSelect = document.getElementById("pais_select");
25
26 function CargarDatos() {
27   const proveedorRef = ref(database, "proveedores");
28
29   get(proveedorRef).then(snapshot => {
30     const proveedores = snapshot.val();
31     let rowNumber = 1;
32
33     for (const key in proveedores) {
34       if (Object.hasOwnProperty.call(proveedores, key)) {
35         const proveedor = proveedores[key];
36
37         const proveedorTable = document.createElement("tr");
38         proveedorTable.innerHTML = "
39           <tr>
40             <td><strong>${rowNumber}</strong></td>
41             <td>${proveedor.Nombre}</td>

```

Figura. 393 Conexión a la base de datos y incorporación de registros en la tabla ubicaciones, Fuente. Elaboración propia.

En la figura 394 dentro de la tabla de visualización de registro se coloca un botón que permitirá la descarga y conversión de los registros de la tabla a un formato tipo Excel, mediante el uso de un script colocado posteriormente del llamado “ubicación.js”.

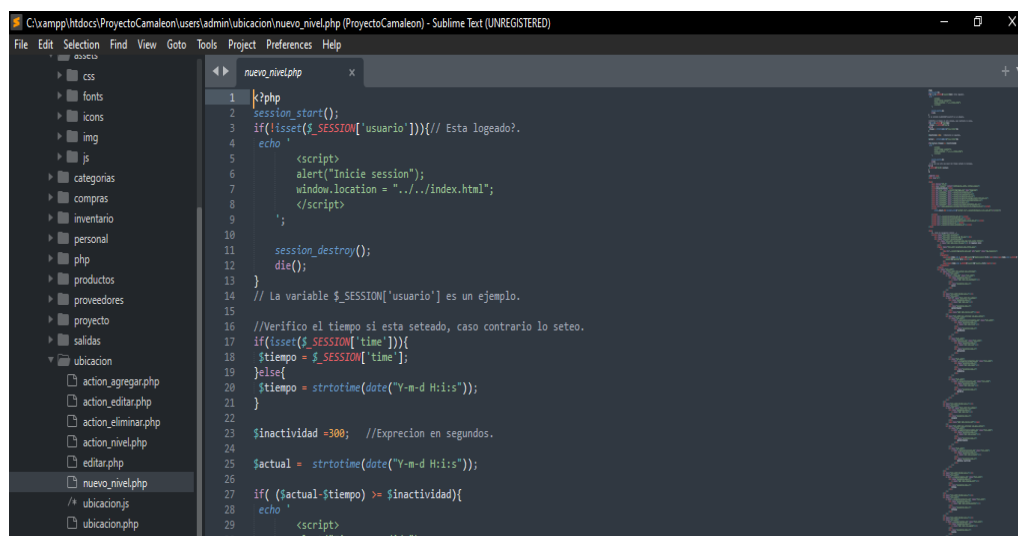
```

425   <button id="btnExportar" class="btn btn-success">
426     <i class="fas fa-file-excel"></i> Exportar datos a Excel
427   </button>
428 </div>
429 </div>
430 </div>
431 </div>
432 </div>
433 </div>
434 </div>
435 </section>
436
437 <script type="module" src="proveedores.js"></script>
438
439 <!-- script para exportar a excel -->
440 <script>
441   const $btnExportar = document.querySelector("#btnExportar");
442   $datatablesSimple = document.querySelector("#datatablesSimple");
443
444   $btnExportar.addEventListener("click", function() {
445     let tableExport = new TableExport($datatablesSimple, {
446       exportButtons: false, // No queremos botones
447       filename: "Reporte de Proveedores", //Nombre del archivo de Excel
448       sheetname: "Reporte de Proveedores", //Titulo de la hoja
449     });
450     let datos = tableExport.getExportData();
451     let preferenciasDocumento = datos.datatablesSimple.xlsx;
452     tableExport.export2file(preferenciasDocumento.data, preferenciasDocumento.mimeType, preferenciasDocumento.filename,
453       preferenciasDocumento.fileExtension, preferenciasDocumento.merges, preferenciasDocumento.RTL,
454       preferenciasDocumento.sheetname);
455   });
456 </script>
457 </body>
458 </html>

```

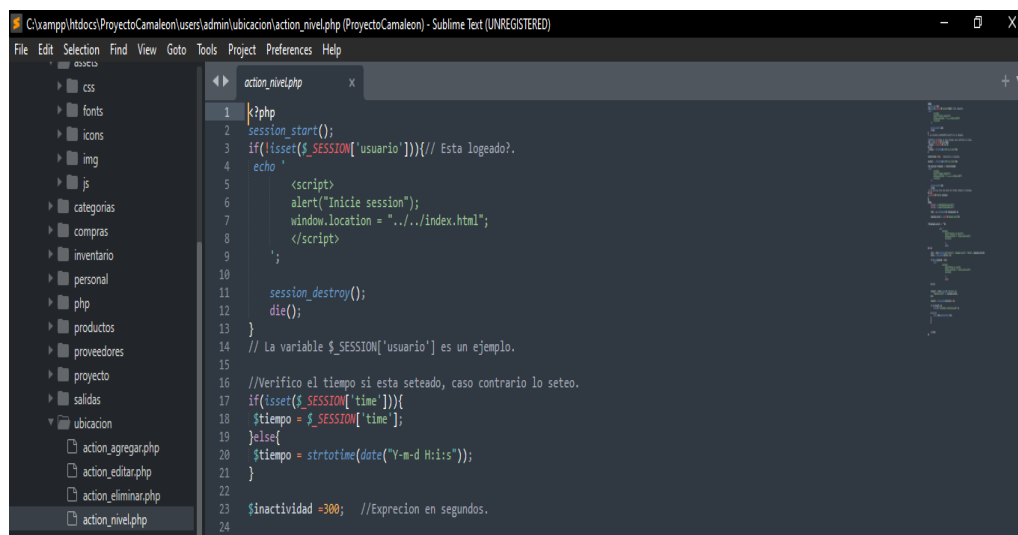
Figura. 394 Codificación y conversión de registros a formato Excel, Fuente. Elaboración propia.

Posteriormente como se muestra en las figuras 395 y 396, se encuentra el archivo “nuevo_nivel” que contendrá un formulario para la inserción de nuevos niveles dentro de la tabla de ubicación en la base de datos, se hace uso del archivo action_nivel como guía de conexión y almacenamiento hacia la base de datos.



```
1 <?php
2 session_start();
3 if(!isset($_SESSION['usuario'])){// Esta logeado?.
4     echo '
5     <script>
6         alert("Inicie session");
7         window.location = "../index.html";
8     </script>
9 '
10
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Exprecion en segundos.
24
25 $actual = strtotime(date("Y-m-d H:i:s"));
26
27 if( ($actual-$tiempo) >= $inactividad){
28     echo '
29     <script>
30         alert("Tiempo expirado");
```

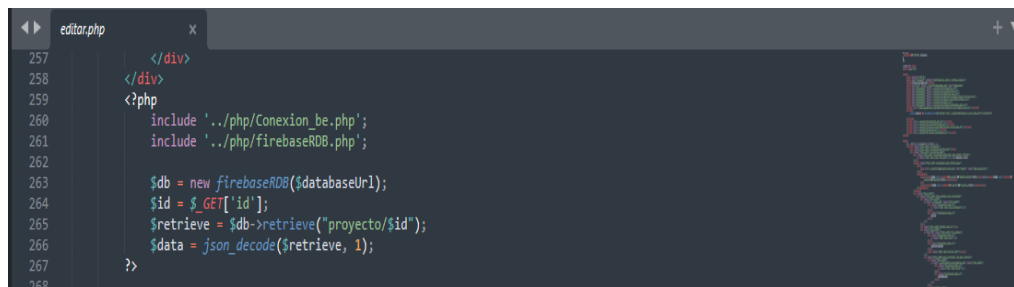
Figura. 395 Codificación de tiempo de inactividad del archivo nuevo_nivel.php. Fuente. Elaboración propia.



```
1 <?php
2 session_start();
3 if(!isset($_SESSION['usuario'])){// Esta logeado?.
4     echo '
5     <script>
6         alert("Inicie session");
7         window.location = "../index.html";
8     </script>
9 '
10
11     session_destroy();
12     die();
13 }
14 // La variable $_SESSION['usuario'] es un ejemplo.
15
16 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17 if(isset($_SESSION['time'])){
18     $tiempo = $_SESSION['time'];
19 }else{
20     $tiempo = strtotime(date("Y-m-d H:i:s"));
21 }
22
23 $inactividad =300; //Exprecion en segundos.
24
```

Figura. 396 Codificación de tiempo de inactividad del archivo action_nivel.php. Fuente, Elaboración propia.

En relación con el archivo ubicaciones.php se encuentra “editar.php” perteneciente a la misma carpeta, se encuentran las configuraciones iniciales de la barra de navegación, no obstante, como se muestra en la figura 397, se hace un llamado a la conexión de la base de datos dentro de la interpretación de php, se especifica la tabla a la cual se tendrá acceso de información.

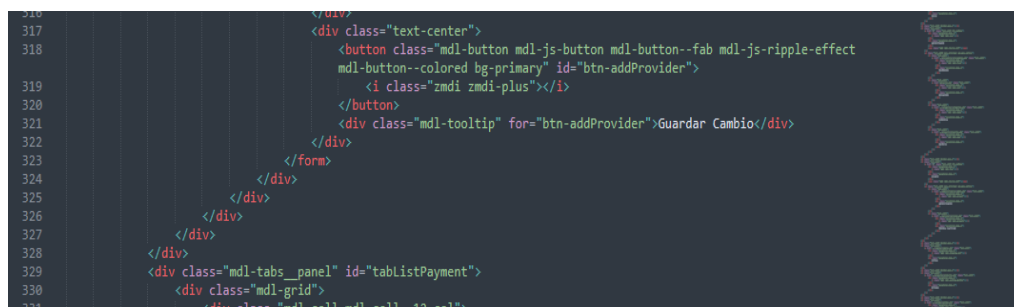


```

257     </div>
258 </div>
259 <?php
260     include '../php/Conexion_be.php';
261     include '../php/firebaseRDB.php';
262
263     $db = new firebaseRDB($databaseUrl);
264     $id = $_GET['id'];
265     $retrieve = $db->retrieve("proyecto/$id");
266     $data = json_decode($retrieve, 1);
267
268 >>
    
```

Figura. 397 Creación y configuración del archivo editar.php, Fuente. Elaboración propia.

Se coloca un botón permitiendo la redirección al archivo “action_editar” continuando con el proceso de actualización de datos del formulario, como se observa en la figura 398.



```

310     </div>
311 <div class="text-center">
312     <button class="mdl-button mdl-js-button mdl-button--fab mdl-js-ripple-effect
313         mdl-button--colored bg-primary" id="btn-addProvider">
314         <i class="zmdi zmdi-plus"></i>
315     </button>
316     <div class="mdl-tooltip" for="btn-addProvider">Guardar Cambio</div>
317 </div>
318 </form>
319 </div>
320 </div>
321 </div>
322 </div>
323 </div>
324 </div>
325 </div>
326 </div>
327 </div>
328 </div>
329 <div class="mdl-tabs_panel" id="tabListPayment">
330 <div class="mdl-grid">
331 <div class="mdl-cell mdl-cell--12-col">
    
```

Figura. 398 Creación del botón guardar cambio, Fuente. Elaboración propia.

Así que como se muestra en la figura 399, al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad.



```

10     session_destroy();
11     die();
12 }
13 // La variable $_SESSION['usuario'] es un ejemplo.
14
15 //Verifico el tiempo si esta seteado, caso contrario lo seteo.
16 if(isset($_SESSION['time'])){
17     $tiempo = $_SESSION['time'];
18 }else{
19     $tiempo = strtotime(date("Y-m-d H:i:s"));
20 }
21
22 $inactividad = 300; //Exprecion en segundos.
23
24 $actual = strtotime(date("Y-m-d H:i:s"));
25
26 if( ($actual-$tiempo) >= $inactividad){
27     echo '
28     <script>
29     alert("Tiempo excedido");
30     window.location = "../index.html";
31     </script>
32     '
33
34     session_destroy();
35     die();
36 // En caso que este sea mayor del tiempo seteado lo deslogea.
37 }else{
38     $_SESSION['time'] = $actual;
39 }
40 }
41 >>
42 <?php
    
```

Figura. 399 Codificación de tiempo de inactividad del archivo action_editar.php. Fuente. Elaboración propia.

Se hace el llamado a la conexión de la base de datos y creación de variables que almacenaran los cambios indicados en los input y listas, como se muestra en la figura 400, a su vez se verifica que dichos inputs contengan información ya que al no contenerla se coloca una alerta y redireccionamiento al archivo “ubicaciones.php”.

```

40 }
41 }>
42 <?php
43 include '../php/Conexion_be.php';
44 include '../php/firebaseRDB.php';
45
46 $rdb = new firebaseRDB( $databaseUrl );
47
48 $uid = $_POST['uid'];
49
50 $nombre_del_proyecto = $_POST['Nombre_del_proyecto'];
51
52
53 if($nombre_del_proyecto == ""){
54
55     echo '
56         <script>
57             alert("Ingrese el nombre");
58             window.location = "proyectos.php";
59         </script>

```

Figura. 400 Codificación de verificación para datos repetidos de los campos. Fuente. Elaboración propia.

En cambio, como se observa en la figura 401, al proceso de ingreso de información, se anula la evaluación de datos repetidos y se comienza el proceso de actualización de datos, el cual una vez finalizado dará redireccionamiento al archivo ubicaciones.php.

```

63 }else{
64     $update = $rdb->update( "proyecto", $uid, [
65         "Nombre_del_proyecto" => $nombre_del_proyecto,
66     ] );
67
68     if ( $update ) {
69         header( "Location: proyectos.php" );
70
71     } else {
72         echo $rdb->getLastError();
73     }
74
75
76     exit();
77 }>

```

Figura. 401 Codificación de verificación para datos repetidos de los campos, Fuente. Elaboración propia

Se muestra en la figura 402 el llenado de la tabla de los registros se ocupa el archivo “ubicacion.js” el cual importa la configuración de conexión de Firebase a la base de datos y manda a llamar a los campos de los registros que fueron actualizados y mostrarlos en el orden indicado con anterioridad.

```

327 </div>
328 <div class="full-width panel-content">
329     <div class="table-responsive text-center">
330         <table class="table table-bordered table-hover" id="datatablesSimple">
331             <thead>
332                 <tr>
333                     <th scope="col"><strong>ID</strong></th>
334                     <th scope="col"><strong>Nombre del Proyecto</strong></th>
335                     <th scope="col"><strong>Acción</strong></th>
336                 </tr>
337             </thead>
338             <tbody id="tr_proyecto">
339             </tbody>
340         </table>

```

Figura. 402 Codificación de ingreso de datos a la tabla ubicaciones, Fuente. Elaboración propia.

Por otra parte, como se muestra en la figura 403, se encuentra el archivo “action_eliminar” ubicado dentro de la misma carpeta de administración, Al igual que el archivo action_agregar se conservan las instrucciones sobre inactividad.

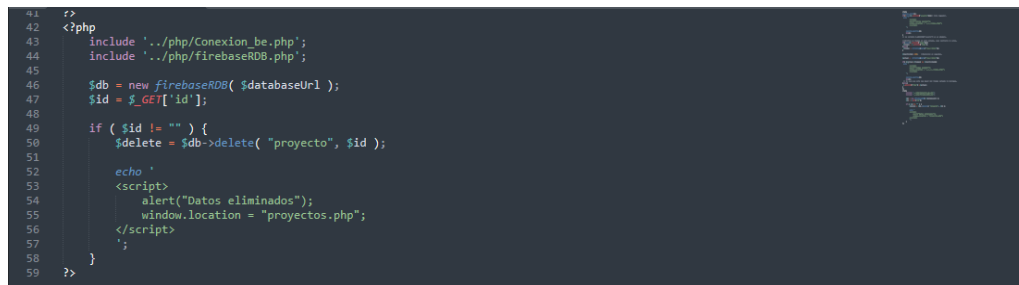


```

1  <?php
2  session_start();
3  if(isset($_SESSION['usuario'])){// Esta logeado?.
4      echo '
5          <script>
6              alert("Inicie session");
7              window.location = "../index.html";
8          </script>
9      ';
10
11      session_destroy();
12      die();
13  }
14  // La variable $_SESSION['usuario'] es un ejemplo.
15
16  //Verifico el tiempo si esta seteado, caso contrario lo seteo.
17  if(isset($_SESSION['time'])){
18      $tiempo = $_SESSION['time'];
19  }else{
20      $tiempo = strtotime(date("Y-m-d H:i:s"));
21  }
22
23  $inactividad =300; //Expresion en segundos.
24
25  $actual = strtotime(date("Y-m-d H:i:s"));
26
27  if( ($actual-$tiempo) >= $inactividad){
28      echo '
29          <script>
30              alert("Tiempo excedido");
31              window.location = "../index.html";
32          </script>
33      ';
34  }
    
```

Figura. 403 Codificación de tiempo de inactividad del archivo action_eliminar.php. Fuente. Elaboración propia.

Se hace el llamado a la conexión de la base de datos y se comienza el borrado de registro mediante una sentencia de evaluación que hace uso del identificador del registro y la variable delete, una vez finalizada la eliminación se coloca una alerta sobre la eliminación y se redirecciona al archivo “ubicaciones.php”, así como se muestra en la figura 404.

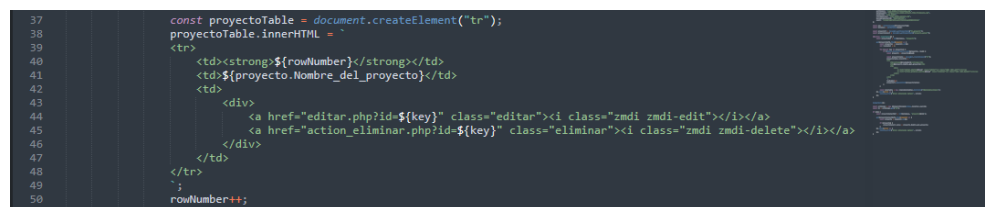


```

41  <?php
42
43  include '../php/Conexion_be.php';
44  include '../php/firebaseRDB.php';
45
46  $db = new firebaseRDB( $databaseUrl );
47  $id = $_GET['id'];
48
49  if ( $id != "" ) {
50      $delete = $db->delete( "proyecto", $id );
51  }
52
53  echo '
54      <script>
55          alert("Datos eliminados");
56          window.location = "proyectos.php";
57      </script>
58  ';
59  }
    
```

Figura. 404 Codificación y eliminación de registros de la tabla ubicaciones. Fuente. Elaboración propia.

En la figura 405 se muestra que para que las funcionalidades de edición y eliminación empiecen su proceso, dentro de la tabla de visualización de registros se visualizan 2 botones de acción para cada proceso correspondiente, dichos botones son ubicados en el archivo “ubicacion.js”.



```

37  const proyectoTable = document.createElement("tr");
38  proyectoTable.innerHTML =
39      <tr>
40          <td><strong>${rowNumber}</strong></td>
41          <td>${proyecto.Nombre_del_proyecto}</td>
42          <td>
43              <div>
44                  <a href="editar.php?id=${key}" class="editar"><i class="mdi mdi-edit"></i></a>
45                  <a href="action_eliminar.php?id=${key}" class="eliminar"><i class="mdi mdi-delete"></i></a>
46              </div>
47          </td>
48      </tr>
49      ';
50  rowNumber++;
    
```

Figura. 405 Codificación de vínculos para los botones eliminar y editar.Fuente. Elaboración propia.

3.10.16.Carpeta Usuario:

Como se muestra en la figura 406, la carpeta usuario se encuentra el desarrollo de compras, inventario, salidas y home, así que en este caso al ser paginas para el uso personal de la empresa se limitan las funcionalidades de estas permitiendo solamente pocas interacciones con el sistema y no involucrando al usuario más de lo debido.

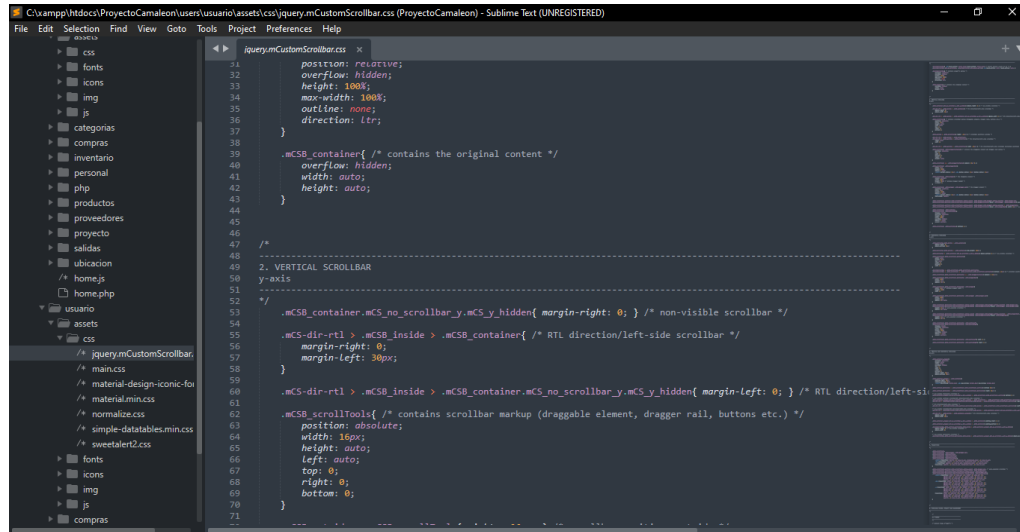


Figura. 406 Contenido de la carpeta usuario. Fuente. Elaboración propia.

CAPÍTULO IV: RESULTADOS

4. Resultados

En la siguiente sección se muestran los resultados obtenidos después del análisis y desarrollo del sistema, a partir de la implementación y evaluación de este. Se llevo a cabo el objetivo principal el cual fue; Desarrollar un sistema web de inventario para gestionar las entradas, salidas y compras de pinturas en el departamento de calidad y almacén de CC camaleón color., una solución diseñada para la empresa CC Camaleón Color por lo que los resultados presentados aquí se alinean con los objetivos establecidos al inicio del proyecto.

4.1.Prueba de Funcionalidad

La prueba de funcionalidad se enfoca en los requerimientos para verificar que corresponden directamente a casos de usos o funciones y reglas del negocio. Los objetivos de estas pruebas son verificar la aceptación de los datos, el proceso, la recuperación y la implementación correcta de las reglas de la empresa, para su uso en producción.

4.2. Objetivo de la prueba

Asegurar la funcionalidad apropiada del objeto de prueba, incluyendo la navegación, entrada de datos, proceso y recuperación.

4.2.1. Técnica

Ejecutar los casos de uso usando datos válidos y no válidos, para verificar lo siguiente:

- Se obtienen los resultados esperados cuando se usan datos válidos.
- Cuando se usan datos no válidos se despliegan los mensajes de error o advertencia apropiados.

4.2.2. Criterio de aceptación

Todas las pruebas planificadas se realizaron. Todos los defectos encontrados han sido debidamente identificados.

Tabla 17 muestra la prueba de la función autenticarse.

Tabla 17 Prueba de autenticarse. Elaboración propia.

DESARROLLO DE PRUEBA: Autenticar.	
Objetivo de la prueba	Verificar el funcionamiento del inicio de sesión de un usuario en el sistema web de inventarios
Descripción de la prueba y valor esperado	La prueba consiste en el proceso de inicio de sesión de un usuario en el sistema, que previamente de haya registrado por el administrador. Se ingresarán los datos requeridos y necesarios para esta función las cuales son Correo electrónico y contraseña, por lo que si el usuario no se registra previamente no será posible iniciar sesión. Se espera que la aplicación verifique los campos obligatorios, valide el correo electrónico y la contraseña. El valor esperado es que el usuario pueda acceder exitosamente a la aplicación.
Técnica	1. Acceder al URL proporcionado para que cargue el inicio de sesión directamente en la página web 2. Acceder en la pantalla principal de inicio de sesión. 3. Completar el formulario de inicio con correo electrónico y contraseña. 4. Dar clic en iniciar sesión. 5. Verificar que se complete el proceso inicio de sesión exitosamente y se redirija al usuario a la pantalla de bienvenida y visualice el menú.
Proceso de medición	-Inicio de sesión exitoso: El usuario puede completar el proceso de autenticarse sin errores y se redirige a la pantalla principal con el menú de opciones. -Registro no exitoso: No se encontraron errores durante el proceso de autenticarse, el usuario pudo ingresar exitosamente.
Criterios de terminación/valor obtenido	La prueba se considera exitosa ya que el usuario puede acceder al portal web y entrar al menú principal.
Consideraciones especiales	Se necesita una conexión a internet estable para realizar la prueba. Se aseguró que la versión del navegador de internet sea la más actualizada.
Interpretación de resultados	La prueba fue exitosa, el usuario pudo completar el proceso de inicio de sesión sin errores. Se verificó que los campos obligatorios estén completos y que el correo electrónico sea válido Además de que se inicia sesión en un tiempo menor a 5 segundos. Al finalizar el proceso, se redirigió al usuario a la página de menú principal y entra al sistema.
Corrección de error	No se encontraron errores durante el proceso de inicio de sesión. La prueba se realizó y se comprobó que el módulo de autenticación funciona correctamente. No se requiere corrección de errores.

PANTALLA DE EVIDENCIA DE EJECUCIÓN

Figura 407 muestra la prueba de inicio de sesión.

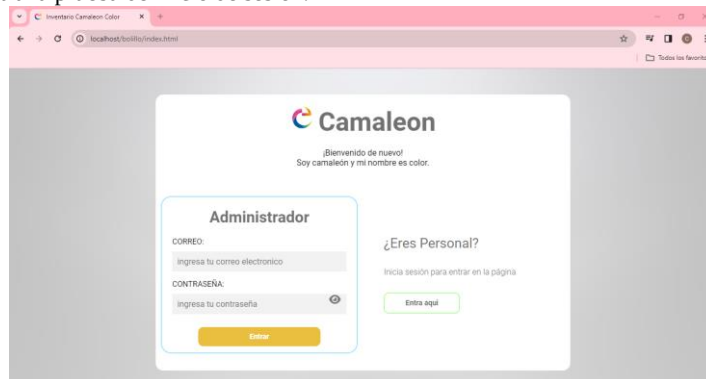


Figura. 407 Prueba de inicio de sesión. Fuente elaboración propia.

La tabla 18 muestra la tabla de prueba, registro de un producto

Tabla 18 Prueba, registro de producto. Fuente. Elaboración propia.

DESARROLLO DE PRUEBA: Registrar producto.	
Objetivo de la prueba	Verificar el funcionamiento del registro de un nuevo producto
Descripción de la prueba y valor esperado	La prueba consiste en el proceso de registro de un nuevo producto en el sistema. Se ingresarán los datos requeridos, y se enviará el formulario de registro. Se espera que el sistema verifique los campos obligatorios, El valor esperado es que el usuario pueda completar el registro de un nuevo producto.
Técnica	1. Abrir e iniciar sesión al sistema de inventario web. 2. Acceder a la sección de registro de entradas 3. Completar el formulario de registro de nuevo producto. 4. Una vez completado del formulario mostrara un mensaje de registro exitoso.
Proceso de medición	-Registro exitoso: El usuario puede completar el proceso de registro sin errores y se redirige a la pantalla de registrar nuevo producto nuevamente, -Registro no exitoso: No se encontraron errores durante el proceso de registro y el usuario recibió el correo de confirmación.
Criterios de terminación/valor obtenido	La prueba se considera exitosa ya que el usuario puede completar el proceso de registro sin errores recibiendo el correo de confirmación y se redirige correctamente a la pantalla de inicio de sesión.
Consideraciones especiales	-Se necesita una conexión a internet estable para realizar la prueba.

Interpretación de resultados	La prueba fue exitosa, ya que el usuario pudo completar el registro sin errores. Se verificó que los campos obligatorios estén completos y que el correo electrónico sea válido con una contraseña mayor a 6 caracteres. Además, el usuario recibió el correo de confirmación y pudo verificar su cuenta dentro del plazo de 15 segundos. Al finalizar el proceso, se redirigió al usuario a la pantalla de inicio de sesión.
Corrección de error	No se encontraron errores durante el proceso de registro. La prueba se realizó sin inconvenientes y se comprobó que el módulo de registro de nuevos usuarios funciona

PANTALLA DE EVIDENCIA DE EJECUCIÓN

En las figuras 408 y 409 se muestra la ejecución de la prueba de registro de producto

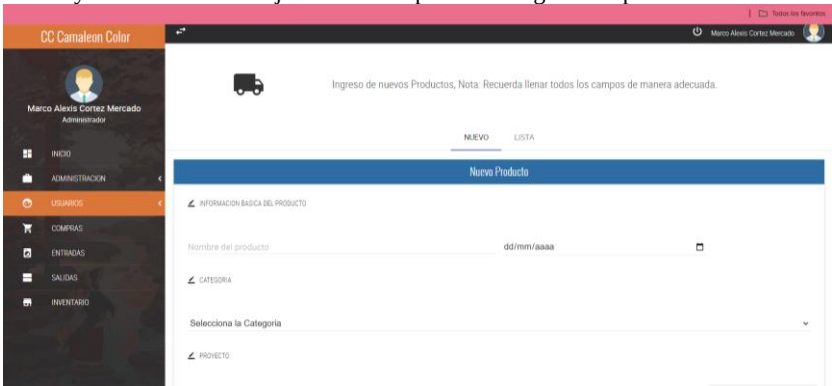


Figura. 408 Pruebas de registro de producto. Fuente. Elaboración propia.

Una vez registrado se muestra en la lista de productos:

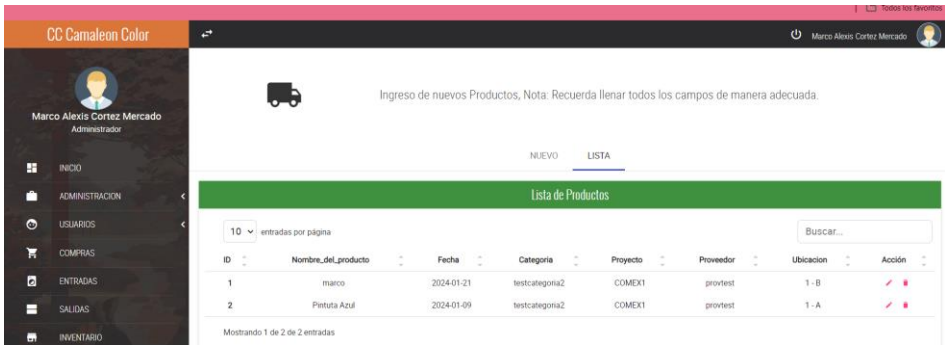


Figura. 409 Prueba de lista de productos agregados. Fuente. Elaboración propia.

En la tabla 19 se detallan las pruebas de editar un producto.

Tabla 19 Prueba de editar producto. Elaboración propia.

DESARROLLO DE PRUEBA: Editar producto.	
Objetivo de la prueba	Verificar el funcionamiento editar un producto en la aplicación móvil de Sistema de inventario.
Descripción de la prueba y valor esperado	La prueba consiste en el proceso de editar datos de un producto en el sistema de inventario. Se ingresa a la lista de los productos encontrando un icono de editar lo que le permitirá actualizar la información.
Técnica	1.Abrir e iniciar sesión al sistema de inventario web. 2. Acceder a la sección de registro de entradas 3.dar clic en lista de productos 4. Cambiar dato que se necesite y dar clic en el icono de +
Proceso de medición	-Editar datos del producto exitoso: El usuario puede completar el proceso de editar sus datos sin errores y poder guardar los datos actualizados, mantiene sus datos actualizados en el momento que lo desee. -durante el proceso de editar datos, el usuario pudo guardar exitosamente.
Criterios de terminación/valor obtenido	La prueba se considera exitosa ya que el usuario pudo actualizar y editar productos en caso de no haberlo ingresado correctamente al momento del registro.
Consideraciones especiales	-Se necesita una conexión a internet estable para realizar la prueba.
Interpretación de resultados	La prueba fue exitosa, el usuario pudo completar el proceso de poder editar producto en el sistema.
Corrección de error	No se encontraron errores durante el proceso de editar nombre sin contratiempo, no se traba la aplicación y su flujo alterno funciona correctamente.

PANTALLA DE EVIDENCIA DE EJECUCIÓN:

Se muestra a continuación las pantallas de ejecución de las pruebas en las figuras 410 411.

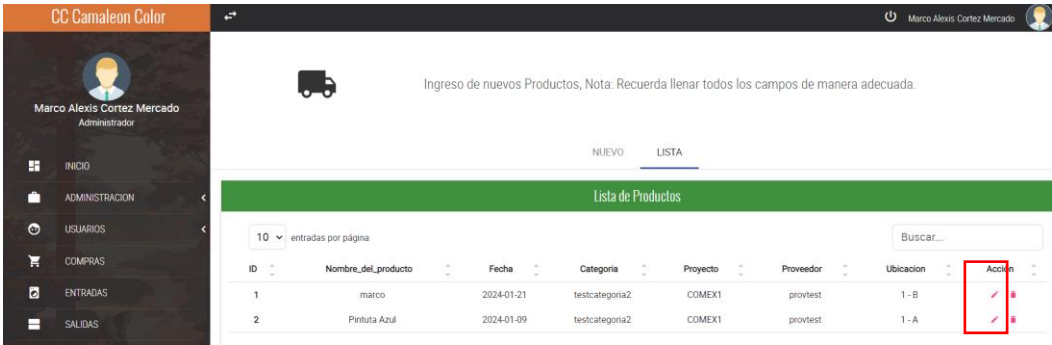


Figura. 410 Prueba de editar producto. Fuente. Elaboración propia.

Lista de Productos						
10 entradas por página		Buscar..				
ID	Nombre_del_producto	Fecha	Categoría	Proyecto	Proveedor	Ubicación
1	marco	2024-01-21	testcategoria2	COMEX2	provtest	1 - A
2	Pintura Azul	2024-01-09	testcategoria2	COMEX1	provtest	1 - A
Mostrando 1 de 2 de 2 entradas						

Figura. 411 Prueba de cambio de proyecto. Fuente. Elaboración propia.

4.3. Pruebas de seguridad

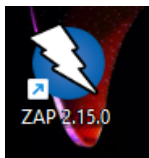
Se consideran las pruebas de seguridad en el proyecto con herramientas con el fin de mejorar e identificar como mejorar la aplicación web.

Se realizaron escaneos con OWASP con el fin de detectar vulnerabilidades y con esto abordar las amenazas si es que las tuviera, así mismo pruebas con Nmap con el objetivo de la detección de puertos y datos, escanea las direcciones IP.

4.3.1. Prueba de penetración – OWASP

OWASP o por sus siglas Open Web Application Security Project aborda la seguridad en aplicaciones web, por lo que esta herramienta puede detectar vulnerabilidades y/o amenazas a las que este expuesta la aplicación web.

Como primer paso para la exploración de la herramienta, se instala el software en su versión:



El cual ayuda a realizar un análisis de la aplicación web, esto nos da una idea al mostrar posibles fallas.

GENERAR ESCANEEO

Para escanear la aplicación web se da clic en nuevo escaneo o en su defecto escaneo automatizado ingresar el URL y comienza el escaneo una vez que se le da clic en atacar, figura 413.



Figura. 412 Interfaz de OWASP

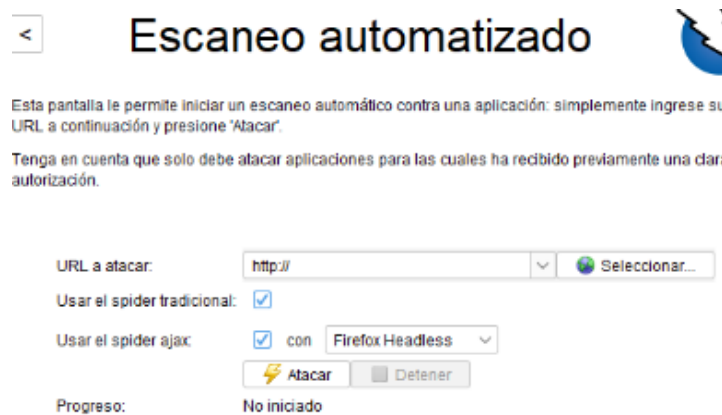


Figura. 413 Interfaz de escaneo automatizado

En el punto de inicio se colocó el URL de la aplicación web se selecciona iniciar escaneo figura 414.

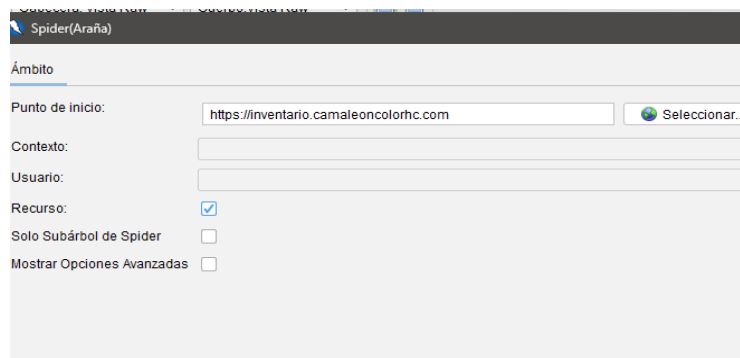
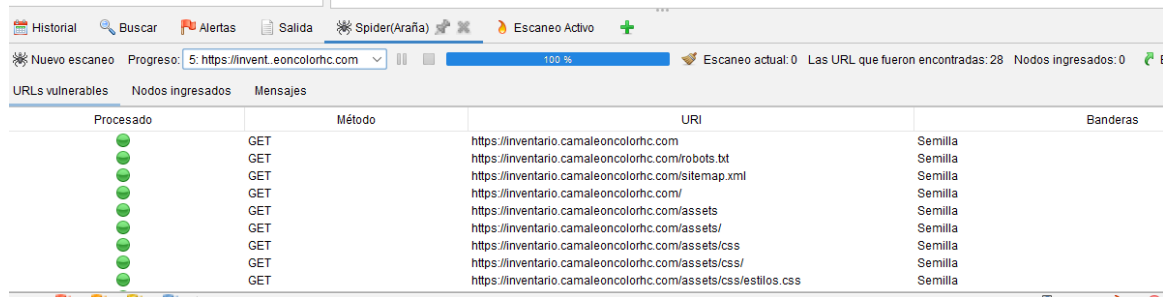


Figura. 414 URL a escanear

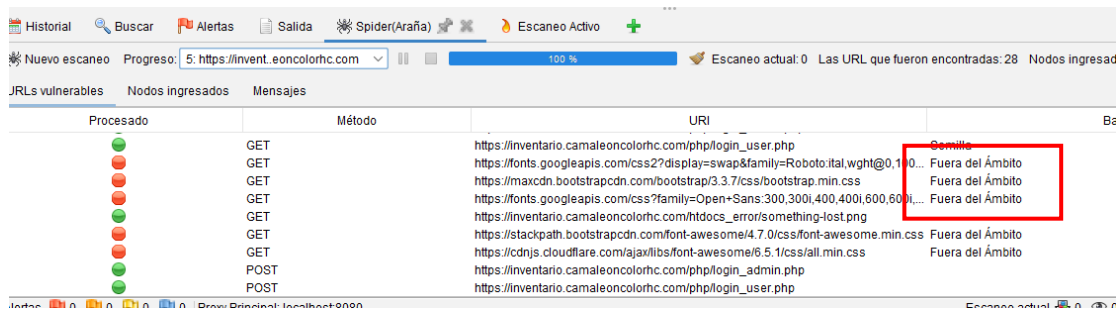
Una vez que se le da clic en **atacar** se realiza un escaneo del URL en la barra de bajo se muestran las opciones como el historial, alertas del escaneo como se muestra a continuación en la figura 415:



Procesado	Método	URI	Banderas
●	GET	https://inventario.camaleoncolorhc.com	Semilla
●	GET	https://inventario.camaleoncolorhc.com/robots.txt	Semilla
●	GET	https://inventario.camaleoncolorhc.com/sitemap.xml	Semilla
●	GET	https://inventario.camaleoncolorhc.com/	Semilla
●	GET	https://inventario.camaleoncolorhc.com/assets	Semilla
●	GET	https://inventario.camaleoncolorhc.com/assets/	Semilla
●	GET	https://inventario.camaleoncolorhc.com/assets/css	Semilla
●	GET	https://inventario.camaleoncolorhc.com/assets/css/	Semilla
●	GET	https://inventario.camaleoncolorhc.com/assets/css/estilos.css	Semilla

Figura. 415 Historial de análisis

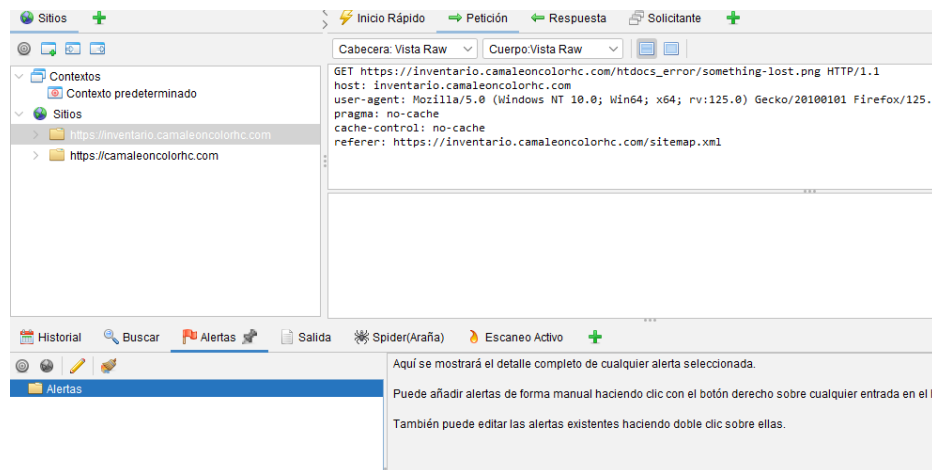
En caso de los círculos rojos como se muestra en la figura 416 el cual dice fuera del ámbito lo puede identificar como vulnerabilidad porque son servidores externos o en su caso Frameworks como lo son Bootstrap y en caso de Firebase que es externo perteneciente a Google se requiere autorización de Google por lo que la herramienta no puede realizar análisis de ese servicio externo.



Procesado	Método	URI	Banderas
●	GET	https://inventario.camaleoncolorhc.com/php/login_user.php	Semilla
●	GET	https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,100,200,300,400,500,600,700&display=swap	Fuera del Ámbito
●	GET	https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css	Fuera del Ámbito
●	GET	https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i&display=swap	Fuera del Ámbito
●	GET	https://inventario.camaleoncolorhc.com/htdocs_error/something-lost.png	Fuera del Ámbito
●	GET	https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css	Fuera del Ámbito
●	POST	https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css	Fuera del Ámbito
●	POST	https://inventario.camaleoncolorhc.com/php/login_admin.php	Semilla
●	POST	https://inventario.camaleoncolorhc.com/php/login_user.php	Semilla

Figura. 416 URL detectadas

Las alertas generadas se muestran en la pestaña de Alertas, en este caso no muestra alertas significativas.



Alertas
Aquí se mostrará el detalle completo de cualquier alerta seleccionada. Puede añadir alertas de forma manual haciendo clic con el botón derecho sobre cualquier entrada en el historial. También puede editar las alertas existentes haciendo doble clic sobre ellas.

Figura. 417 Pestaña de análisis de alertas.

4.3.2. Prueba de Nmap

Se opta por un escaneo de posibles vulnerabilidades mediante el uso de la herramienta de Nmap así como se muestra en la figura 418, Una vez culminado el escaneo de vulnerabilidades, muestra como resultado que no se hayan dentro de la página web, a su vez muestra los puertos que se encuentran habilitados y los cuales están cerrados.

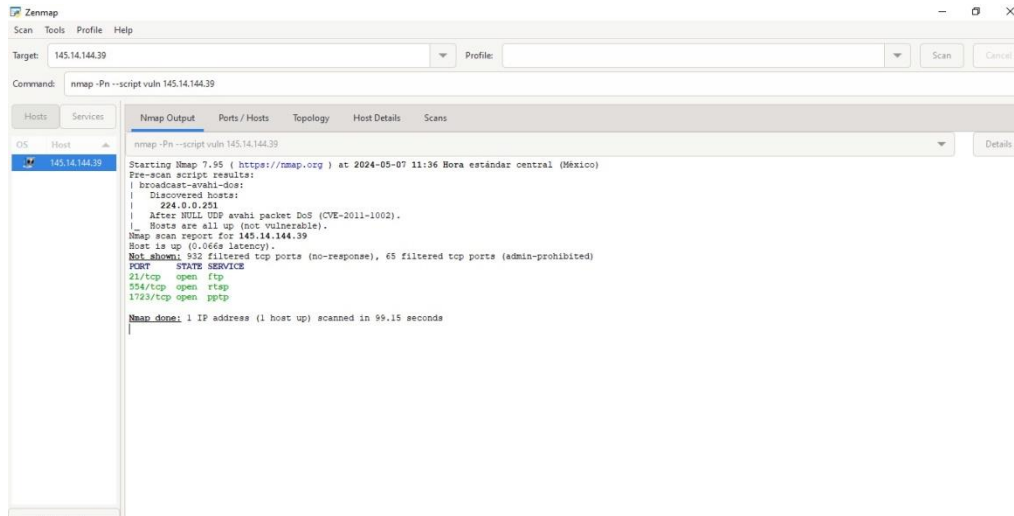


Figura. 418 Resultados de vulnerabilidades. Fuente. Elaboración propia

Por otro lado, como se observa en la figura 419, Se representa la topología del host escaneado, proporcionada por Nmap.

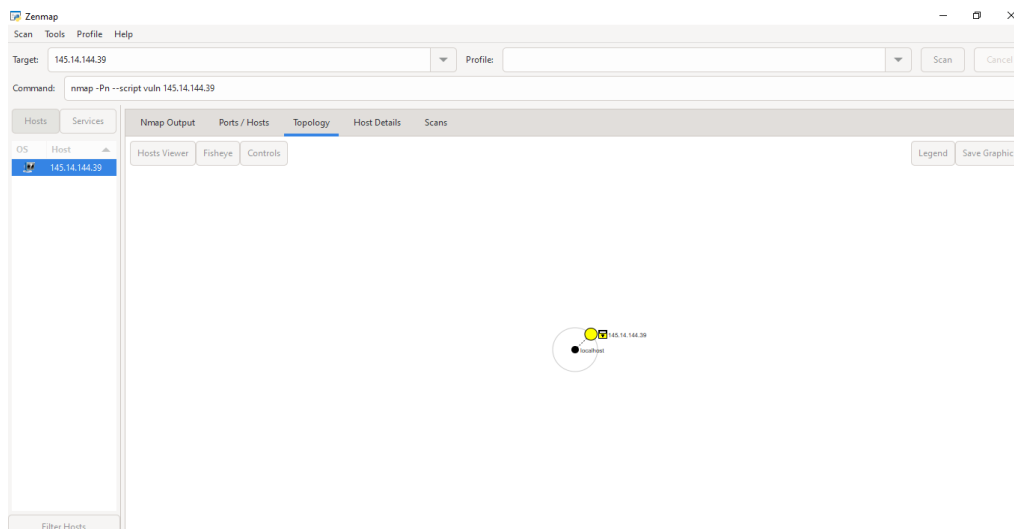


Figura. 419 Topología de host, Fuente .Elaboración propia

A su vez se muestran los detalles del host sobre los puertos y el estatus de este, como es mostrado en la figura 420.

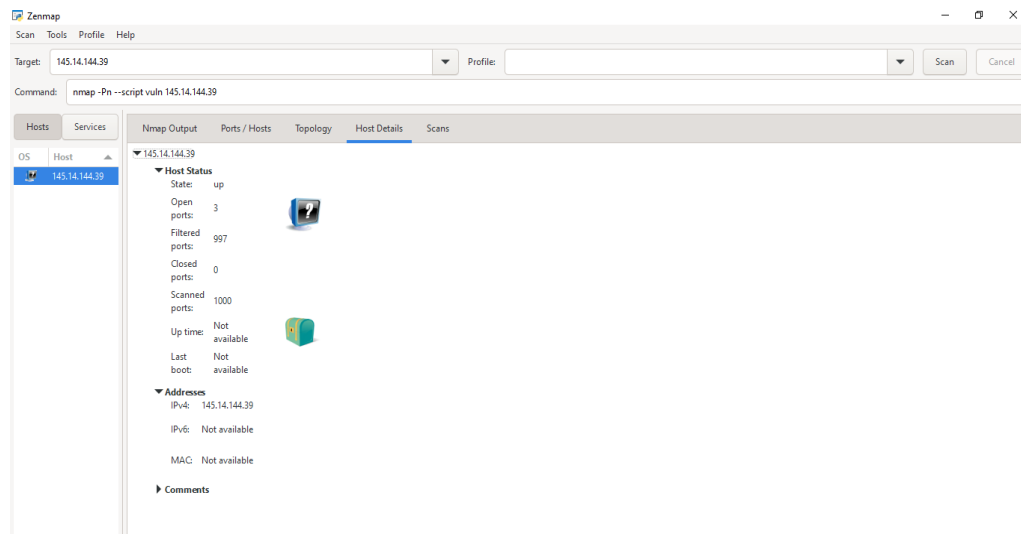


Figura. 420 Detalles del Host, Fuente. Elaboración propia

CONCLUSIONES

Con la implementación del proyecto desarrollado se creó una herramienta que aborda la problemática identificada de la empresa CC Camaleón Color. Este sistema ha proporcionado perspectivas sobre cómo mejorar los procesos y optimizar la eficiencia en la gestión de su inventario.

Durante el proceso de desarrollo de software se abordaron estrategias que pasan de teóricas a prácticas relacionado con la gestión de proyectos y mejora de procesos. Se llevo a cabo un análisis de requerimientos en colaboración con la empresa y los usuarios operativos para cubrir sus necesidades, por otro lado, se abordó el diseño del sistema con base a los requisitos que se recabaron con la empresa. Al desarrollar el sistema con herramientas actuales permite obtener ventajas en la aplicación del sistema web de inventario, tales como bases de datos en tiempo real lo que permite una sincronización instantánea de datos útil para que el inventario este actualizado en todo momento, en caso de que los datos crezcan la herramienta de Firebase permite escalabilidad para poder manejar un gran volumen de datos, trabajar con esta herramienta facilito el trabajo para el desarrollo web, por lo que se consideró una buena opción.

Se implementa el sistema realizando pruebas de funcionamiento, se trabajó en entregar un sistema para que la empresa determine su capacidad para realizar el seguimiento en tiempo real del inventario con el que cuenta y así mejoré significativamente la visibilidad y la toma de decisiones, al realizar la implementación del sistema se identificó que en cuanto al seguimiento de productos se redujeron los errores en la gestión de registros de la existencia de productos, se tomó en cuenta y se consideró proporcionar una capacitación detallada a los usuarios con una presentación digital, así como el soporte para garantizar el uso correcto del sistema.

Este proyecto ha sido una experiencia que ha ampliado la comprensión sobre la importancia de la tecnología en la industria. La colaboración con diferentes departamentos y la resolución de desafíos operativos han fortalecido en nosotros habilidades en la identificación de soluciones prácticas.

COMPETENCIAS DESARROLLADAS

Esta práctica profesional permite consolidar los conocimientos adquiridos durante la formación de la carrera en Ingeniería en Sistemas Computacionales, destacando la habilidad para analizar, diseñar e implementar estrategias destinadas a impulsar a la empresa CC Camaleón Color S.A. de C.V, hacia el logro de sus metas. En este contexto, se puso en práctica las competencias directivas cultivadas a lo largo de la carrera, integrando asignaturas claves para la obtención de conocimiento previo y así poder llevar a cabo este proyecto, incluyendo asignaturas tales como bases de datos, una materia que aborda los principios fundamentales del control de acceso, transacciones y conectividad de base de datos la cual se vincula con otras asignaturas relacionadas con el desarrollo de software las cuales son; programación web, la cual proporciona aplicaciones y herramientas para el desarrollo web considerando esta interconectividad entre otras aplicaciones e involucrar las bases de datos en la nube que fue con lo que se trabajó en este proyecto, se considera igual de importante la gestión de proyectos de software la que consiste en la búsqueda y comprensión de información aplicable a la modelación de sistemas logísticos, desarrollar la habilidad de poder organizar y planificar la información para la resolución de problemas y como abordarlos, así como la ingeniería de software la cual menciona que su finalidad es considerar metodologías para la aplicación del desarrollo de software implementando el análisis diseño, creación, gestión, desarrollo, e innovación de sistemas informáticos. Enfocada también en una orientación sistémica y sustentable, fundamentada en la toma de decisiones efectivas.

Aplicando conocimientos previos se realizó un exhaustivo análisis de la situación actual de la empresa, centrado en el proceso del inventario de productos. A partir de esta evaluación, se diseñó una estrategia orientada a asegurar que solo ciertas personas autorizadas tengan acceso a la información del inventario y manipulación de este, así mismo, se llevó a cabo un análisis e interpretación preciso de la información recabada, con el objetivo de identificar oportunidades de mejora y control. Posteriormente, se desarrolló e implemento estrategias específicas para contribuir al control de acceso de usuarios otorgando cuentas para administradores y personal de la empresa, mejorando el uso y restringiendo información y control a personales no calificados.

Para este desarrollo, se emplearon las Tecnologías de Información en la empresa. A través de su aplicación web, se optimizó el proceso de almacenamiento de productos y se facilitó el control de acceso a usuarios específicos, contribuyendo así al éxito de las estrategias implementadas.

FUENTES DE INFORMACIÓN

Alfaro López, H. G. (2002). Gordon Graham. Internet. Una indagación filosófica. Investigación Bibliotecológica: Archivonomía, bibliotecología E información, 16(32). <https://doi.org/10.22201/iibi.0187358xp.2002.32.3992>

Celaya Luna, A. (2019). CREACIÓN DE PÁGINAS WEB: HTML 5. España: ICB, S.L. (Interconsulting Bureau S.L.).

Cobo, Á. (2005). PHP y MySQL. Tecnología para el desarrollo de aplicaciones web. Ediciones Díaz de Santos. <https://dialnet.unirioja.es/servlet/libro?codigo=309227>.

Colobran Huguet, M., Arqués Soldevila, J. M., Marco Galindo, E. (2008). Administración de sistemas operativos en red. España: Editorial UOC

Creación de un sitio web con PHP y MySQL. 5ª Edición actualizada. (n.d.). (n.p.): Grupo Editorial RA-MA.

Desarrollo de Software Ágil: Extreme Programming y Scrum. (2014). CreateSpace Independent Publishing Platform.

Debrauwer, L., Van der Heyde, F. (2016). UML 2.5: iniciación, ejemplos y ejercicios corregidos. España: Ediciones ENI.

Dimes, T. (2015). JavaScript Una Guía de Aprendizaje para el Lenguaje de Programación JavaScript. Estados Unidos: Babelcube Incorporated.

Firebase. (s. f.). Firebase. <https://firebase.google.com/?hl=es>

Fontela, C. (2012). UML: Modelado de Software para Profesionales. Argentina: México.

García Rubio, F. O., Piattini Velthuis, M. (2014). Desarrollo Global de Software. España: RA-MA Editorial.

Heurtel, O. (2016). PHP 7: Desarrollar un sitio web dinámico e interactivo. España: ENI.

Alfaro López, H. G. (2002). Gordon Graham. Internet. Una indagación filosófica. Investigación Bibliotecológica: Archivonomía, bibliotecología E información, 16(32).
<https://doi.org/10.22201/iibi.0187358xp.2002.32.3992>

Powell, Thomas A. (2001). Diseño de Sitios Web: manual de referencia: Osborne McGraw-Hill.

Pulido Romero, E., Escobar Domínguez, Ó., Núñez Pérez, J. Á. (2019). Base de datos. México: Patria Educación.

Ramos Cardozzo, D. (2016). Desarrollo de Software: Requisitos, Estimaciones y Análisis. 2 edición. España: CreateSpace Independent Publishing Platform.

Rubio Jiménez, M. Á. (2022). Historia de los Lenguajes de Programación: Años 1940-1959. España: Altenwald Books.

Sommerville, I. (2005). Ingeniería del software. España: Pearson Educación.

Sublime Text. (s.f.). Sublime Text - Download. Retrieved from
<https://www.sublimetext.com/3/>.

Taha, H. A. (1995). Investigación de operaciones. México: Alfa Omega.

Tomás, J., Carbonell, V., Bataller, J., Lloret, J. (2019). Firebase: trabajar en la nube. Colombia: Alpha Editorial.

XAMPP Installers and Downloads for Apache Friends. (s. f.).
<https://www.apachefriends.org/es/index.html>

ANEXOS

MANUAL DE USUARIO

Introducción

El presente manual de usuario consiste en poder dar información sobre la funcionalidad del sistema, con el fin de que el usuario comprenda la funcionalidad del sistema de información web para la gestión y control de inventario de pinturas.

Dentro de los puntos que se abarcaran en el presente documento son:

- Alcances, donde se da a conocer de que trata el sistema y por qué es fundamental de acuerdo con este manual de usuario.
- Guía de uso, se especifica paso a paso las pantallas de cada función del sistema.
- Selección de solución de problemas, se muestran los problemas más comunes que surgen, así como su solución.
- Contacto de soporte técnico, se da a conocer algún correo o número de teléfono para brindar asistencia,

Alcances

El sistema de información web para la gestión y control de inventario de pinturas consiste en ofrecer una amplia selección de ingreso de información y administración de inventario para que los usuarios puedan encontrar el artículo requerido para su ingreso, compra o edición de datos para cada artículo, así como la administración de usuarios para el uso del sistema, las funciones con las que contara el sistema son las siguientes:

- Alerta de baja reservas en inventario: Esta función permite la posibilidad de visualizar la alerta de baja cantidad de un producto, mediante un color resaltado sobre la cantidad baja los cuales son: Amarillo para cantidades que van desde 500 a 201 y rojo para cantidades de 200 a 0.
- Cierre de sesión por inactividad del sistema: Esta función permite cerrar sesión de forma automática en caso de no interactuar con el sistema por más de 5 minutos.
- Personalización de usuario: Esta función permitiría a los usuarios personalizar el diseño del entorno de su sesión, mediante cambios como lo son: ingreso y cambio de foto de perfil de usuario, cambio de color y fondo del sistema
- Informes y Análisis: El sistema cuenta con una funcionalidad para generar las tablas de

los registros en un formato tipo Excel, con estas funcionalidades se podría generar informes personalizables sobre el estado del inventario, así como también análisis de tendencias y comportamientos para una mejor toma de decisión.

- Resultado automatizado en el proceso de compra: Esta función permite dar de forma automática el resultado total de la compra, mediante el ingreso de las cantidades adquiridas y el precio unitario de la unidad, el IVA ya está implementado dentro del resultado final.

Hasta donde se quiere llegar, pues la función principal de nuestro manual de usuario es llegar a proporcionar a los usuarios toda la información necesaria para utilizar nuestro sistema de manera efectiva y eficiente pues en él se podrán encontrar descripción general de nuestro sistema, instrucciones paso a paso sobre el proceso de instalación, instrucciones paso a paso de inicio de sesión y creación de cuentas, instrucciones de uso, solución de problemas comunes que nuestros usuarios puedan encontrar durante el uso del sistema, así como un apartado de donde se proporcionaran pautas sobre cómo mantener y cuidar el sistema para asegurar su buen funcionamiento y prolongar su vida útil.

Guía de uso

El manual de usuario tiene como finalidad facilitar al usuario el uso del sistema, que los usuarios que tengan acceso al sistema puedan consultar el manual si es que llegaran a presentar alguna duda figura 421.

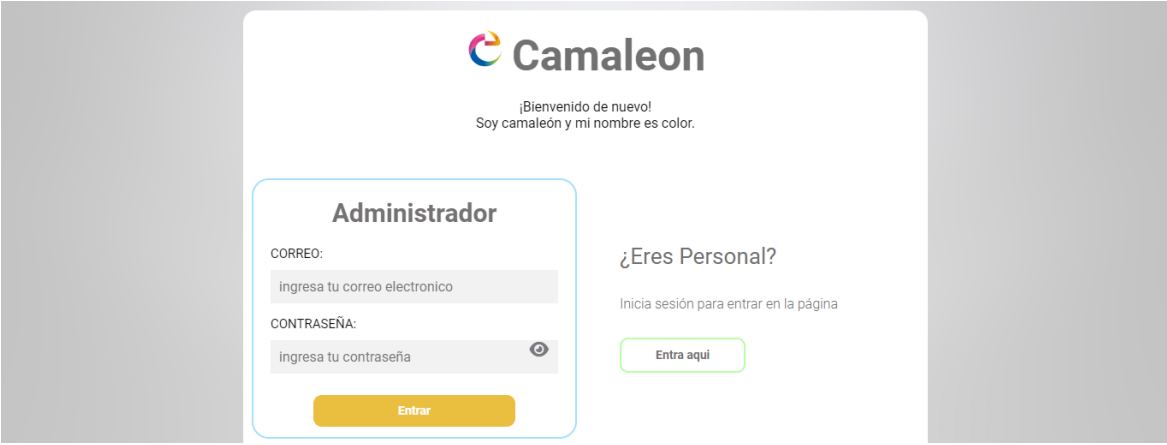
The image shows a login interface for a system named 'Camaleon'. At the top center is the 'Camaleon' logo, which consists of a stylized 'C' made of four colored segments (red, yellow, green, blue) followed by the word 'Camaleon' in a bold, sans-serif font. Below the logo is a welcome message: '¡Bienvenido de nuevo! Soy camaleón y mi nombre es color.' The interface is divided into two main sections. On the left, under the heading 'Administrador', there are two input fields: 'CORREO:' with a placeholder 'Ingresa tu correo electronico' and 'CONTRASEÑA:' with a placeholder 'Ingresa tu contraseña' and an eye icon to toggle visibility. Below these fields is a yellow 'Entrar' button. On the right, under the heading '¿Eres Personal?', there is a green 'Entra aquí' button and a line of text 'Inicia sesión para entrar en la página'.

Figura. 421 Inicio de sesión- administrador, Fuente. Elaboración propia

Se muestra la pantalla de inicio de sesión para los administradores, donde se podrá autenticar mediante el correo electrónico y contraseña, posteriormente se tiene la opción para observar la contraseña y esta no se muestre de forma privada evitando errores de ingreso.

Se cuenta con la implementación de iniciar sesión como personal mediante la interacción de un botón con el texto de (Entra aquí) botón que se observa en la figura 422.

The image shows a login interface for a system named 'Camaleon', specifically for regular users. The interface is divided into two main sections. On the left, under the heading '¿Eres Administrador?', there is a green 'Entra aquí' button and a line of text 'Inicia sesión para entrar en la página'. On the right, under the heading 'Personal', there are two input fields: 'USUARIO:' with a placeholder 'Ingresa tu usuario' and 'CONTRASEÑA:' with a placeholder 'Ingresa tu contraseña' and an eye icon to toggle visibility. Below these fields is a yellow 'Entrar' button.

Figura. 422 Inicio de sesión- usuario operativo, Fuente. Elaboración propia

Se muestra la figura 423 el inicio de sesión para el personal, donde se podrá autenticar mediante el usuario y contraseña, posteriormente se tiene la opción para observar la contraseña y esta no se muestre de forma privada evitando errores de ingreso.

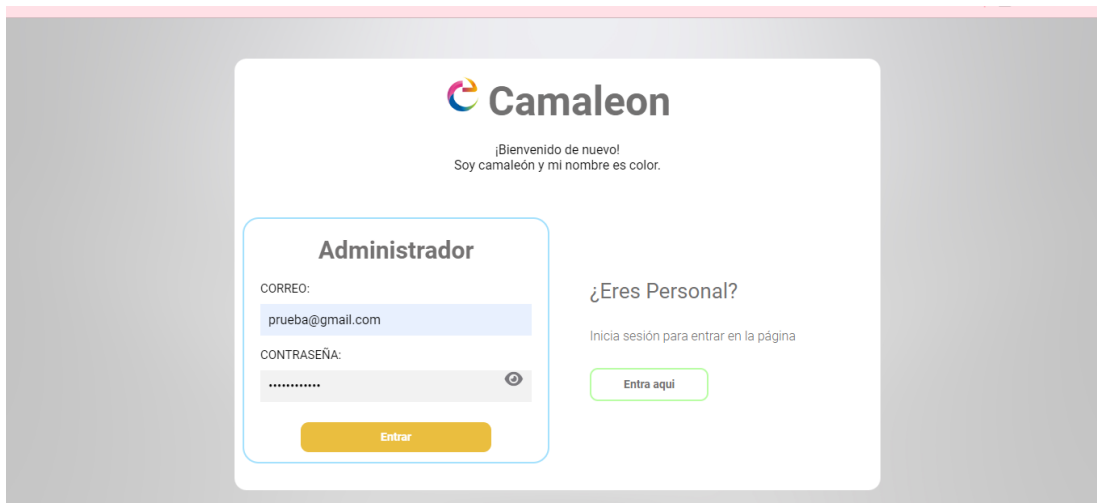


Figura. 423 Inicio de sesión, Fuente. Elaboración propia

En caso de haber ingresado sin coloca la contraseña correspondiente el sistema mandará una alerta explicando el motivo del fallo del ingreso con el mensaje de “Ingrese la contraseña”, como se observa en la figura 424 donde se deberá de aceptar la advertencia y regresara a la pantalla de inicio de sesión

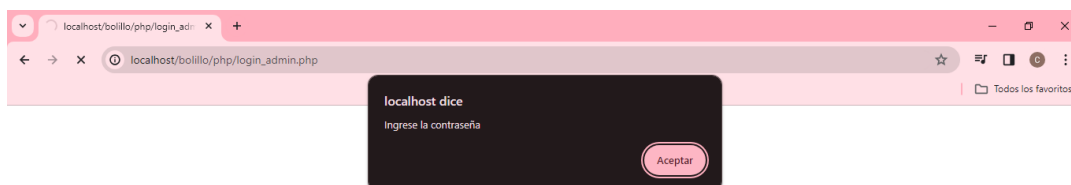


Figura. 424 Alerta de fallo de inicio se sesión, Fuente. Elaboración propia

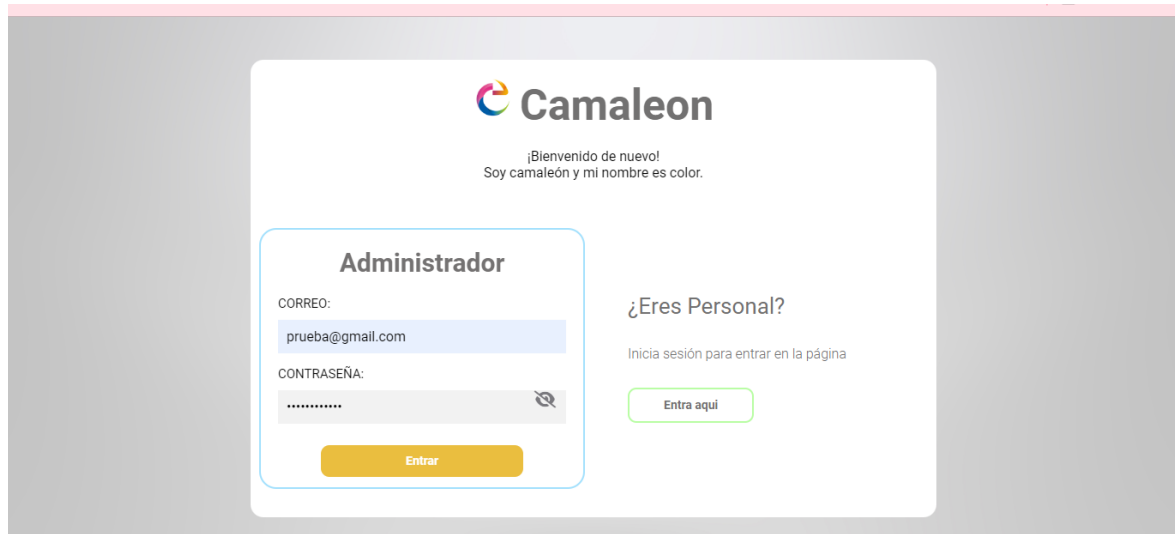


Figura. 425 Inicio de sesión v3, Fuente. Elaboración propia

En caso de haber puesto incorrectamente la contraseña ya sea por una cantidad menor de dígitos o símbolos no registrados se mostrará en pantalla un mensaje de advertencia “Contraseña Incorrecta” en figura 426 por lo que se deberá de aceptar la advertencia y regresara a la pantalla de inicio de sesión para el ingreso de la contraseña correctamente.

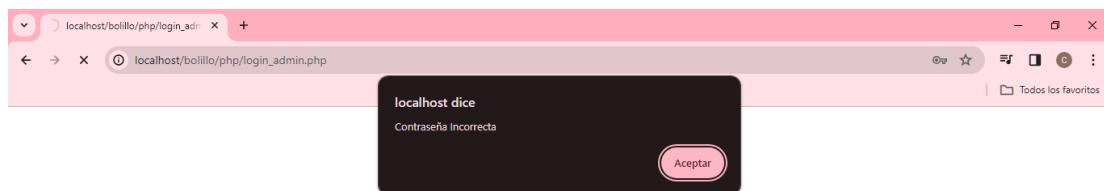


Figura. 426 Alerta contraseña incorrecta, Fuente. Elaboración propia

Vuelve a la página de inicio como se muestra en la figura 427.

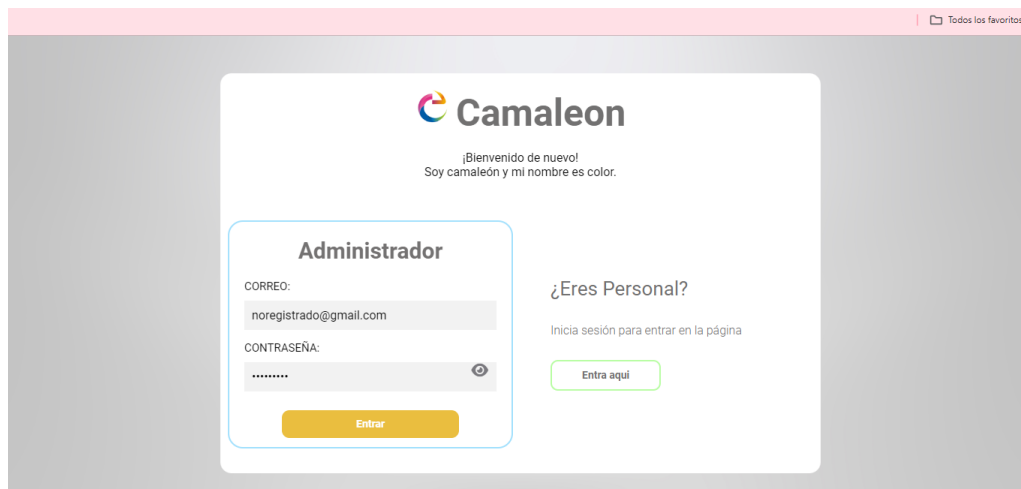


Figura. 427 Inicio de sesión v4, Fuente. Elaboración propia

Si la contraseña no fue el único motivo de advertencia se mostrará un segundo mensaje de advertencia especificando que el correo no está registrado (Figura 428 y 429) o que este deberá de ser ingresado de manera obligatoria antes de continuar, por lo que se aceptará el mensaje y regresará al inicio de sesión para el ingreso correcto del correo.

NOTA: En caso del inicio de sesión del personal son los mismos protocolos de ingreso especificando el ingreso de usuario y contraseña de manera correcta



Figura. 428 Alerta de correo. Fuente. Elaboración propia



Figura. 429 Alerta de correo incorrecto. Fuente. Elaboración propia

Una vez ingresado los datos de manera correcta, el sistema mostrará un mensaje de advertencia “Autenticación Exitosa” una vez visualizado el mensaje deberá de ser aceptado, dando paso al menú de inicio del sistema. Figura 430

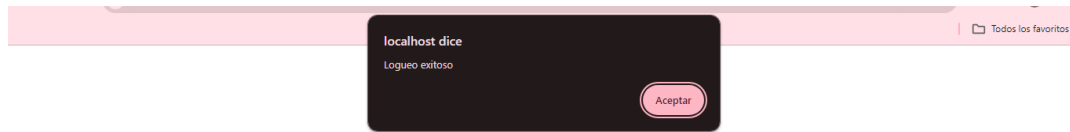


Figura. 430 Inicio de sesión v5, Fuente Elaboración propia

Una vez dentro del sistema, este mostrara un menú interactivo de opciones del lado izquierdo en la figura 431 432 y 433presentando el nombre del usuario que inicio sesión y su usuario respectivo, se muestran 7 botones de selección para el uso del sistema los cuales son:

- Inicio: Regresa al menú del sistema.
- Administración: Botón desplegable que muestra las opciones



Figura. 431 Pantalla de inicio, Fuente. Elaboración propia



Figura. 432 Pantalla de menú, Fuente. Elaboración propia



Figura. 433 Cinta de opciones del menú, Fuente. Elaboración propia

1. Proveedores: Opción que redirige al formulario de proveedores.
2. Ubicaciones: Opción que redirige al formulario de ubicaciones.
3. Categorías: Opción que redirige al formulario de categorías.
4. Proyectos: Opción que redirige al formulario de proyectos.
5. Usuarios: Botón desplegable que muestra las opciones.
 1. Administradores: Opción que redirige al formulario de administradores.
 2. Personal calificado: Opción que redirige al formulario del personal.
 - Compras: Botón que redirige al formulario de compras.
 - Entradas: Botón que redirige al formulario de productos.
 - Salidas: Botón que redirige a la tabla de registro de salidas.

- Inventario: Botón que redirige a la tabla de inventario.

menú de opciones mostrada en la figura 434.



Figura. 434 Cinta de opciones del menú 2, Fuente. Elaboración propia

Por otra parte, en las opciones del lado derecho se muestran los azulejos de información de registros ejemplo en la figura 435 que visualizan el número de inserciones u salidas registradas, en la parte superior derecha se muestra el apartado de cerrar sesión y el nombre completo del usuario que haya iniciado sesión.

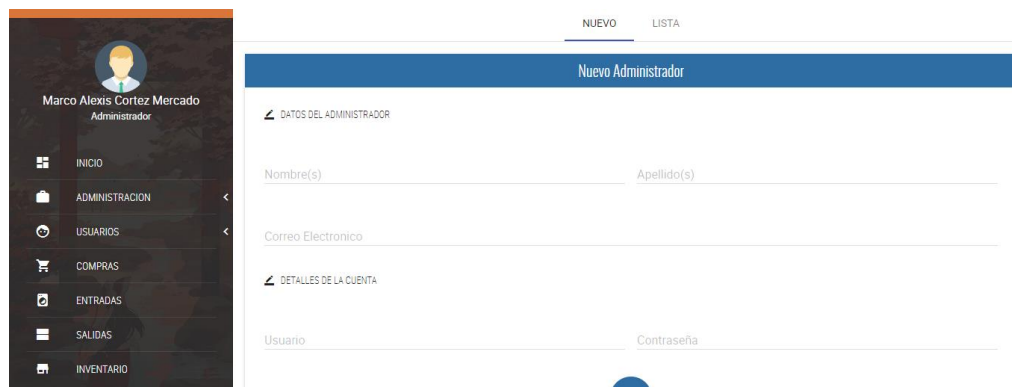


Figura. 435 Registro de usuarios (Administrador), Fuente. Elaboración propia

Se muestra la pantalla de registro de usuarios en donde se ingresarán los datos personales: Nombre(s), Apellido(s), Correo electrónico, Usuario, Contraseña, en el apartado de los campos anteriores al igual que en el inicio de sesión deberán de llenarse de manera correcta y con los datos (figura 436)

Correctos ya que al no cumplir con estas especificaciones se mostrará una advertencia ya sea de “Error de llenado” o “Dato en uso) una vez colocado los datos, debajo se encuentra el botón de registrar el cual almacenará los datos ingresados.

Figura. 436 Registro de usuarios v2, Fuente. Elaboración propia

Se muestra la pantalla de registro en caso de que los datos de llenado no coincidan con el formato correcto, se deberá de evitar ingresar símbolos y números en los campos de Nombre(s) Apellido(s) y Usuario ejemplo en la figura 437

Correo Electronico
aracelimercado01gmail.com
Correo Invalido

Figura. 437 Ingreso de correo, Fuente. Elaboración propia

En caso de que el formato de correo no coincida, se deberá de ingresar nuevamente y verificar que se coloque la terminación: @gmail.com, @hotmail.com, etc.

Una vez ingresado los datos correctamente en el apartado de lista se muestra el registro exitoso del usuario dentro de una tabla, donde se puede ordenar de mayor a menor y buscar un registro específico mediante la barra de búsqueda. A su vez se muestran los botones de acción los cuales permiten la eliminación y edición de los registros en figura 438.

The screenshot shows a web application interface. On the left is a sidebar menu for 'CC Camaleon Color' with a user profile 'Marco Alexis Cortez Mercado Administrador'. The main area has a header 'Ingreso de nuevos administradores, Nota: Recuerda llenar todos los campos de manera adecuada.' and tabs for 'NUEVO' and 'LISTA'. The 'LISTA' tab is active, displaying a table titled 'Lista de Administradores'. The table has columns: Id, Nombre(s), Apellido(s), Usuario, Correo, Contraseña, and Acción. It contains 3 rows of data. Below the table, it says 'Mostrando 1 de 3 de 3 entradas' and has a button 'Exportar datos a Excel'.

Id	Nombre(s)	Apellido(s)	Usuario	Correo	Contraseña	Acción
1	marcos	cortez mercado	sapo	bolillo@gmail.com	12345678	[Edit] [Delete]
2	Marco Alexis	Cortez Mercado	Administrador	prueba@gmail.com	Bolillo34	[Edit] [Delete]
3	Diana Ivonne	Hernández de la Cruz	DianaAdmin	Diana@gmail.com	12345678	[Edit] [Delete]

Figura. 438 Lista de usuarios registrados, Fuente. Elaboración propia

Para eliminar a un usuario es necesario dirigirse al apartado de lista y colocarse en el icono de eliminación y un clic mostrado en la figura 439

This is a close-up of the 'Lista de Administradores' table. It shows the first two rows of data. The 'Acción' column contains two icons: a pencil for editing and a trash can for deleting. The first row corresponds to 'Marco Alexis Cortez Mercado' and the second to 'Diana Ivonne Hernández de la Cruz'.

Id	Nombre(s)	Apellido(s)	Usuario	Correo	Contraseña	Acción
1	Marco Alexis	Cortez Mercado	Administrador	prueba@gmail.com	Bolillo34	[Edit] [Delete]
2	Diana Ivonne	Hernández de la Cruz	DianaAdmin	Diana@gmail.com	12345678	[Edit] [Delete]

Figura. 439 Eliminar usuario, Fuente. Elaboración propia

Una vez completado el proceso de eliminación saldrá la advertencia de “Datos eliminados” la cual una vez aceptada regresará al formulario de registro como se observa en la figura 440.

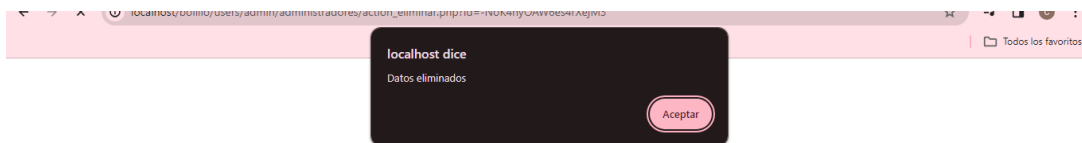


Figura. 440 Alerta de datos eliminados, Fuente. Elaboración propia.

Para editar la información de un usuario es necesario dirigirse al apartado de lista y colocarse en el icono de edición y un clic marcado en la figura 441.

10

entradas por página

Buscar...

Id	Nombre(s)	Apellido(s)	Usuario	Correo	Contraseña	Acción
1	Marco Alexis	Cortez Mercado	Administrador	prueba@gmail.com	Bolillo34	
2	Diana Ivonne	Hernández de la Cruz	DianaAdmin	Diana@gmail.com	12345678	

Mostrando 1 de 2 de 2 entradas

Exportar datos a Excel

Figura. 441 Editar usuario, Fuente. Elaboración propia

Se presenta la pantalla de edición de usuarios en donde los datos colocados previamente son mostrados en los campos correspondientes para su edición, como anteriormente se mencionó los campos deberán de llenarse de manera indicada con el formato indicado, ya que de lo contrario se mostrarán mensajes de advertencias de llenado de campos y error en el formato de los campos en la figura 442.

EDITAR

LISTA

Edición de Administrador

Nombre(s)

Diana Ivonne

Apellido(s)

Hernández de la Cruz

Correo Electrónico

Diana@gmail.com

Usuario

DianaAdmin

Contraseña

+

Figura. 442 Editar usuario formulario, Fuente. Elaboración propia

Una vez completado los cambios de edición se redirigirá al formulario de registro de usuarios y se verán reflejados los cambios en la tabla de registros mostrado en la figura 443.

Lista de Administradores						
10 entradas por página		Buscar...				
Id	Nombre(s)	Apellido(s)	Usuario	Correo	Contraseña	Acción
1	Marco Alexis	Cortez Mercado	Administrador	prueba@gmail.com	Bolillo34	 
2	Diana Ivonne	Hernández de la Cruz	DianaAdmin	Diana@gmail.com	123456789	
Mostrando 1 de 2 de 2 entradas						
Exportar datos a Excel						

Figura. 443 Lista de usuarios administradores, Fuente. Elaboración propia

Se muestra la figura 444 de registro de proveedores en donde se ingresarán los datos personales del proveedor: Nombre del proveedor, Selección de país (Lista de selección), Código postal, Ciudad, Estado, Calle/Avenida, Colonia, Teléfono y Correo electrónico, en el apartado de los campos anteriores al igual que en el inicio de sesión deberán de llenarse de manera correcta y con los datos. Correctos ya que al no cumplir con estas especificaciones se mostrará una advertencia ya sea de “Error de llenado” o “Dato en uso). Una vez colocado los datos de manera correcta en la parte inferior se encuentra el botón de registrar el cual almacenara los datos ingresados.

CC Camaleon Color



Marco Alexis Cortez Mercado
Administrador

- INICIO
- ADMINISTRACION
- USUARIOS
- COMPRAS
- ENTRADAS
- SALIDAS
- INVENTARIO

Nuevo Proveedor

← DATOS DEL PROVEEDOR

Nombre del proveedor

← DIRECCION

Selecciona el Pais

Codigo Postal Ciudad

Estado Calle/Avenida Num Int. Num Ext.

Colonia

Figura. 444 Registro de proveedores, Fuente. Elaboración propia

Se muestra la figura 445 de registro en caso de que los datos de llenado no coincidan con el formato correcto, se deberá de evitar ingresar símbolos y números en los campos de Nombre del proveedor, Código postal, Ciudad, Estado, Calle/Avenida, Colonia.

The screenshot shows a web application interface for 'CC Camaleón Color'. On the left is a sidebar menu with options: INICIO, ADMINISTRACION, USUARIOS, COMPRAS, ENTRADAS, SALIDAS, and INVENTARIO. The main area is titled 'Nuevo Proveedor' and contains a form with the following fields and errors:

- DATOS DEL PROVEEDOR:**
 - Nombre del proveedor: Comex (Error: Nombre invalido)
- DIRECCION:**
 - Selecciona el Pais: (Dropdown menu)
 - Código Postal: 55400- (Error: Nombre invalido)
 - Ciudad: mexico= (Error: Ciudad Invalido)
 - Estado: Indica la provincia, (Error: Estado Invalido)
 - Calle/Avenida Num Int. Num Ext.: Cerrada de Fresno Mt25 L5 (Error: Datos Invalidos)
 - Colonia: Ejidos de santa maria Tulpetlac- (Error: Datos Invalidos)

Figura. 445 Sección nuevo proveedor, Fuente. Elaboración propia

En la parte de selección de país se deberá de elegir el país de origen del proveedor, ejemplo en la figura 446.

The screenshot shows the 'DIRECCION' section of the form. It includes a dropdown menu for 'Selecciona el Pais' with 'Canadá' selected.

Figura. 446 Dirección, Fuente. Elaboración propia

En caso de que el formato de correo no coincida, se deberá de ingresar nuevamente y verificar que se coloque la terminación: @gmail.com, @hotmail.com, etc. Figura 447.

The screenshot shows the 'CONTACTOS' section of the form. It includes two input fields:

- Telefono:** 5621503696
- Correo electronico:** aracelimercado01@gmail.com (Error: Correo Invalido)

Figura. 447 Contactos, Fuente Elaboración propia

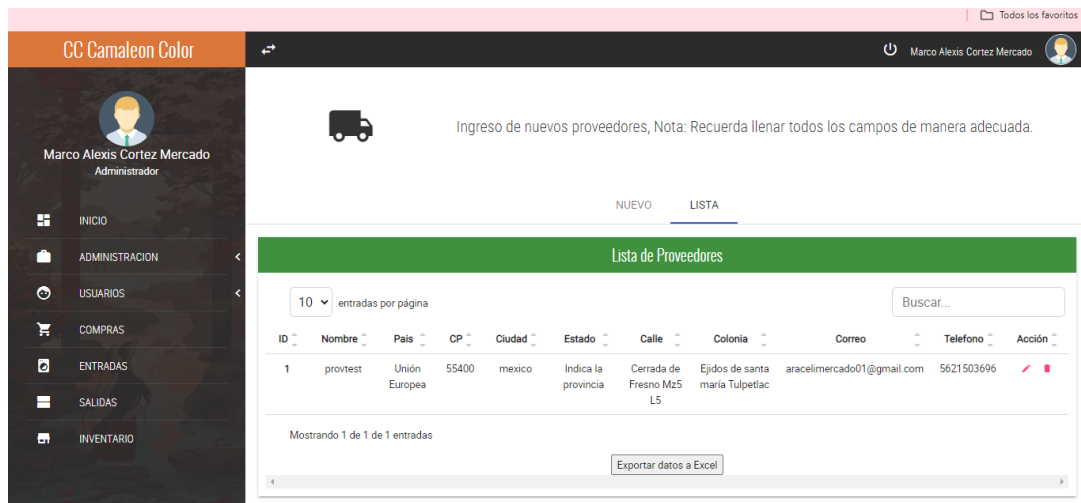


Figura. 448 Lista de proveedores, Fuente. Elaboración propia

Una vez ingresado los datos correctamente en el apartado de lista se muestra el registro exitoso del proveedor dentro de una tabla, donde se puede ordenar de mayor a menor y buscar un registro específico mediante la barra de búsqueda. A su vez se muestran los botones de acción los cuales permiten la eliminación y edición de los registros figura 442.

Para eliminar a un usuario es necesario dirigirse al apartado de lista y colocarse en el icono de eliminación y un clic. Marcado en la figura 449.



Figura. 449 Eliminar registro, Fuente. Elaboración propia

Una vez completado el proceso de eliminación saldrá la advertencia de “Datos eliminados” la cual una vez aceptada regresará al formulario de registro muestra en figura 450.



Figura. 450 Alerta de datos eliminados

Para la descarga de un reporte se dará clic en exportar datos a Excel como se muestra en la figura 451.

Lista de compras								
10 entradas por página		Buscar...						
ID	Usuario	Producto	Unidad_de_Medida	Cantidad	Precio Unitario	Fecha de Compra	Proveedor	Total de Compra
1	Marco Alexis	marco	MiniLitros	10000	22	2024-01-21	provtest	51.04
2	Marco Alexis	marco	Gramos	3	2	2024-01-24	provtest	27.84
Mostrando 1 de 2 de 2 entradas								
Exportar datos a Excel								

Figura. 451 Descarga y genera reporte

Se muestra la pantalla de descarga de Excel, para este proceso es necesario situarse en la tabla de registros deseada, en la parte inferior de la tabla se encuentra un botón en el cual deberá de darse un clic para que empiece el proceso de descarga la descarga se verá en el navegador como se muestra en la figura 452.

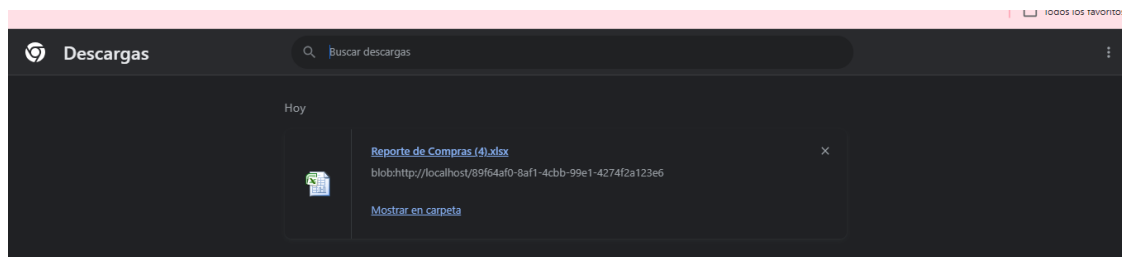


Figura. 452 Reporte descargado

Se muestra la figura 453 muestra la finalización de sesión del usuario, para lo cual es necesario situarse en el icono de salida el cual está ubicado en la parte superior derecha del sistema, será necesario dar un clic para empezar el proceso.



Figura. 453 Cerrar sesión

Se mostrará un mensaje de advertencia con dos opciones: Salir o Cancelar, una vez elegida la acción salir se redirigirá a la página de autenticación de usuarios finalizando el proceso de cierre de sesión. Figura 454

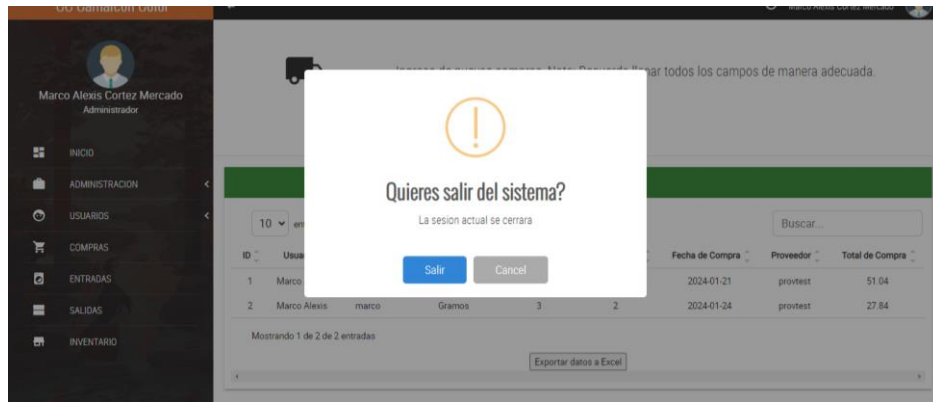


Figura. 454 Confirmación de cerrar sesión

Sección de solución de problemas.

1. No puedo iniciar sesión:

- Verifique que está utilizando la dirección de correo electrónico y la contraseña correcta.
- Asegúrese de que su cuenta haya sido registrada correctamente por un administrador.
- Si olvidó su contraseña: Informe a un administrador para verificarla o solicitar un cambio.
- Compruebe su conexión a Internet, ya que la falta de conexión puede impedir el inicio de sesión.

2. No puedo registrar usuarios:

- Verifique si los campos se llenaron de manera correcta.
- Asegúrese de tener conexión a internet, ya que la falta de conexión puede impedir la finalización del registro.
- Espere unos minutos, ya que a veces puede haber retrasos en el registro del usuario.

3. No puedo agregar productos:

- Asegúrese de que los campos se llenaron de manera correcta.
- Verifique si el producto a agregar no esté registrado previamente.
- Asegúrese de tener conexión a internet, ya que la falta de conexión puede impedir la finalización del registro.

4. No agregar compras:

- Asegúrese de que los campos se llenaron de manera correcta.
- Verifique si el Id de la compra no esté registrado previamente.
- Asegúrese de tener conexión a internet, ya que la falta de conexión puede impedir la finalización del registro.

5. No puedo editar registros:

- Asegúrese de estar utilizando la función de edición proporcionada en el sistema.
- Verifique que ha ingresado los nuevos datos correctamente y que cumplan con los requisitos establecidos (longitud, formato, etc.).

Si experimenta algún otro problema que no se mencione aquí, le recomendamos ponerse en contacto con el soporte técnico del sistema.

EVIDENCIAS DE IMPLEMENTACIÓN

Durante el desarrollo del proyecto de sistema de inventario para la empresa CC Camaleón Color, se llevó a cabo una exposición demostrativa el día 11 de enero, tal evento fue diseñado para presentar los resultados del proyecto dirigido a la dirección y el departamento involucrado que es almacén y calidad.

La exposición contó con una variedad de elementos visuales, tales como diapositivas, demostración en vivo con ejemplos reales en el sistema web destacando las características y funciones, se tomaron fotografías que capturaron la presentación.

La imagen 455 muestra la presentación de las diferentes funcionalidades del sistema de inventario.

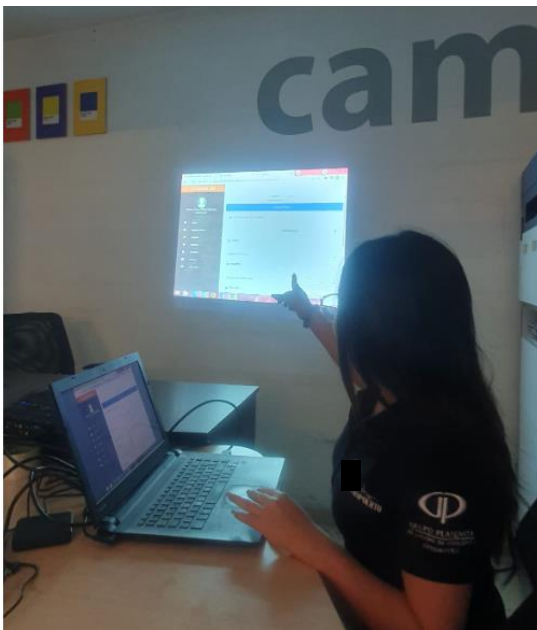


Figura. 455 Presentación img1

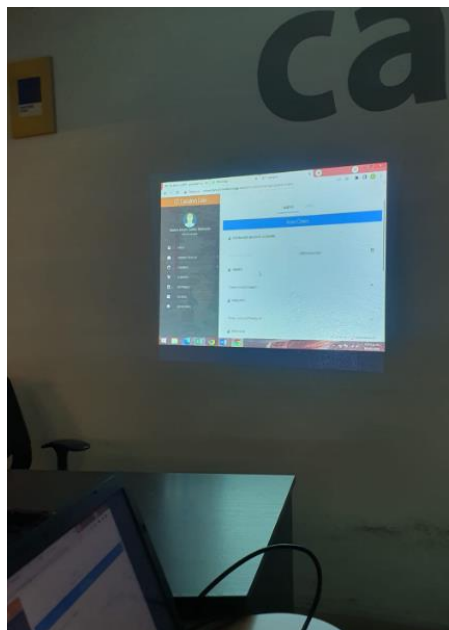


Figura. 456. Presentación img2

En los anaqueles se fueron acomodando y clasificando, según el material y proveedor, se registran en el sistema para identificarlos por ubicación de anaquel. Ejemplo de anaqueles en figuras 457, 458 453 y 459provisionalmente se identifican con carteles con letras como se muestra en la figura 461 posteriormente se acordó hacer los identificadores con vinil para mejor resistencia



Figura. 457 Anaqueles1



Figura. 458 Anaqueles2



Figura. 459 Maderas



Figura. 460 Anaqueles 3



Figura. 461 Anaqueles4

Se acondiciona un espacio para que el usuario operativo pueda hacer uso del sistema web, gracias a que la base de datos es en la nube se puede usar en cualquier dispositivo por lo que solo necesitaran su computadora espacio mostrado en figuras 462 y 463.

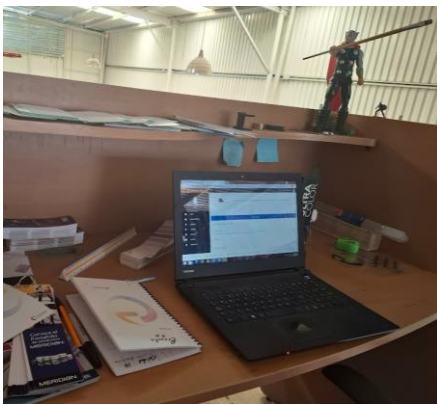


Figura. 462 Espacio de trabajo

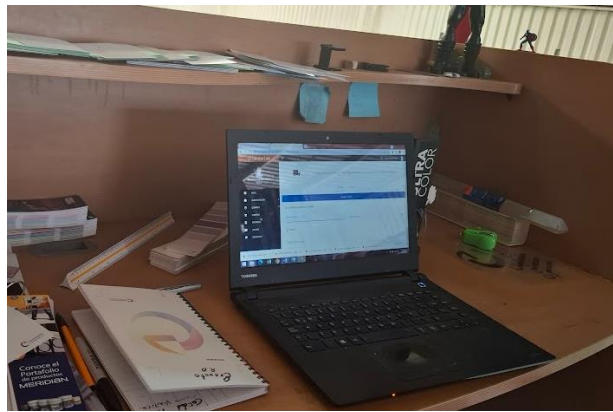


Figura. 463 Espacio de trabajo

El sistema se presta para usarse en cualquier espacio, de esta manera poder moverse y capturar datos de los productos en el momento que se necesite.

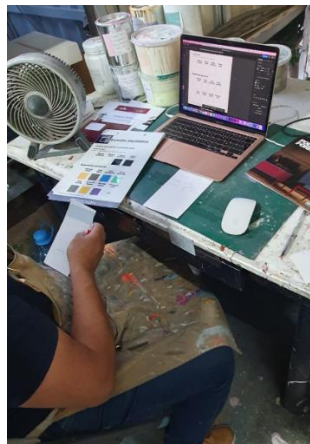


Figura. 464 Espacio de trabajo