INSTITUTO TECNOLÓGICO SUPERIOR DE MISANTLA

SISTEMA PLANIFICADOR DE LOS RECURSOS DE LA EMPRESA GRUPO MULTICLIMAS

TESIS

QUE PARA OBTENER EL TITULO DE

INGENIERO EN SISTEMAS COMPUTACIONALES

P R E S E N T A

JOSE LUIS MELGAREJO ORTIZ

DIRECTOR:

MSC. JOSÉ ANTONIO HIRAM VÁZQUEZ LÓPEZ

CODIRECTORES:

DR. SIMÓN PEDRO ARGUIJO HERNÁNDEZ

MIA. ROBERTO ÁNGEL MELÉNDEZ ARMENTA

MISANTLA, VERACRUZ Enero 2019

INSTITUTO TECNOLÓGICO SUPERIOR DE MISANTLA
DIVISIÓN DE ESTUDIOS PROFESIONALES
AUTORIZACIÓN DE IMPRESIÓN DE TRABAJO DE TITULACIÓN

FECHA: 15 de Enero de 2018.

ASUNTO: **AUTORIZACIÓN DE IMPRESIÓN DE TESIS PROFESIONAL.**

A QUIEN CORRESPONDA:

Por medio de la presente hago constar que el (la) C:

JOSÉ LUIS MELGAREJO ORTIZ

pasante de la carrera de INGENIERÍA EN SISTEMAS COMPUTACIONALES con No. de Control 142T0194 ha cumplido satisfactoriamente con lo estipulado por el **Manual de Procedimientos para la Obtención del Título Profesional de Licenciatura** bajo la **opción Titulación Integral (Tesis Profesional)**

Por tal motivo se **Autoriza** la impresión del **Tema** titulado:

"SISTEMA PLANIFICADOR DE LOS RECURSOS DE LA EMPRESA GRUPO MULTICLIMAS"

Dándose un plazo no mayor de un mes de la expedición de la presente a la solicitud del Acto de Recepción para la obtención del Titulo Profesional.

ATENTAMENTE

ING. GERBACIO TLAXALO ESPINOZA
DIVISIÓN DE ESTUDIOS PROFESIONALES

DIVISION DE ESTUDIOS PROFESIONALES

Archivo.

VER. 01/03/09

F-SA--39

Agradecimientos

A mis padres, dado que sin su apoyo incondicional no habría podido llegar hasta este punto, quienes me alentaron a seguir adelante, les estaré por siempre agradecido puesto que son quienes me han enseñado con hechos, a lo largo de mi vida, a no rendirme, de quienes aprendí a ser perseverante y es por ellos que he logrado tantas cosas.

A mis familiares, que estuvieron conmigo durante toda mi formación académica, quienes me brindaron apoyo en todo momento, de quienes igualmente aprendí muchas cosas y valores, con quienes pasé divertidos y agradables momentos que jamás olvidaré.

A mis docentes, por su tiempo, paciencia, conocimiento y experiencia proporcionadas durante la etapa de formación académica, sin mencionar su gran apoyo, motivación y confianza que fueron pieza clave en la resolución para alcanzar el éxito.

A mis amigos, con quienes, a pesar de estar siempre divididos, siempre encontrábamos la manera de apoyarnos mutuamente durante nuestra formación profesional, especialmente, a mis amistades más cercanas y a quiénes estuvieron conmigo brindándome apoyo, motivación y aconsejándome hasta el cansancio.

A todas las personas que conocí en el camino, que de igual manera me brindaron apoyo, me hicieron ver las cosas desde otro punto de vista y, junto a mis amistades más cercanas, compartía mis logros y metas alcanzadas, por muy simples que estas fuesen.

Dedicatoria

El presente trabajo de tesis está dedicado a:

Mi hermanita, Zuleyma, por ser el centro de mi familia, quien es y será el pilar de mi determinación, mi motivación para salir adelante y a pesar de las adversidades, superar todos los obstáculos que se presenten en el camino; a quien se atribuyen mis aspiraciones por llegar aún más lejos y seguir creciendo tanto de forma personal como profesional en la vida.

Todo este trabajo, mi éxito e hitos alcanzados, están dedicados a Zuleyma.

Resumen

La aplicación desarrollada en el presente proyecto mantiene un enfoque y desarrollo a la medida para la automatización de diversas prácticas de negocio, principalmente relacionadas con aspectos operativos, así como la integración de información de logística, recursos humanos, entre otras diversas funciones que se llevan a cabo en la empresa Grupo Multiclimas.

Entre los beneficios más destacados derivados del desarrollo del presente proyecto, es la automatización y optimización de procesos empresariales de alto grado de importancia, puesto que en tales procesos se manipulan principalmente las utilidades de la empresa.

Además, permitió el acceso a información centralizada y unificada sin tener que tomar en cuenta la ubicación y horario de acceso. Simultáneamente existe la posibilidad de compartir información y registros que incluso se pueden crear desde la misma aplicación entre los distintos miembros y componentes de la organización.

Dada la unificación y centralización de la información se obtuvo como resultado la depuración de datos y operaciones que no resultaron ser de suma importancia para la organización. Cabe destacar que el sistema se llevó a cabo de acuerdo a las especificaciones del cliente y de las situaciones que se presentaron internamente en la compañía.

Dado que los componentes de la aplicación ERP mantienen una interacción entre sí; consolidan las operaciones realizadas, aportando nuevos beneficios; entre los cuales destacan la resolución tanto de problemas contables como mercantiles de la empresa Grupo Multiclimas.

Índice

Índice de figuras

Índice de requerimientos

Índice de requerimientos funcionales

Índice de requerimientos no funcionales

Capítulo 1: Generalidades del proyecto

1.1 Introducción

La falta de registros claros y ordenados sobre inventario, ventas y facturación son algunos de los muchos problemas a los que se enfrentan cotidianamente las PyMES, tal fue el caso de la empresa Grupo Multiclimas, en donde además de no contar con un correcto orden y control en los registros que se llevaban a cabo sobre los movimientos que se realizaban en la empresa, dichos registros se realizaban de forma manual dado que no contaban con herramientas tecnológicas que dieran paso a la implementación de sistemas que permitieran la automatización de las actividades más importantes realizadas en la empresa.

Partiendo de ese punto, se necesitó proceder con investigaciones y estudios sobre sistemas que permiten automatizar las actividades más importantes que se efectúan internamente en las empresas. Como resultado, la opción más viable que planteó solución a las situaciones por las que pasaba la empresa Grupo Multiclimas, fue un sistema de planificación de recursos empresariales (ERP). El cual se desarrolló como aplicación web dando como resultado una herramienta de trabajo modular y versátil que consume la mínima cantidad de recursos computacionales.

La aplicación ERP cuenta, entre todas sus funciones, con una administración sistemática y precisa sobre los productos existentes en inventario, asimismo permite llevar un control sobre la cantidad de unidades por productos con los que cuenta cada una de las sucursales de la empresa Grupo Multiclimas.

Por otro lado, se implementó en la aplicación la funcionalidad de efectuar ventas, lo que representa una considerable reducción en tiempos de espera, puesto que es la misma aplicación la que lleva a cabo el proceso de gestión de productos en el stock para completar satisfactoriamente las ventas realizadas, por lo tanto, una sola operación representa un serie de actividades ejecutadas en paralelo, dando como resultado congruencia tanto en la información registrada en el sistema como en las existencias físicas de los productos que dispone la empresa.

De igual manera se implementó un módulo de administración del sistema, en el cual, las funciones más destacadas es el control de entradas de producto, puesto que además de controlar las utilidades de la empresa, es posible realizar solicitudes de producto a proveedores estableciendo una comunicación vía correo electrónico. Por otro lado, se cuenta con la opción de controlar los datos tanto de clientes como de proveedores, lo que permite permanecer en contacto con los mismos con sólo actualizar sus registros. Al mismo tiempo se implementaron herramientas que permitieron generar documentos con los cuales es posible llevar registros precisos y ordenados tanto de manera física como virtual.

1.2 Descripción de la problemática

En la actualidad, uno de los principales problemas a lo que se enfrentan algunas micro y pequeñas empresas, es la falta de registros en su control de inventarios, aspecto administrativo que es pocas veces atendido, de no contarse con registros congruentes, un responsable, políticas o sistemas que le ayuden a esta sencilla pero pesada tarea entonces la función de inventario deja de operar con efectividad y la empresa se desvía de su objetivo primordial: obtener utilidades, las cuales son la repercusión de una óptima gestión del inventario de toda empresa.

Una contabilidad concreta de producto en inventario conlleva a una mejor toma de decisiones, además de reducir costos al no saturar el inventario con excesos o carencias de producto, dada la ausencia de control de inventario, se puede percibir un servicio deficiente brindando así una mala atención al cliente. El no contar con niveles adecuados de inventario puede evitar el correcto flujo de efectivo.

La ausencia o mal control de inventario genera reducción de rentabilidad, esto se traduce como un gran costo para la organización. Al contarse con un exceso de inventario, o adquirir producto adicional innecesario, se corre el riesgo de que con el paso del tiempo pueda extraviarse o estropearse por diversas razones, tales como salidas no registradas o almacenamiento inapropiado.

Una planificación deficiente es ocasionada por un seguimiento de inventario incorrecto, por lo que los pedidos de los clientes no se pueden entregar en tiempo y forma. El partir de un mal inventario puede llevar a la organización a no planificar adecuadamente y a generar costos extras por no poder completar los pedidos solicitados por el cliente.

Actualmente, una empresa que emplea un control de inventario con registros físicos se encuentra en gran desventaja en diversos factores frente empresas que emplean recursos tecnológicos, uno de estos factores es el tiempo, por ejemplo, un recuento de inventario a mano es una labor que puede llegar a prolongarse por días, disminuyendo así la eficiencia del negocio y, por consiguiente, pérdida de utilidades.

Al mismo tiempo, se puede generar un caso de desestimación oportuna de documentos, tales como reportes, órdenes, cheques e incluso facturas y estados de cuenta en los que se requiere un análisis y contabilidad previos que, de igual manera, son tareas en las que se invierte tiempo.

Otra de las cuestiones que se pueden presentar son los errores de cálculo en el inventario, obteniendo datos imprecisos cuya actualización varía en función del último recuento que se realizó en el almacén, de esta manera, se desestima la cantidad de unidades que se tengan existencia lo que se representa como pérdida de efectivo para la empresa.

1.3 Objetivos

1.3.1 Objetivo general

Diseñar un sistema de planificación de recursos empresariales para centralizar la información de almacén y ventas de la empresa Grupo Multiclimas con el propósito de obtener un mejor control y seguimiento de sus registros.

1.3.2 Objetivos específicos

- Implementar un módulo de ventas al sistema general para generar registros de las entradas y salidas de producto.
- Desarrollar un módulo de punto de venta que permita a los clientes consultar existencia sobre un producto determinado en la sucursal más cercana.
- Diseñar y crear una base de datos relacional, así como los procedimientos almacenados correspondientes que favorezcan la normalización y unificación de la información.
- Diseñar y desarrollar servicios y formularios para facilitar el uso del sistema por sus operadores permitiendo un control óptimo del inventario.

1.4 Justificación

De acuerdo a Hong y Kim (2001) un sistema ERP (sistema de planificación de recursos empresariales del inglés *Enterprise Resource Planning*) es una solución a problemas crónicos de la industria de tecnologías de la información (IT), por ejemplo, reducen costos, poseen una rápida implementación y son sistemas de alta calidad.

De igual manera, Holland y Light (1999) señalan que los sistemas ERP automatizan actividades corporativas centrales tales como manufactura, recursos humanos, financias y administración de cadenas de abastecimiento. Dichos sistemas se venden en base a incorporar la "mejor práctica" que facilite la rápida toma de decisiones, reducción de costos y un mejor control gerencial.

El deficiente control de inventario y la ausencia de registros de ventas son algunas de las situaciones que se presentaron en la empresa Grupo Multiclimas, dichas situaciones afectaban la gestión interna de la empresa, reduciendo dramáticamente la obtención de utilidades.

Por tal motivo se desarrolló una aplicación web la cual se encargó de fungir como sistema ERP para la empresa Grupo Multiclimas, de esta manera se logró determinar la solución a muchas situaciones que se presentaron con el paso del tiempo en la compañía.

La importancia del proyecto radica en garantizar la facilidad de comprensión y aplicación de la información, puesto que la aplicación se instaló en un servidor central, permite que toda la información de las distintas sucursales pertenecientes a Grupo Multiclimas se pueda centralizar y unificar, por consiguiente, es posible consumir dicha información mediante la aplicación web, a la que se puede acceder desde prácticamente cualquier dispositivo que cuente con una conexión a Internet. Sumado a esto, el desarrollo de la aplicación es responsivo, es decir, es capaz de adaptarse a distintos tamaños de monitores para computadoras e incluso algunos dispositivos móviles como tabletas.

Como se menciona anteriormente, el desarrollo del proyecto tiene como fin brindar soporte a los clientes de la empresa, rápidos tiempos de respuesta a los problemas que se puedan presentar, así como un eficiente manejo de la información, contando con datos útiles y necesarios que

permitan una mejor toma de decisiones, mantener un adecuado nivel de almacén y recurrir a la implementación de la tecnología para potenciar la competitividad de Grupo Multiclimas.

La implementación de un sistema ERP supone grandes beneficios en diversos aspectos, uno de ellos es el económico, dado que al permitir la integración entre distintas áreas como: ventas, recursos humanos, almacén, finanzas, se reducen tiempos de intercambio de información entre las distintas áreas junto con los respectivos costos que representa dicho proceso en comparación de utilizar un sistema computarizado aislado o manual.

Por otro lado, dentro del aspecto tecnológico, algunos de los beneficios que se obtienen de la implementación de sistemas ERP es la automatización y optimización de procesos gerenciales, procesos rutinarios que se realizan en la empresa que, además de automáticos, reducen drásticamente su margen de error, además de que un ERP provee acceso a la información de manera confiable y segura; una empresa con una correcta gestión de la información añade confianza en los clientes con los que trabajan haciendo que los lazos de confianza crezcan exponencialmente.

Capítulo 2: Marco teórico

2.1 Sistema de Planificación de Recursos Empresariales (ERP)

Chiesa (2004) define a un sistema ERP como una aplicación informática que permite gestionar todos los procesos de negocio de una compañía en forma integrada. Sus siglas provienen del término en inglés ENTERPRISE RESOURCE PLANNING. Generalmente este tipo de sistemas está compuesto de módulos como Recursos Humanos, Ventas, Contabilidad y Finanzas, Compras, Producción entre otros, brindado información cruzada e integrada de todos los procesos del negocio. Este software debe ser parametrizado y adaptado para responder a las necesidades específicas de cada organización. Una vez implementado un ERP permite a los empleados de una empresa administrar los recursos de todas las áreas, simular distintos escenarios y obtener información consolidada en tiempo real.

Por otro lado, Vera (2006) menciona que los sistemas del tipo ERP se han definido como sistemas globales de planificación de los recursos y de gestión de la información que de forma estructurada puede satisfacer la demanda de las necesidades de gestión de una empresa. Además, detalla que estos paquetes de software disponen de módulos específicos para cubrir las exigencias de cada de las áreas funcionales de la empresa, de tal manera que crean un flujo de trabajo entre los distintos usuarios. Este flujo permite evitar tareas repetitivas, y mejora la comunicación en tiempo real entre todas las áreas que integran la empresa.

2.2 Metodología de desarrollo ágil: Scrum

La metodología Scrum para el desarrollo ágil de software es un marco de trabajo diseñado para lograr la colaboración eficaz de equipos en proyectos, que emplea un conjunto de reglas y artefactos y define roles que generan la estructura necesaria para su correcto funcionamiento (Cadavid, Martínez, y Vélez, 2013).

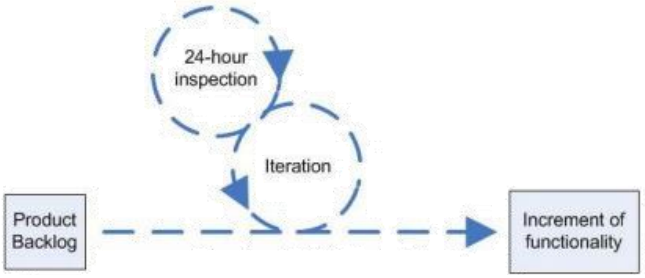

Figura 2.1: Esquema de Scrum.

Scrum implementa este esquema iterativo e incremental (ver Figura 2.1) por medio de tres roles: el dueño del producto (*Product Owner*), el equipo (*Team* o *Scrum Team*), y el maestro Scrum (*ScrumMaster*) (Mahnic y Drnovscek, 2005).

El dueño del producto es responsable de representar los intereses de todos con su participación en el proyecto y su sistema resultante. El dueño del producto mantiene la *Pila de Trabajo*, es decir, una lista priorizada de los requerimientos del proyecto con tiempos estimados para convertirlos en productos funcionales completados (Mahnic y Drnovscek, 2005).

Por otro lado, el equipo es el responsable del desarrollo de la funcionalidad. Los equipos son auto administrados y auto organizados, son los responsables de como descifrar la *Pila de trabajo* para convertirla en una funcionalidad incremental dentro de una iteración y como administrar su propio trabajo para llevar a cabo dichas actividades. Los miembros de cada equipo son colectivamente responsables del éxito de cada iteración y del proyecto como un todo (Mahnic y Drnovscek, 2005).

Por último, se cuenta con el rol de *Scrum Master*, el cual cuenta con la función de asegurar que el equipo está adoptando la metodología, sus prácticas, valores y normas, puesto que es el líder del equipo, pero no gestiona el desarrollo (Cadavid, Martínez, y Vélez, 2013).

A cada iteración dentro del marco de trabajo de Scrum se le conoce como Sprint, un Sprint es el procedimiento de adaptación a las variables en un entorno cambiante (requerimientos, tiempo, recursos, conocimiento, tecnología, etc.). El equipo de Scrum se organiza por sí mismo para producir un nuevo producto ejecutable incremental en un Sprint que dura aproximadamente treinta días del calendario. Scrum se puede dividir de forma general en tres fases, las cuales se pueden entender como reuniones. Las reuniones forman parte de los artefactos de esta metodología junto con los roles y elementos que lo forman (ver Figura 2.2) (Gallego, 2012).

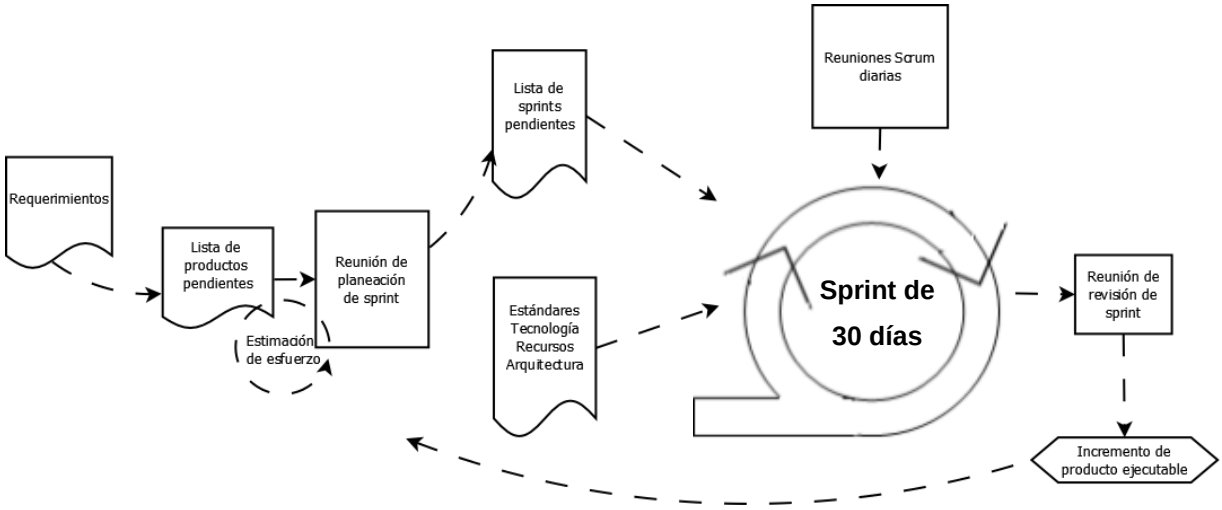

Figura 2.2: Prácticas y entradas de Sprint.

Planificación del Backlog (Pila de trabajo)

Se define un documento en el que se reflejarán los requisitos del sistema por prioridades. Durante esta fase se define la planificación del Sprint 0, en el cual se decide cuáles serían los objetivos y el trabajo por realizar para dicha iteración. Además, de la reunión se obtiene un Sprint Backlog, el cual trata de una lista de tareas y es el objetivo más importante del Sprint (Gallego, 2012).

Seguimiento del Sprint

Durante esta fase el equipo se reúne para realizar una reunión de máximo quince minutos conocida como "Scrum diario", en donde cada miembro debe responder tres preguntas:

- ¿Qué trabajo se realizó desde la última reunión de Scrum diario?
- ¿Qué trabajo se realizará para la próxima reunión?
- ¿Se presentaron obstáculos?

En caso de presentarse inconvenientes, se deben solucionar para poder continuar con el desarrollo del proyecto (Mahnic y Drnovscek, 2005).

Revisión del Sprint

Al finalizar un Sprint se realiza una revisión del incremento que se ha generado. Se presentan los resultados finales y una demo o versión ejecutable del avance, esto representa una mejora de la retroalimentación con el cliente (Gallego, 2012).

Posterior a la revisión del Sprint y previo a la siguiente reunión de planeación de Sprint, el *Scrum Master* también mantiene una reunión de retrospectiva de Sprint con el fin de fomentar al equipo a repasar, dentro del marco de trabajo del proceso de Scrum, el proceso de su desarrollo para volverlo más efectivo y agradable para el siguiente Sprint (Mahnic y Drnovscek, 2005). Un esquema más detallado acerca del flujo de Sprints de Scrum se muestra en la Figura 2.3.

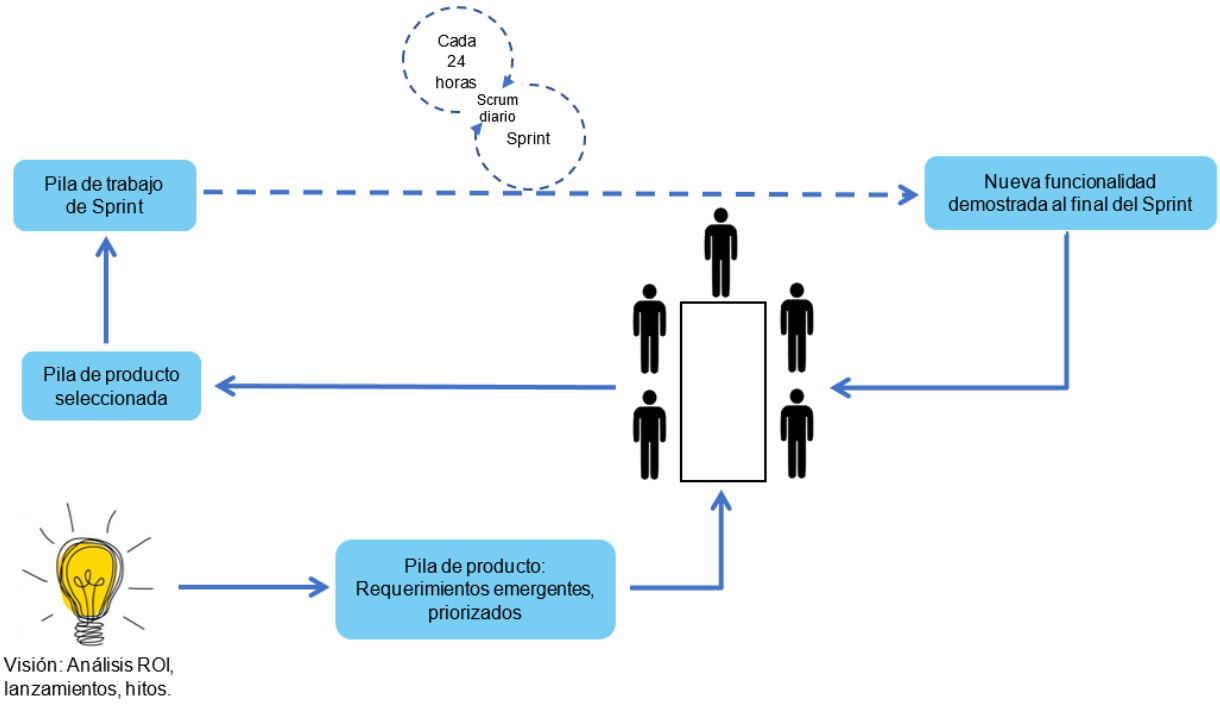

Figura 2.3: Esquema detallado del flujo de Scrum.

Ventajas de utilizar Scrum (Figueroa, Solís y Cabrera, 2008)

- Apropiado para entornos volátiles
- Estar preparados para el cambio, significa reducir su coste.
- Planificación más transparente para los clientes, conocen las fechas de entrega de funcionalidades. Vital para su negocio
- Permitirá definir en cada iteración cuales son los objetivos de la siguiente
- Permite tener realimentación de los usuarios muy útil.

2.3 Estilo arquitectónico

El objetivo del proceso de diseño arquitectónico por el cual debe pasar el software es establecer una solución para satisfacer requerimientos técnicos y operacionales solicitados por el cliente. Dicho proceso delimita los componentes que deben conformar el sistema, las condiciones en que se deben relacionar y la forma en que sus relaciones deben llevar a cabo la función requerida.

De la Torre Llorente, Castro, Barros y Nelson (2010) señalan que el estilo arquitectónico se conforma de una agrupación de componentes, de conexiones entre dichos componentes y un conjunto de restricciones que definen el modo de comunicación entre dos componentes conectados. En función de la aplicación a desarrollar es la condición en que se deben organizar los estilos arquitectónicos; los principales aspectos de una aplicación son los siguientes: comunicaciones, despliegue, dominio, interacción y estructura. A continuación, se enlistan y describen brevemente algunos de los principales estilos arquitectónicos.

2.3.1 Cliente/Servidor

Este estilo arquitectónico (ver Figura 2.4) delimita una relación entre dos aplicaciones, en las que una (cliente) realiza peticiones a la segunda (servidor), la cual provee los datos solicitados (de la Torre Llorente, Castro, Barros y Nelson, 2010).

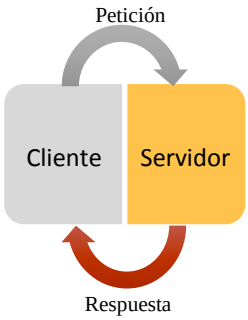

Figura 2.4: Esquema ilustrativo del estilo Cliente/Servidor.

Sus principales características son:

- Estilo ideal para sistemas distribuidos.
- El sistema se compone de tres partes fundamentales: aplicación en cliente, aplicación en servidor y una red que permita la comunicación entre ambos.

- Determina la relación entre el cliente y el servidor; en la cual, el cliente realiza peticiones y el servidor debe enviar respuestas concretas.
- Permite el uso de un extenso rango de protocolos y formatos de datos para la transmisión y recepción de datos.

2.3.2 Basado en componentes

El estilo arquitectónico basado en componentes (ver Figura 2.5) determina un enfoque al diseño de sistemas como una agrupación de componentes que exponen interfaces definidas correctamente y que trabajan en conjunto con el fin de solucionar un problema determinado (de la Torre Llorente, Castro, Barros y Nelson, 2010).

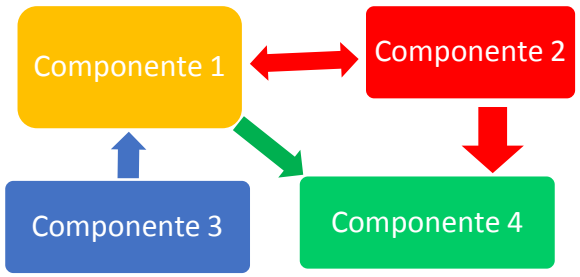

Figura 2.5: Esquema ilustrativo del estilo basado en componentes.

Sus principales características son:

- Estilo que se ajusta al diseño de aplicaciones que parten de componentes individuales.
- Hace énfasis en la descomposición del sistema en componentes con interfaces correctamente definidas.
- Determina un acercamiento al diseño a través de componentes que se comunican por medio de interfaces exponiendo métodos, eventos y propiedades.
- Sus componentes pueden ser extendidos a partir de otros componentes obteniendo nuevos comportamientos.

2.3.3 Orientado a objetos

El estilo arquitectónico orientado a objetos (ver Figura 2.6) es un estilo que define al sistema como una colección de objetos que colaboran en conjunto en lugar de colaborar como una colección de procedimientos. Los objetos son discretos, escasamente acoplados e

independientes, se comunican por medio de interfaces y permiten el envío y recepción de mensajes (de la Torre Llorente, Castro, Barros y Nelson, 2010).

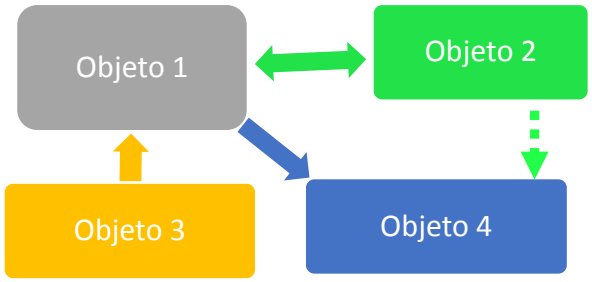

Figura 2.6: Esquema ilustrativo del estilo orientado a objetos.

Sus principales características son:

- Estilo para el diseño de aplicaciones establecido en una determinada cantidad de unidades lógicas y código reutilizable.
- Determina la utilización de objetos que contienen los datos y el comportamiento para manipular tales datos, además cada uno cuenta con un rol o responsabilidad distinta.
- Basado en la reutilización empleando encapsulamiento, modularidad, polimorfismo y herencia.
- Contrasta con el enfoque procedimental donde existe una secuencia de tareas y acciones predefinidas. El enfoque orientado a objetos apunta a realizar diversas tareas por medio de la interacción entre objetos.

2.3.4 En capas (N-Layer)

El estilo arquitectónico en capas (ver Figura 2.7) se basa en una distribución jerárquica de roles y responsabilidades para brindar una segmentación efectiva de los problemas a resolver. Los roles definen el tipo y forma de interacción con otras capas y las responsabilidades la implementación de su funcionalidad (de la Torre Llorente, Castro, Barros y Nelson, 2010).

Figura 2.7: Esquema ilustrativo del estilo en capas.

Sus principales características son:

- Descomposición de los servicios de tal manera que las interacciones que se presenten sean entre capas que se encuentren colindando unas con otras.
- Las capas de la aplicación pueden alojarse en la máquina cliente, o bien, encontrarse en diferentes equipos.
- Cada componente perteneciente a determinada capa cuenta con comunicación a componentes de otras capas a través de interfaces.
- Cada nivel adiciona responsabilidades y abstracciones del nivel inferior.
- No existe una cantidad restringida de capas, éstas varían en función de las necesidades del proyecto.

Para el desarrollo de este proyecto, se pretende emplear el estilo arquitectura en capas, el cual se describe a detalle a continuación.

2.4 Arquitectura en N capas (N-Layer) para aplicaciones

2.4.1 Diferencias entre Capas (Layers) y Niveles (Tiers)

A menudo los términos “Capas” (Layers) y “Niveles” (Tiers) se confunden y/o denominan de forma incorrecta, por lo tanto, es imprescindible distinguirlos.

De acuerdo con de la Torre Llorente, Castro, Barros, y Nelson (2010) las capas son el modelo que determina la segmentación lógica de los componentes y su función asignada sin tomar en cuenta la ubicación física de los componentes en los servidores u otros equipos requeridos.

Por otro lado, se encuentran los niveles, cuya funcionalidad se centra en determinar la distribución física de componentes y roles asignados en servidores separados, por ejemplo:

- 2-Tier
- 3-Tier

  ⋮
- N-Tier

A continuación, se muestran tanto un esquema de capas (N-Layer) en la Figura 2.9 y un esquema de Niveles (3-Tier) en la Figura 2.8 con el fin de mostrar sus diferencias de manera gráfica.

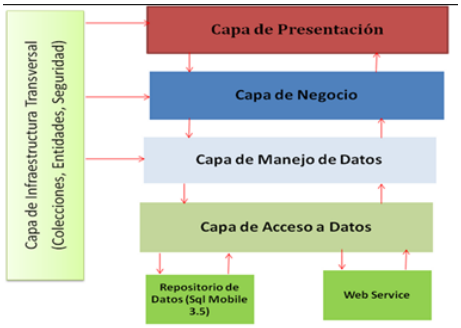

Figura 2.9: Arquitectura tradicional N-Layer (Lógica).

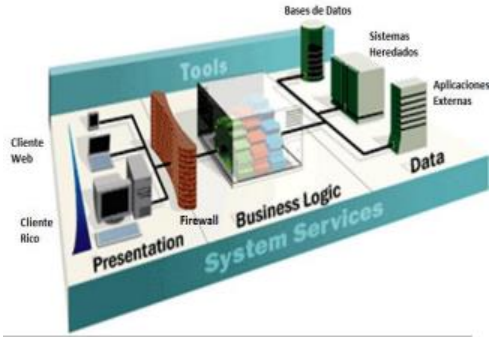

Figura 2.8: Arquitectura 3-Tier (Física).

2.4.2 Programación en N-Capas

Como se mencionó anteriormente, el estilo arquitectónico en capas se basa en una distribución jerárquica de roles y responsabilidades para brindar una segmentación efectiva de los problemas a resolver. Los roles definen el tipo y forma de interacción con otras capas y las responsabilidades la implementación de su funcionalidad. Para el desarrollo de este proyecto, se pretende usar un estilo arquitectónico de 3 Capas, como se muestra en la Figura 2.10.

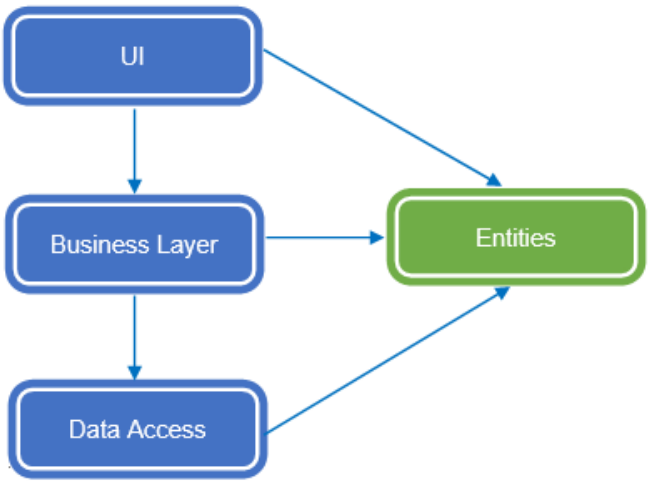

Figura 2.10: Estilo arquitectónico en 3-Capas.

2.4.3 Tipos de capas

2.4.3.1 Capa de Presentación (UI)

La función de la capa de presentación es mostrar al usuario los controles y contenido visual de la aplicación. La capa de presentación mantiene comunicación directamente con la de un nivel inferior, ya sea la capa de Negocios (Business Layer) o alguna otra capa intermedia, la cual se encarga de enviar mensajes a los objetos de la capa de Acceso de Datos (DAL - Data Access Layer) para proporcionar la información solicitada por el usuario a la capa de presentación.

2.4.3.2 Capa de Negocios

Se denomina capa de negocios o lógica de negocios a la capa responsable del procesamiento de la información que tiene lugar en la aplicación. Dentro de esta capa intermedia se contienen objetos que corresponden a las entidades de la aplicación, esta capa comprende una alta capacidad de mantenimiento y reutilización. Puesto que a diferencia de la capa de presentación o de acceso de datos, que pueden ser alteradas o removidas sin mayores complicaciones, la capa de negocios requiere de mayor capacidad de mantenimiento, puesto que, al mantener comunicación con ambas capas, se requiere una reestructuración concreta en los objetos para poder mantener una comunicación estable entre capas dentro de la aplicación.

2.4.3.3 Capa de Acceso de Datos

Esta capa se encarga de la sincronización de información, es aquí donde únicamente se debe contar con las conexiones a la base de datos y al servidor, también es en esta capa donde se ejecutan los procedimientos almacenados, los cuales reciben los datos requeridos de la capa de negocios para realizar su función. En función de la complejidad del sistema, dependerá de la cantidad de equipos en los que cada capa puede residir; todas estas capas se pueden encontrar frecuentemente alojadas en un mismo ordenador, por otro lado, en sistemas complejos se pueden llegar a encontrar distintas capas en distintos equipos (Henríquez, Huerta y Grados, 2010).

2.4.3.4 Entidades

Como se muestra en la Figura 2.10, las capas que se mencionaron anteriormente pueden acceder y hacer uso de sus métodos y objetos, puesto que esta capa funciona como medio de comunicación entre el resto de capas.

La capa de entidades es una colección de objetos enfocada al estilo POCO; acrónimo de “Plain Old CLR Object” y se trata de un objeto estándar o simple libre de herencia o atributos, este tipo de objetos se crean en el “Common Language Runtime” (CLR) del Framework .NET. Este tipo de objetos son frecuentemente utilizados en contraste con objetos complejos o especializados que usualmente se requieren por frameworks de mapeo objeto-relacional. En esencia, los objetos POCO no dependen de ningún framework.

2.5 Paradigma de programación

Se puede definir a un paradigma como un modelo, el cual se admite sin contradicción alguna y proporciona la base para la resolución a problemas específicos.

Por lo antes mencionado, se puede decir que un paradigma de programación (ver Figura 2.11) se trata de un estilo de programación, el cual proporciona la teoría o conjunto de teorías que indican un procedimiento para llevar a cabo determinados procesos de cómputo, así como la estructuración y organización de las tareas que debe realizar el software a desarrollar. Con frecuencia, los lenguajes de programación suelen implementar de forma parcial varios paradigmas (Rodríguez, 2011).

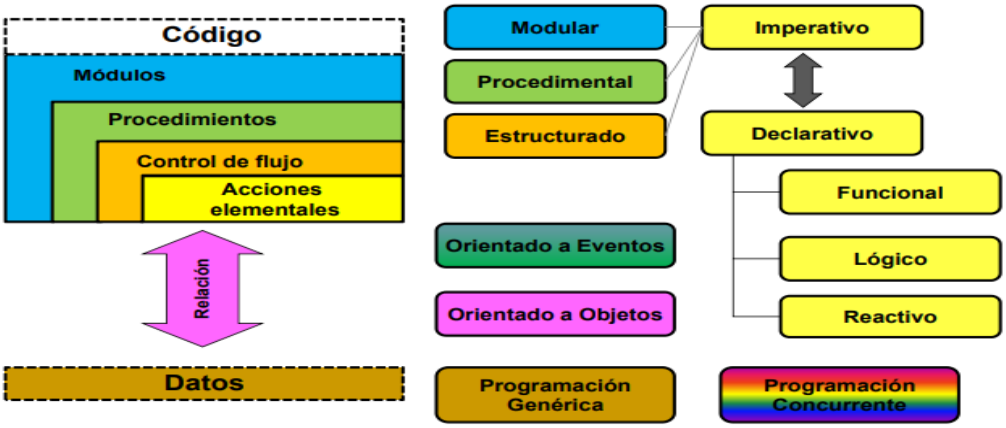

Figura 2.11: Tipos de paradigmas de programación.

2.6 Programación Orientada a Objetos (POO)

De acuerdo a Smith (2011), la programación orientada a objetos es la práctica de crear una arquitectura de software que permite la flexibilidad por medio de un diseño modular.

A diferencia de otros paradigmas como el de programación imperativa, donde se basa en la asignación de una serie de sentencias, que describen la programación en términos de estado y

que modifican dicho estado del programa. Las instrucciones de los lenguajes de programación pertenecientes a este paradigma se ejecutan según un flujo secuencial (excepto si las instrucciones se encuentran dentro de estructuras de control condicionales o cíclicas).

En el paradigma declarativo, que, en contraste a la programación imperativa, donde se desarrolla un algoritmo en donde se describen las instrucciones necesarias para solucionar un problema, aquí las sentencias se utilizan para definir el problema a solucionar, se programa definiendo lo que se busca resolver a nivel de usuario, pero no las instrucciones que dan solución al problema. Esto último se realiza por medio de mecanismos de inferencia a partir de la descripción realizada.

Gallardo López, y Pomares Puig (2008) definen como concepto fundamental de la POO el término "objeto"; el cual se trata de una entidad que comprende datos (o variables) y funciones (métodos).

El modelo de programación orientada a objetos cuenta con cuatro principios fundamentales:

- Abstracción
- Encapsulamiento
- Herencia
- Polimorfismo

2.6.1 Abstracción

Según Sanders y Cumaranatunge (2007), es posible describir a la abstracción como un modelo o ideal, donde no se cuenta con detalles, pero se cuentan con los parámetros generales que se pueden llenar con información. La abstracción es una propiedad que tiene como función tomar las principales características de un objeto, así como su comportamiento y diferenciarlas de su implementación.

Otra idea fundamental de la programación orientada a objetos es el concepto "clase". Una clase se puede definir como un modelo o prototipo abstracto de un grupo de objetos que define datos y funciones comunes al grupo de objetos pertenecientes a dicha clase, donde cada uno de los cuales se distingue por su propio estado y capacidad de realizar distintas operaciones (Joyanes Aguilar, 1996).

2.6.2 Encapsulamiento

El encapsulamiento (también conocido como encapsulación u ocultación de la información) es una propiedad cuya función consiste en asegurar que la información contenida dentro de un objeto se encuentra oculta. En esencia, es el acto de ocultar toda la información posible de un objeto que no contribuyan a sus principales características (Joyanes Aguilar, 1996).

De acuerdo con Sanders y Cumaranatunge (2007), dentro del contexto de la POO, al encapsulamiento también se le suele llamar "caja negra"; metáfora que se refiere a una entidad estructural dentro de un modelo abstracto cuyo funcionamiento permite al usuario observar el funcionamiento que se halla entre la entrada y salida sin conocer sobre su funcionamiento interno. Las bondades del concepto de caja negra es que no es necesario preocuparse sobre el funcionamiento interno del sistema, sólo se necesita saber cómo utilizarlo.

Para poner en práctica el encapsulamiento, es necesario que cada clase conste de dos partes:

- La interfaz: que permita capturar sólo la vista externa de la clase.
- La implementación: que contenga la representación de la abstracción, así como los componentes que permitan realizar la funcionalidad deseada.

2.6.3 Herencia

El tercer concepto clave de la programación orientada a objetos es la herencia. La herencia se refiere a la relación entre clases, donde una clase comparte propiedades, métodos y eventos definidos en una o más clases (a esta relación se le conoce como herencia simple y múltiple respectivamente) (Sanders y Cumaranatunge, 2007).

Joyanes Aguilar señala que para que un sistema complejo pueda implementar el concepto de herencia, se necesita de una *jerarquía*. Propiedad que permite el ordenamiento de la abstracción. Las dos jerarquías más importantes que se encuentran en un sistema complejo son:

- Estructura de clases: generalización/especialización, "*es un*" del inglés "*is a*".
- Estructura de objetos: agregación, "*parte de*" del inglés "*part of*".

2.6.4 Polimorfismo

El polimorfismo en programación orientada a objetos es la propiedad o técnica empleada para optimizar la funcionalidad en un grupo de objetos. Lo que permite que una entidad tenga la

posibilidad de tomar muchas formas. Prácticamente, el polimorfismo permite tomar objetos de diferentes clases e indicarles realizar diversas acciones, estas acciones se ejecutan, de diferentes formas, dependiendo el objeto al cual se esté haciendo referencia.

Existen tres tipos de polimorfismo:

- Sobrecarga: se presenta cuando existen dos o más funciones con el mismo nombre, con funcionalidad similar, pero en clases independientes una de la otra.
- Paramétrico: se presenta cuando existen funciones con el mismo nombre, dentro de la misma clase, pero usan diferentes parámetros (nombre o tipo de dato). Se selecciona el método en función de los tipos de datos que se envíen.
- Inclusión: se trata de métodos a los que se pueden llamar, sin tener conocimiento sobre su tipo, de esta manera no se toman en cuenta los detalles de las clases especializadas, utilizando una interfaz común.

2.6.5 Modularidad

La modularidad es otra propiedad de la programación orientada a objetos cuya función consiste en subdividir una aplicación en partes pequeñas (conocidas como módulos), cada una de las cuales debe ser independiente de la tanto como sea posible de la aplicación general y del resto de las partes (Joyanes Aguilar, 1996).

De acuerdo a Liskov (1986), la modularización consiste en segmentar un programa en módulos que se pueden compilar por separado, pero que tienen conexión a otros módulos. De igual manera que el encapsulamiento, los lenguajes soportan la modularidad de distintas formas.

2.7 Breve historia de Internet

La creación de las bases de todo lo que actualmente es Internet se remonta a la década de los sesenta, bajo la dirección de DARPA, la Agencia de Proyectos de Investigación Avanzados de Defensa (del inglés *Defense Advanced Research Projects Agency*) de los Estados Unidos, originalmente denominada ARPA, inició un programa de investigación de técnicas y tecnologías con el fin de unir diferentes redes de conmutación de paquetes, de esta manera existiría una comunicación entre los ordenadores conectados a estas redes de manera fácil y transparente de forma paralela y en tres centros de investigación (sin que existiera conocimiento

entre ellos de las actividades que realizaban cada uno); los tres centros de investigación participantes fueron el *Massachusetts Institute of Technology* (MIT) dentro de lo que comprenden los años de 1961 y 1967, *The RAND Corporation* entre 1962 y 1965, y el *National Physical Laboratory* (NPL) de 1964 a 1967 (Mateu, 2012).

2.7.1 Breve historia de la web

Luján Mora (2002) señala que el surgimiento de la WWW (World Wide Web), o mejor conocida como la "web", se remonta a 1989 en el CERN (Centro Europeo de Investigación Nuclear) como el desarrollo de un sistema a cargo de Tim Berners-Lee cuyo propósito era facilitar la accesibilidad a la información del CERN.

El nacimiento de la web se debe a tres elementos clave:

- HTML: *Hyper Text Markup* Language, empleado para desarrollar los contenidos de la web, tomando *Standard Generalized Markup Language* (SGML) como base.
- HTTP: *Hyper Text Transfer Protocol*, empleado como protocolo de comunicación entre los equipos de la web, se encarga de transferir páginas web y demás recursos.
- URL: *Uniform Resource Locator*, como identificador de distintos recursos de Internet.

2.7.2 Navegadores web

Un navegador web es una herramienta de software que permite a los usuarios realizar diversas tareas, tales como recuperar y visualizar el contenido de diversas páginas web por medio de internet y ejecutar código embebido en dichas páginas web. Los navegadores web se encuentran disponibles en la internet sin cargo alguno, usualmente los usuarios eligen uno en base a experiencia satisfactoria de navegación, experiencia particularmente determinada por la velocidad del navegador (Nielson, Williamson y Arlitt, 2008).

2.8 Lenguajes de programación para desarrollo web

2.8.1 Python

Halterman (2011) señala que Python es un lenguaje de programación de alto nivel de propósito general, cuya filosofía de diseño se enfoca en la legibilidad de código.

Los programas hechos en Python se deben escribir con una estructura particular que permite a los programadores definir conceptos en menos líneas de código a diferencia de otros lenguajes

de programación, la estructura de Python tiene la intención de crear programas legibles a pequeña y gran escala.

Python soporta múltiples paradigmas de programación, incluyendo el orientado a objetos, imperativo y estilos de programación (Van Rossum, 2007).

En la Figura 2.12 se muestran las compañías que utilizan Python.

Figura 2.12: Compañías y organizaciones que utilizan Python.

2.8.2 JavaScript

JavaScript es un lenguaje de programación orientado a objetos, dic ho lenguaje es a menudo mejor conocido por su uso como lenguaje scripting para el desarrollo web. JavaScript es la clave de desarrollo de un bloque de construcción de Dynamic HTML (DHTML): una colección de tecnologías que están incluidas en casi todos los navegadores web para brindar soporte a la creación de sitios web interactivos y animados (Mikkonen y Taivalsaari, 2007).

2.8.3 PHP

PHP es un simple pero poderoso lenguaje diseñado para crear contenido HTML. PHP es un acrónimo recursivo que significa "*Hypertext Pre-Processor*", en español se le suele conocer como "*Lenguaje de Programación Interpretado*".

Una de las principales características de PHP es su amplio rango de soporte para bases de datos, puesto que cuenta con soporte para la mayoría de gestores de base de datos (entre ellos PostgreSQL, Oracle, MySQL, Sybase, Microsoft SQL, DB2 y bases de datos ODBC), entre muchas otras. Incluso algunas bases de datos recientes de estilo NoSQL como SQLite y MongoDB que de igual manera cuentan con soporte (Lerdorf, Tatroe y MacIntyre, 2006).

2.8.4 Ruby

Ruby es un lenguaje de programación de propósito general orientado a objetos, dinámico y reflectivo que combina sintaxis basada en lenguajes como Perl con características de Smalltalk, al igual que lenguajes como Eiffel y Lisp. Ruby implementa múltiples paradigmas de programación, incluyendo el paradigma funcional, orientado a objetos, imperativo y reflectivo (Matsumoto y Ishituka, 2002).

2.9 Frameworks para desarrollo web

De acuerdo a Samtani, y Sadhwani (2002) los framework (también conocidos como marcos de trabajo o simplemente marcos) de aplicación son un conjunto holístico de pautas y especificaciones que proveen plataformas, herramientas, y entornos de programación para abordar al diseño, integración, rendimiento, seguridad y confiabilidad de aplicaciones distribuidas y multi-niveladas Los marcos de aplicación incluyen los servicios de presentación, procesamiento por el lado del servidor, gestión de sesiones, marco de lógica de negocios, almacenamiento de datos de aplicación en caché, lógica de aplicación, persistencia, transacciones, seguridad y servicios de inicio de sesión para aplicaciones..

2.9.1 Django

Como indica Plekhanova, Django es un framework web "full-stack" de alto nivel que hace posible la construcción de aplicaciones web con relativamente pocas líneas de código, es sencillo, intuitivo, robusto y flexible, permitiendo al usuario diseñar soluciones sin altos costos generales de desarrollo.

Django se construyó usando Python, un lenguaje de programación el cual combina el rendimiento de distintos lenguajes de programación, tales como C/C++ y Java, con la facilidad y rápido desarrollo de lenguajes scripting, tales como Ruby y Visual Basic. Lo que permite a los usuarios crear aplicaciones que resuelvan una enorme cantidad de problemas (Forcier, Bissex y Chun 2008).

2.9.2 J2EE

J2EE es un conjunto de especificaciones, las cuales definen un estándar para desarrollo de aplicaciones empresariales multi-nivel con Java (ver Figura 2.13). La plataforma J2EE provee un framework completo para diseño, desarrollo, ensamblado y despliegue de aplicaciones Java

construidas en un modelo de aplicación distribuido en múltiples niveles. La especificación de J2EE define numerosas APIs (Application Programming Interfaces) de servicios y múltiples modelos de programación de aplicaciones para el desarrollo de aplicaciones integrándolas con sistemas empresariales (Samtani y Sadhwani, 2002).

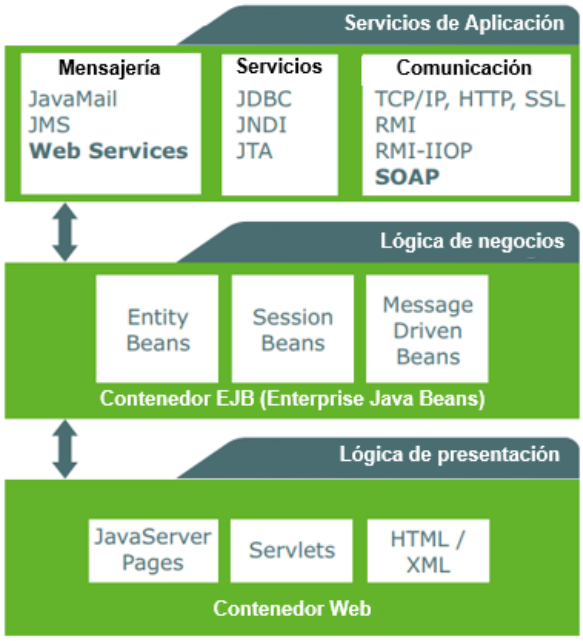

Figura 2.13: Niveles de J2EE.

2.9.3 Ruby on Rails

De igual manera que Django, Rails es un framework "full-stack" para desarrollo de aplicaciones web respaldadas en bases de datos, este framework sigue los principios de desarrollo web ágil, incrementando la productividad y acelerando el desarrollo (Plekhanova, 2009).

Ruby on Rails facilita el desarrollo, despliegue y mantenimiento de aplicaciones web, por otro lado, como su nombre lo indica, las aplicaciones de Rails se escriben en Ruby, un lenguaje de programación moderno, de scripting orientado a objetos (Ruby y Thomas, 2009).

2.9.4 Microsoft .NET

Microsoft .NET es una plataforma que forma que forma parte de los servidores, clientes y servicios. El framework .NET (ver Figura 2.14) incluye todo desde librerías básicas en tiempo de ejecución hasta librerías de interfaz de usuario – tiempo de ejecución de lenguaje común (CLR - Common Language Runtime), lenguajes como C#, C++, VB.NET, JScript.Net y las APIs del framework .NET (Samtani y Sadhwani, 2002).

Richter (2002) señala que el tiempo de ejecución de lenguaje común (CLR) es lo que su nombre indica: un tiempo de ejecución que se usa por diferentes y variados lenguajes de programación. Las características del CLR están disponibles para cualquiera y todos los lenguajes de programación en ese periodo objetivo.

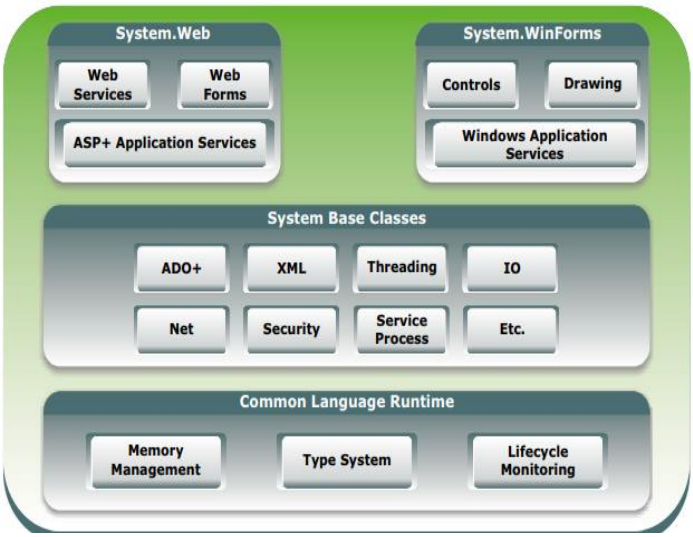

Figura 2.14: Componentes del framework .NET.

2.10 Trabajos relacionados

2.10.1 Odoo

También conocido como OpenERP y TinyERP, de acuerdo a su documentación oficial, Odoo es una suite de aplicaciones de gestión empresarial de código abierto producido por Odoo S.A. Se le considera a Odoo como una alternativa de código abierto a Microsoft Dynamics entre otras. Puesto que se enfoca a cubrir todos los procesos de una empresa, independientemente del giro al que pertenezca, por ejemplo: logística, distribución, producción, ventas, transporte, etc.

De igual manera, Odoo permite a las PMEs gestionar sus procesos de negocio de manera fiable, intuitiva, dinámica y escalable, de esta manera, no sólo las grandes empresas pueden mantener un orden de sus registros. (Odoo, 2018)

2.10.2 Aqua eBS

(Palazón, 2018) El ERP Aqua eBS en su edición 2018 representa una solución modular y flexible capaz de adaptarse en cada momento a las necesidades de los negocios. Es personalizable en su totalidad por medio de un entorno de desarrollo integrado que se encuentra disponible a través de las versiones: on premise, cloud e híbrido.

Entre sus características destacan una interfaz de seguridad, esto es, definir el nivel de acceso a la información para cada usuario de la solución, evitando de esta manera sanciones legales por un inadecuado uso del sistema.

2.10.3 Microsoft Dynamics 365

Palazón (2018) señala que Microsoft Dynamics 365 es la propuesta en la nube que permite elegir entre distintas aplicaciones modulares diseñadas para procesos, roles y sectores específicos, de esta manera se obtiene una mayor productividad. Para ello debe tomar datos obtenidos de Office 365 y LinkedIn para utilizarlos en herramientas como Outlook, Excel y Power Bi.

Microsoft Dynamics 365 Business Central es su más reciente implementación la cual consta de un panel de control todo en uno que permite a los negocios conectarlo todo, automatizar y optimizar operaciones.

2.10.4 Oracle ERP Cloud

Oracle ERP Cloud 2018 se diseñó para crecer con cada negocio, puesto que administra la contabilidad, planificación y análisis financiero, reconocimiento de ingresos, gestión de riesgos, cumplimiento, gobierno, adquisiciones, planificación de proyectos, informes fiscales, cierre financiero entre otras muchas funciones.

Oracle ERP Cloud permite la conexión con aplicaciones de nube pertenecientes a Oracle, ofreciendo así funcionalidad, análisis, seguridad, capacidades móviles y herramientas de colaboración social necesarias para administrar una empresa reduciendo costos operacionales y de capital (Palazón, 2018).

Capítulo 3: Desarrollo

3.1 Descripción de método

3.1.1 Análisis

Para la fase de análisis se planearon ejecutar diversas actividades, entre ellas, se planteó efectuar las actividades relacionadas con la ingeniería de requisitos con el fin de determinar las necesidades y requisitos para satisfacer en el sistema, simultáneamente llevar a cabo recopilación de información mediante el uso de diversos cuestionarios a los usuarios finales, por lo tanto, a partir de dichos cuestionarios sería posible determinar los requerimientos de la aplicación a desarrollar, para posteriormente llevar a cabo un estudio sobre las posibles alternativas que pudiesen plantar solución a las situaciones que se presentaron en la empresa Grupo Multiclimas.

3.1.2 Diseño

Posterior al análisis, y tras determinar una alternativa a desarrollar, las siguientes actividades contempladas a realizar se enfocarían en el diseño y maquetación de la base de datos y las principales interfaces de usuario. Por parte de la base de datos se pensó en diseñar una estructura concreta sobre los registros a almacenar, de igual manera, se planeó la manera de unificar la información con la que ya se disponía. Por otro lado, para el diseño de las interfaces de usuario sería necesario realizar diseños preliminares y bocetos sobre los controles a utilizar con su respectiva presentación visual.

3.1.3 Desarrollo

Durante la fase de desarrollo se realizó un plan de trabajo, en el cual, las primeras actividades estarían relacionadas con la creación de la base de datos, de esta manera sería posible proceder con la migración de la información, puesto que los registros con los que contaba la empresa Grupo Multiclimas se llevaban a mano, posteriormente, la programación de los formularios a implementar en el sistema, los cuales serían desarrollados en conjunto, es decir, desarrollar el front-end tomando como base los diseños y bocetos realizados en la etapa anterior para inmediatamente proceder con el desarrollo del back-end del mismo formulario haciendo uso del estilo arquitectural propuesto (ver Arquitectura en N capas (N-Layer) para aplicaciones).

Para llevar a cabo dicho desarrollo, se pensó en segmentar el proyecto en módulos, por lo tanto, el desarrollo del proyecto tuvo inicio con formularios que permitiesen el acceso al sistema, para posteriormente, continuar con el desarrollo de un módulo que se encargó de la administración del inventario, el cual permitiría llevar un ordenado control sobre los productos de la empresa. De igual forma, se necesitó realizar el desarrollo de un módulo de venta, el cual funcionaría simultáneamente con el módulo de inventario. Finalmente, un módulo importante a incluir en el sistema fue la gestión de clientes y proveedores, con el fin de agilizar distintos trámites.

3.1.4 Pruebas

Entre la etapa de desarrollo y la etapa de despliegue del proyecto, se consideró realizar una serie de pruebas al sistema, tanto en rendimiento como en exactitud y coherencia con la información manipulada.

3.1.5 Implementación

Tras completar la fase de pruebas, se llevó a cabo el despliegue del sistema, en el cual se necesitaba de un servidor el cual sería provisto por Grupo Multiclimas en sus oficinas centrales, para de este modo, concluir con el proyecto propuesto.

3.2 Procedimiento y descripción de las actividades realizadas

El proyecto se propuso por Sanes IT Consulting en respuesta a la solicitud enviada por parte de la empresa Grupo Multiclimas. De acuerdo a la metodología de desarrollo ágil Scrum (ver Figura 2.3), a continuación, se describen las fases de dicha metodología.

3.2.1 Análisis

Se realizó un análisis sobre las problemáticas en la empresa Grupo Multiclimas. Los resultados obtenidos se definieron como requerimientos del sistema (ver Requerimientos), dichos requerimientos se desglosaron en requerimientos funcionales (ver Requerimientos funcionales) y requerimientos no funcionales (ver Requerimientos no funcionales), los cuales se presentan en el capítulo de “Resultados y conclusiones”. En función de los resultados obtenidos, se realizó una investigación sobre los sistemas que satisficieran los requerimientos del sistema.

Se definió como óptima solución un sistema ERP, consecuentemente se optó por desarrollar el proyecto como una aplicación web, la cual, sólo tiene dos requisitos fundamentales, un

navegador web y conexión a Internet en el equipo de implementación. Por el contrario, no se consideró la opción desarrollar el proyecto como un sistema de escritorio tradicional, puesto que necesita de ciertas especificaciones técnicas por parte de los equipos donde se requiera su instalación.

Posteriormente, se analizaron los recursos y herramientas con los que se pretendió realizar el desarrollo del proyecto, en primer lugar, se estableció el lenguaje de programación C# mediante el framework .NET. Para el desarrollo de la base de datos se empleó el lenguaje SQL.

Las herramientas utilizadas fueron, en primer lugar, una computadora personal, en donde se instaló el software requerido para llevar a cabo el desarrollo del proyecto, por lo tanto se utilizó Visual Studio 2015 en su edición Community, por otro lado, se utilizó SQL Server 2017 en su edición Express para gestionar la base de datos del proyecto, puesto que ambas ediciones de las herramientas se pueden conseguir de manera gratuita desde los respectivos sitios oficiales de Microsoft, no se requirió efectuar pagos adicionales sobre licencias de software.

De igual manera, se necesitaron guías de desarrollo para un sistema ERP, por lo tanto, se procedió con un estudio y análisis de dichos sistemas, para lo cual, se buscaron distintas alternativas libres cuyas características fueran similares al sistema ERP que se desarrolló. La mejor opción que se encontró fue un sistema ERP desarrollado la empresa Odoo, dicho sistema posee el mismo nombre que su compañía desarrolladora. Dicho sistema fue un recurso muy importante puesto que fungió como guía para el desarrollo del presente proyecto.

Al mismo tiempo el acceso a Internet fue absolutamente necesario, puesto que, al tratarse de una aplicación web, se requieren de distintas librerías tanto de JavaScript y Bootstrap, para el diseño y funcionalidad del front-end. Durante el desarrollo del proyecto se necesitó de un disco duro externo para realizar respaldos diarios del proyecto y scripts de la base de datos.

Como parte de la fase de análisis de la metodología de desarrollo, se encuentra la planeación de trabajo y delimitación de actividades, así mismo, se delimitan los productos incrementales entregables.

Por lo mencionado anteriormente, se definieron los módulos a incluir en el sistema, adjunto a ellos, se incluyeron breves descripciones sobre las partes que conformaron a cada módulo, dichos módulos se diseñaron para satisfacer con las características de los requerimientos funcionales extraídos de la solicitud del cliente.

De igual manera, se planteó la opción de incluir aditamentos al proyecto que permitieran al sistema se amoldara al uso y necesidades de los usuarios finales, y no al revés, es decir, forzar al usuario que se adapte al sistema. Por lo tanto, se definió la pila de producto (*Product Backlog*) con los requerimientos priorizados y emergentes, así como los tiempos de entrega de los primeros productos.

Finalmente, se realizó un estudio y exploración del sistema ERP Odoo, del cual se extrajeron distintas características las cuales dieron paso a una visión más concreta sobre el desarrollo del proyecto y proporcionó estrategias para abordar las diversas actividades contempladas en las fases de la metodología restantes.

3.2.2 Diseño

Se necesitó desarrollar un administrador de contenido con el que se logró llevar un control ordenado para la construcción del proyecto, por lo cual, las fases de diseño y desarrollo se subdividieron en etapas (o sprints).

Los resultados obtenidos en la fase anterior, se depuraron con el fin de obtener características realmente relevantes para su posterior implementación en el sistema. Por otro lado, no se contaba con una estructura completamente definida para el front-end, por lo tanto, se requirió el diseño y construcción de los formularios que conforman el sistema.

Es importante mencionar que el sistema cuenta con un total de cincuenta y ocho formularios, uno de las cuales se encargó de funcionar como base para el resto de formularios, para ello se nombró como “Site.Master” (o sitio maestro), dicho formulario se construyó con un menú general de tipo DashBoard, menú que se desarrolló con la particularidad de modularidad, es decir, las opciones de dicho menú varían en función del rol de usuario que se encuentre haciendo uso del sistema.

Además de la implementación del menú modular, el objetivo principal del sitio maestro consistió en reducir los tiempos de carga de cada pantalla para acelerar el rendimiento del sistema, por lo tanto, el primer formulario en cargar es el sitio maestro, y, posteriormente, los componentes pertenecientes a cualquier pantalla a la que el usuario trate de acceder.

3.2.2.1 Selección y descripción de controles implementados en formularios

El Site Master se desarrolló en su mayoría con enlaces (o relaciones, conocidas como *links*) a todos los scripts de hojas de estilo en cascada, conocidos como CSS (*Cascading Style Sheets*), archivos de JavaScript los cuales se encargaron de agregar funciones al formulario por el lado del cliente y diseños de Bootstrap los cuales brindan una vista agradable al usuario.

Para el desarrollo del sitio maestro se utilizaron controles de lista desplegable ASP *(<asp:DropDownList>*), listas no ordenadas (*<ul>*), enlaces a otros formularios (*<a>*) y etiquetas tanto propias de ASP como de HTML (*<asp:Label>* y *<Label>* respectivamente). Muchos de estos elementos se desarrollaron anidados como elementos de listas (*<li>*) para las listas no ordenadas; de igual manera se utilizaron elementos *<asp:ListItem>* para las listas desplegables, también en contenedores de línea (**), los cuales, se crearon dentro de bloques de código delimitados con las etiquetas * * o *<section>*, divisiones que se encargaron de diferenciar bloques de código, para asignar distintos estilos y funcionalidades a los controles asignados dentro de cada bloque.

Algunos de los controles mencionados anteriormente se crearon en diferentes áreas de la interfaz, por lo tanto, se necesitaron elementos *<aside>* que permitieron representar contenido considerado, en algunos casos independiente, al cual el usuario pudiera consultar, dicho contenido se logró ubicar en distintas secciones de la interfaz.

Al mismo tiempo se hizo uso de elementos script de HTML (*<script>*) para inserción o referenciación de scripts ejecutables dentro del documento HTML del sitio maestro. Como ejemplo, la API de mapas de Google, la cual permitió mostrar las ubicaciones de las respectivas sucursales de la empresa Grupo Multiclimas.

Por otro lado, también se requirió del uso de elementos que permitiesen el uso de imágenes (<*img*>) anidados en bloques de código (<*figure*>) cuya funcionalidad es similar a las etiquetas <*div*>.

Simultáneamente, el sitio maestro se desarrolló con un manejo de excepciones, en caso de error, automáticamente se redirecciona al usuario a una página llamada *Default* con el fin de evitar caídas del sistema.

Prácticamente todos los controles que se mencionan con anterioridad, se utilizaron en el desarrollo de la mayoría de formularios, igualmente, el resto de formularios se crearon empleando botones tanto de HTML y ASP (<*Button*>, <*asp:Button*> y <*asp:LinkButton*>) con el fin de realizar confirmaciones o cancelar distintas acciones, así como el redireccionamiento a otros formularios. Además se utilizaron cajas y áreas de texto de ASP y HTML (<*asp:TextBox*> y <*TextArea*> respectivamente) para el ingreso de datos del usuario mediante el teclado.

Se implementaron selectores de fecha (*datepickers*) y controles de modos de vista (*GridView*), su función consiste en un modo de mostrar una lista (*ListView*) como si se tratase de una tabla. La clase GridView permitió la visualización de determinadas colecciones de información en formato de tablas, en las cuales se implementaron botones para poder interactuar con la información representada.

En cada formulario se hizo uso de cuadros de diálogo, en los cuales se mostraron distintos mensajes, desde operaciones exitosas hasta operaciones fallidas o advertencias.

Es importante mencionar que todos los controles cuentan con un identificador, y algunos de ellos tienen su funcionalidad del lado del cliente, como lo son los botones radiales (*RadioButton*), los cuales se desarrollaron con lenguaje scripting de JavaScript, el cual se incrustó en el código de la interfaz de usuario, por otro lado, el resto de controles que se implementaron realizan sus funciones predefinidas del lado del servidor, en el código presente detrás de cada interfaz de usuario (conocido como CodeBehind o back-end).

Mediante carpetas, en donde se almacenaron los formularios, los archivos CSS, JavaScript e imágenes, se logró llevar un correcto orden los distintos módulos que se desarrollaron para el sistema. De igual manera el proyecto se segmentó en una arquitectura en capas la cual permitió; además de un orden de archivos, una organización del código del proyecto.

Es importante mencionar que, en todas las interfaces de usuario del proyecto, se aplicaron las herramientas para el diseño de sitios y aplicaciones web de Bootstrap. De este modo se logró implementar controles con un diseño agradable para el usuario manteniendo un óptimo funcionamiento en los formularios.

3.2.2.2 Diseño de la base de datos

En la siguiente etapa de diseño del proyecto se contemplaron actividades relacionadas con la base de datos, la cual, como se mencionó anteriormente, se desarrolló en lenguaje SQL. Dicha base de datos se diseñó mediante el modelo entidad – relación, para ello se hizo uso del gestor de bases de datos SQL Server. El diseño de la base de datos se realizó en función de los resultados obtenidos en etapas anteriores, lo que permitió depurar la información de los registros de la empresa Grupo Multiclimas.

En total se diseñaron veintiocho tablas, y veintisiete llaves foráneas. Con el fin de evitar duplicidad e inconsistencia de la información.

3.2.2.3 Diseño front-end del módulo “Almacén”

En primer lugar, se desarrolló el front-end del formulario de productos, en el cual se hizo uso de imágenes, cajas de texto, botones y listas desplegables que permitieron al usuario elegir entre diversas opciones precargadas.

El diseño del formulario de proveedores se llevó acabo empleando etiquetas, listas desplegables y botones, además se implementaron botones radiales controlados mediante código JavaScript. Asimismo, se desarrolló un manejo de excepciones, para controlar errores del sistema.

En la construcción del front-end del submódulo de stock se hizo uso de etiquetas (Label), listas desplegables (asp:DropDownList), cajas de texto (asp:TextBox), botones (asp:LinkButton) y vistas en cuadrícula (asp:GridView), en la cual se implementaron distintas funcionalidades.

El submódulo de configuración de la información de almacenes se diseñó con seis formularios, de los cuales, uno de ellos se utilizó para un óptimo manejo de errores.

Se diseñó un formulario con el nombre de “WarehouseEdit” para realizar cambios en los registros de almacén, para dicho formulario se requirieron únicamente cajas de texto (asp:TextBox), etiquetas (Label) y botones (asp:Button).

El formulario “WarehousePedidos” se diseñó empleando controles de tipo caja de texto con su respectiva etiqueta y botón, de igual manera se implementó una tabla de control para las peticiones de producto por parte de los clientes a la empresa, dichas peticiones se enlistaron en espera, puesto que a la tabla se añadió un botón, el cual se diseñó con la funcionalidad de redireccionamiento a una nueva ventana llamada “Pedido”, en donde se logró gestionar completamente las peticiones de los clientes.

El siguiente formulario que se diseñó se llamó “Pedido”, en donde se utilizaron cajas de texto con sus respectivas etiquetas, además de botones (asp:LinkButton) y una tabla (GridView) en la cual se permitió visualizar información detallada de productos.

Se diseñó un formulario de proveedores en el cual se incluyeron controles de tipo botón para registrar proveedores y vistas en cuadrícula (GridView) para visualizar los datos del proveedor.

Para el formulario de stock, se incluyeron los mismos controles ASP de tipo botón, mediante el redireccionamiento a el formulario de stock se permitió agregar nuevos registros en la base de datos por medio de la implementación de procedimientos almacenados.

Conjuntamente se empleó una vista en cuadrícula, la cual permitió visualizar al usuario distintos datos del stock registrado a su almacén correspondiente.

3.2.2.4 Diseño front-end del módulo “Ventas”

En primer lugar, se diseñó el formulario de clientes, en el cual se emplearon botones radiales, los cuales se controlaron con código JavaScript ejecutable desde la interfaz de usuario.

De igual manera, se implementaron controles de ASP tales como cajas de texto y listas desplegables con sus respectivas etiquetas, además de botones para la efectuar diversas acciones, entre ellas cancelar o confirmar cualquier operación que se encuentre en proceso.

En el formulario “Ventas” se implementaron distintos controles, como los son cajas de texto y listas desplegables, cada control con su respectiva etiqueta. Además del uso de botones y vistas en cuadrícula para visualizar los productos agregados a la venta y sus respectivas características. De igual manera se agregó un botón que permite eliminar productos, previo a la confirmación de las ventas.

A la par se diseñaron más formularios enfocados a ventas de producto, se identifican con los nombres de “VentaTraslado” y “VentaEspecial”, ambos formularios se crearon con un diseño y componentes similares, tales como listas desplegables y cajas texto con sus respectivas etiquetas. También se hizo uso de vistas en cuadrícula con funciones adicionales, tales como la posibilidad de descartar productos en una orden de venta.

Los formularios que se mencionan anteriormente permiten la gestión de productos en stock, la diferencia que existe entre ambos radica en los diversos tipos de venta que se pueden efectuar.

Para la creación y modificación de los tipos mencionados, se diseñó un formulario con el nombre “TypeSale”, para el cual se emplearon únicamente cajas de texto, etiquetas y botones.

El siguiente formulario que se diseñó se utilizó para la gestión de garantías de los productos, dicho formulario se creó con listas desplegables, cajas de texto y áreas de texto con sus respectivas etiquetas, además de la implementación de distintos botones y vistas en cuadrículas.

Posteriormente, se diseñó el formulario de facturación de ventas, en el cual se implementaron listas desplegables y cajas de texto con sus respectivas etiquetas. Además, se incluyeron botones y vistas en cuadrícula para la visualización de información de una manera más clara y concreta. De igual manera se incluyó la opción de impresión de las facturas generadas. Cabe destacar que de dicho formulario sólo se desarrolló la parte correspondiente al front-end, puesto que no se contó con el back-end y desarrollo de procedimientos almacenados y métodos en las capas de negocio y acceso a datos.

En el desarrollo front-end del formulario de clientes se utilizaron botones y vistas en cuadrícula para visualizar información de los clientes pertenecientes a Grupo Multiclimas. Además, se agregaron se agregaron botones para la modificación y eliminación de los registros plasmados en las vistas en cuadrícula.

Se desarrollaron distintos formularios de ventas relacionados con pedidos; el primero de ellos se nombró como “SalesPedidos”, el cual permitió el almacenamiento, búsqueda, edición y eliminación de preventas, al mismo tiempo se añadió la opción de imprimir un ticket de las preventas con los datos correspondientes.

Los controles implementados fueron botones, caja de texto y etiquetas, además de vistas en cuadrícula para visualizar las distintas preventas existentes.

El siguiente formulario se nombró como “SalesPedidosEspeciales”, el cual se creó con botones, una caja de texto y una vista en cuadrícula, la cual, de igual manera que en el caso anterior, a la vista en cuadrícula se le añadió la funcionalidad de edición y eliminación de cada uno de los registros presentados al usuario, de igual manera, se incluyó la opción de imprimir el ticket de la venta especial.

Seguido al desarrollo del formulario de ventas especiales, se desarrolló el front-end de los formularios de ventas para solicitar producto a proveedores. Para ello se desarrollaron dos formularios que se nombraron como “SalesPedidosProveedor”, en donde se agregan los productos a solicitar al proveedor y “SalesPedidosProveedorView” en el cual se mostró de manera detallada los productos por adquirir, la función principal de dicho formulario fue la confirmación de la petición de producto al proveedor.

Para el desarrollo front-end del primer formulario se utilizaron listas desplegables, cajas de texto y áreas de texto con sus respectivas etiquetas. Igualmente, se incluyeron botones, imágenes y vistas en cuadrícula con la opción de eliminar registros por fila.

Finalmente, se requirió de un formulario para la verificación de los equipos en la solicitud de producto, para ello, previa la confirmación de pedido se implementó un redireccionamiento del usuario al formulario “SalesPedidosProveedor”, en el cual se llevó a cabo dicha verificación.

En dicho formulario se implementó un control de tipo vista en cuadrícula, en donde fue posible visualizar todos los pedidos y sus respectivos datos en función del almacén desde donde se opera el sistema. Asimismo, se incluyeron las opciones de cancelación, impresión y confirmación del pedido.

Seguido de los formularios de pedidos a proveedor se desarrollaron los formularios de pedidos a sucursales, esto permitió la venta de producto a los clientes independientemente de si la sucursal en donde se solicite el producto no cuente con el suficiente stock.

Ese tipo de pedidos se logró mediante el formulario "SalesPedidosSucursales", el cuál consta de dos secciones, creación y consulta de pedidos, en la primera sección se utilizaron distintos controles como botones, listas desplegables, cajas de texto, áreas de texto, etiquetas y vistas en cuadrícula en donde visualizar la información correspondiente, de igual manera se añadió la opción de descartar productos agregados por error.

La segunda sección del formulario consistió en la búsqueda de pedidos a la sucursal que emite la solicitud, en la estructura de la segunda sección se implementaron controles selectores de fecha, así como cajas de texto y botones.

Se hizo uso de controles de tipo vista en cuadrícula para la visualización de las solicitudes de producto, así como sus respectivos datos. En conjunto se agregó la opción de cancelar pedidos e imprimir los comprobantes de las diversas peticiones.

El módulo "Pre-VentasTraslado" se encargó de implementar el formulario "VentaTraslado", dicho módulo se diseñó con controles de tipo caja de texto, botones y vista en cuadrícula, asimismo, se incluyeron opciones de edición, eliminación e impresión de los tickets de venta. Para añadir nuevas preventas se añadió un botón el cual se encargó de redireccionar al usuario al formulario "Venta Traslado".

Sucesivamente, se desarrolló el formulario de venta de producto, en el cual se agregaron únicamente controles de tipo botón y vista en cuadrícula. Se implementó un redireccionamiento del usuario a un formulario que permitió llevar a cabo el registro de salidas de productos, para

ello, dicho formulario se nombró como “Salidas”, en el cual se gestionó de manera óptima el stock que se encontrase en la sucursal que efectúa dicha operación.

El diseño del formulario “Salidas” constó de controles de tipo cajas de texto, listas desplegables y botones cuyas funciones se enfocaron en agregar y registrar salidas, de igual manera se utilizó un control de tipo vista en cuadrícula para la visualización más detallada de los registros consultados.

El siguiente formulario se nombró “SalesProd” y se diseñó con controles de tipo botón para acceder al formulario “Salidas” y vistas en cuadrícula para una visualización a detalle de los registros.

Él último formulario que se desarrolló en el módulo de ventas se nombró como “SalesTypeVenta” en el cual se implementaron botones que permitieron el almacenamiento y actualización de registros mostrados en una vista en cuadrícula, a la que se añadieron las opciones de edición y eliminación.

El primer formulario que permitió la generación de reportes se nombró “SalesRepSales”, el cual se creó con diversos controles ASP, como los son cajas de texto, botones, etiquetas y la vista en cuadrícula; donde, en ésta última se cargaron los datos más relevantes al usuario.

Posteriormente se diseñaron formularios que incluyeron la herramienta de generar reportes fueron los formularios “SalesRepProd” y “SalesRepProdCounting”, ambos fueron diseñados con controles de tipo lista desplegable, botones y vistas en cuadrícula. La diferencia entre ambos formularios radica que, en el primer caso, se permitió generar reportes de los productos en stock, por otro lado, en el segundo formulario se permitió reportar el producto contabilizado monetariamente.

Previo a la finalización del módulo de ventas se desarrolló el formulario de inicio de sesión, los controles implementados en dicho formulario fueron cajas de texto, listas desplegables y botones.

3.2.2.5 Diseño de formularios para normalización del sistema

La normalización del sistema consistió en un módulo de administración de las cuentas de usuario. A continuación, se describen los formularios que se consideraron para llevar a cabo la normalización del sistema.

Se diseñó un formulario para el registro de usuarios, el cual se diseñó con cajas texto, botones y etiquetas.

Se diseñó un formulario de confirmación de cuenta, en caso de registrar a un usuario con un correo electrónico válido, para ello, se requirieron únicamente etiquetas y un enlace a un inicio de sesión alternativo distinto al que se menciona anteriormente.

El formulario de inicio de sesión alternativo, se compuso de cajas de texto, etiquetas y botones, además de incluir distintas funciones, como la recuperación de contraseñas y el registro de nuevos usuarios.

También, se diseñó un módulo de autenticación en dos factores (también conocida como autenticación en dos pasos) para el cual se requirieron controles de tipo lista desplegable, caja de texto y botones.

En caso de presentarse un inicio de sesión con una cuenta bloqueada, se agregó el redireccionamiento a un formulario con el nombre de “Lockout”, en el cual se mostró un mensaje al usuario sobre el bloqueo de su cuenta.

En caso de un inicio de sesión exitoso se adicionó el redireccionamiento del usuario al formulario de administración de la cuenta, en donde se incluyeron diversas opciones, como, por ejemplo, modificar la contraseña, al mismo tiempo se añadió la opción de agregar los servicios de autenticación de dos factores mencionados con anterioridad.

Se diseñó un formulario dedicado a la visualización de los inicios de sesión de usuario. Dicho formulario constó de únicamente una lista con los inicios de sesión registrados, en la cual, a cada registro se incluyó la opción de eliminar la sesión.

Se diseñó un formulario con el nombre “ManagePassword”, el cual se pensó implementar una doble funcionalidad, tales como, establecer una contraseña para el registro de nuevos usuarios, para la cual se utilizaron cajas de texto, botones y etiquetas.

La segunda funcionalidad considerada consiste en la modificación de contraseña para usuarios registrados, para ello se utilizaron cajas de texto, botones y etiquetas, es importante mencionar que las funcionalidades mencionadas se presentaron en distintos subformularios, es decir, en función del formulario de acceso se distingue la opción de administración de contraseña mostrada.

Se adicionó un formulario que permitió la gestión de números de teléfono de los usuarios, dicho formulario se construyó únicamente con cajas de texto, botones y etiquetas. En el cual al ingresar un número de teléfono válido se redirecciona al usuario a un formulario de verificación del número de teléfono por medio de un código que se recibe por SMS.

Como se mencionó anteriormente, en el formulario de inicio de sesión alternativo, se incluyó la opción de recuperar la contraseña en caso de olvidarla. Para ello se utilizaron controles de tipo caja de texto, etiquetas y botones, los cuales se diseñaron para solicitar al usuario su correo electrónico para enviar un enlace de recuperación de contraseña.

Por otro lado, se requirió diseñar formularios para facturación, por lo tanto, se necesitó llevar a cabo una investigación acerca del tema en base a las últimas normativas propuestas. De igual manera se realizaron diseños preliminares sobre el formulario de facturación de pedidos.

Simultáneamente se diseñaron los wireframes del módulo de pedidos para su implementación en el módulo de punto de ventas. Tanto los formularios de facturación como éste último de pedidos se diseñaron para ser implementados en el módulo de punto de ventas.

Dicho módulo se diseñó con controles ASP como botones, listas, imágenes, cajas de texto, y vista en cuadrícula en donde se visualizaron los productos en stock para su comercialización, así como los datos más relevantes para el usuario.

El módulo de punto de venta se desarrolló en un único formulario, el cual se diseñó para agregar los productos por medio de su código de barras, es posible escanear o introducir manualmente el código.

3.2.3 Desarrollo de módulos

En primer lugar, se realizó la migración de los registros de la empresa Grupo Multiclimas a la base de datos. Para ello todos los registros se capturaron manualmente en la base de datos, puesto que dichos registros se encontraban descritos en papel.

Con el fin de efectuar un desarrollo ordenado del proyecto, se creó un plan de trabajo, el cual consistió en dividir el proyecto en tres grandes módulos, y, a su vez, subdividir cada módulo en tareas más pequeñas.

El módulo de "Almacén", el cual es el primero, consistió de cuatro submódulos, los cuales son: productos, proveedores, stock, y una pantalla de configuración de la información de almacén.

3.2.3.1 Acoplamiento de base de datos al proyecto

En primera instancia, se necesitó integrar la base de datos al proyecto, para ello se creó una capa dentro del proyecto la cual se enfocó en la gestión de la información de la base de datos, dicha capa se nombró como "POCO" (también se le conoce como capa de "Entidades"). Dicha integración de la base de datos al proyecto se logró mediante una herramienta del framework .NET llamada "ADO.NET Entity Data Model", herramienta encargada de crear modelo de código basado en una base de datos existente, para este caso, la base de datos mencionada anteriormente.

Para lograrlo, se creó una conexión desde el proyecto en Visual Studio a la base de datos en SQL Server, posteriormente, las tablas de la base de datos se añadieron al proyecto como clases, y, las columnas de cada tabla se añadieron como atributos de clase, a los cuales se pudo acceder desde cualquier otra capa del proyecto.

Seguidamente a la creación de la capa de Entidades (llamada POCO en este caso) se crearon las capas de Negocio y la de acceso a datos (DAL).

Posteriormente, se crearon los procedimientos almacenados a implementar en el proyecto, básicamente se crearon CRUDs (operaciones básicas de bases de datos, creación, consultas, actualización y eliminación) para el manejo de la información almacenada en la base de datos.

En la capa de acceso a datos se crearon clases de tipo Beans, las cuales permiten la conexión a la base de datos. Todas las clases del proyecto que se encuentran en la capa de acceso a datos hacen uso de los beans de conexión para la manipulación de la información. Por lo tanto, el flujo de información se logra por medio de objetos de conexión (a la base de datos) creados por las clases de la capa de acceso a datos.

3.2.3.2 Desarrollo del módulo "Almacén"

Se procedió con la codificación del formulario de productos, en el cual se utilizaron los métodos correspondientes en la capa de negocios, para ello se hizo uso de clases de tipo interfaz (también llamadas protocolo), el cual se definieron métodos necesarios a utilizar por el usuario final.

La clase de productos en la capa de acceso a datos se desarrollaron los métodos requeridos para hacer uso de los procedimientos almacenados creados previamente.

Se desarrollaron procedimientos almacenados para el submódulo de proveedores, además del CRUD que se desarrolla normalmente, se crearon procedimientos que comparten información con la tabla de productos, por ejemplo, algunos procedimientos devuelven productos proporcionados por un proveedor determinado.

En el submódulo de proveedores, se creó en su capa de negocios una interfaz en donde se implementaron distintos métodos que permitieron el correcto consumo de procedimientos almacenados creados con anterioridad.

Los procedimientos almacenados desarrollados previamente para el submódulo de proveedores se implementaron en la capa DAL (capa de acceso de datos), la información almacenada en la base de datos se captura desde la interfaz de usuario, dicha información se valida en la capa de Negocios con el fin de evitar errores en el sistema.

Los procedimientos almacenados desarrollados para el submódulo de stock, se encargaron de manipular los registros de mercancías y productos almacenados para su comercialización.

Mediante clases que se desarrollaron en la capa de negocio para el submódulo de stock, se logró implementar los procedimientos almacenados que se crearon previamente (Creación, Lectura y Eliminación). En la clase de stock dentro de la capa de negocios se crearon las interfaces que se usaron para enviar los datos de la capa de vistas a la capa de acceso a datos. Los distintos métodos implementados devuelven como resultados distintos valores, en función de lo que requiera realizar el usuario final.

Como se mencionó anteriormente, la información a guardar se captura desde la capa de la vista, la información se almacena en los objetos de la capa POCO, y, en la capa de acceso a datos, se crean objetos de conexión para registrar los valores a la base de datos.

Para llevar a cabo el desarrollo del formulario "WarehousePedidos", se desarrollaron y probaron procedimientos almacenados que hicieron uso de distintas tablas. En la capa de negocio del formulario "Pedido", se hizo uso de la interfaz de stock, mencionada anteriormente.

"Pedido" es el submódulo con la mayor cantidad de clases dentro de las capas de negocio y de acceso a datos, puesto que se subdividió en pedidos para sucursales por parte de los clientes, y pedidos a los proveedores de producto para las sucursales.

Para la clase de pedidos para sucursales se desarrolló una interfaz, en la cual se añadieron los métodos que permitieron la creación, modificación y eliminación de registros, además de distintos tipos de consultas con filtrados de búsqueda.

Se nombró a una clase como "PedidoSuc" en la capa de acceso a datos, en dicha clase se utilizaron distintos métodos que implementaron los procedimientos almacenados correspondientes, los cuales, en base a su rol dentro de lo que representa un CRUD devuelven distintos valores.

Los procedimientos almacenados de consulta de información se desarrollaron con el objetivo de devolver tablas en memoria representados como DataTables y visualizados por los controles tipo vista en cuadrícula de ASP implementados en la interfaz de usuario.

En el desarrollo de la interfaz en la capa de negocios para la clase de pedidos a proveedores, además de los métodos que permitieron la creación, modificación y eliminación de registros, se

desarrollaron métodos de consulta de información, los cuales permitieron mostrar un total de pedidos a proveedores mediante los controles de tipo vista en cuadrícula.

En la clase de pedidos a proveedores se incluyó una opción que permitió operar con los datos del proveedor. Para ello, se hizo uso de CRUDs desarrollados previamente, los cuales se implementan en la capa de acceso a datos por medio de interfaces creadas en la capa de negocio del proyecto.

De igual manera, se añadió la posibilidad de registrar detalles de nuevos pedidos y conjuntamente, se lograron realizar modificaciones a distintos detalles de pedidos ya existentes a los proveedores.

Es importante mencionar que se implementaron distintas validaciones en cada una de las clases del módulo de almacén, dichas validaciones se encargaron de evitar errores al ingresar datos incorrectos o con un formato distinto definido, por ejemplo, se puede presentar el caso de ingresar caracteres de texto en campos donde se solicitan valores numéricos, o incluso, dejar campos vacíos.

3.2.3.3 Desarrollo del módulo “Ventas”

En el formulario de clientes se implementaron las funcionalidades de inserción de nuevos clientes y la modificación de los clientes que ya se encontraran registrados. Para lograr una modificación exitosa de los datos de algún cliente se requirió obtener sus datos, mediante una consulta a la base de datos tomando como parámetro el número de identificación de cliente.

Para lograr la completa funcionalidad del módulo se creó una clase de tipo interfaz en la capa de negocios en la cual se crearon los respectivos métodos a utilizar en el formulario “Cliente”, lo cuales permitieron la inserción, actualización (los cuales devuelven un valor que va entre verdadero y falso) y selección de un único cliente, dicha selección permitió modificar los datos del cliente obtenido.

Dichos métodos de la capa de negocios se implementaron en la capa de acceso a datos, donde se generó una comunicación del proyecto a la base de datos por medio del objeto de conexión, los parámetros que se recibieron en cada método de esta capa son objetos de la capa POCO, de

los cuales se extrajeron únicamente los atributos necesarios en función de método que se encontrase haciendo uso de dichos objetos.

El correcto funcionamiento del módulo de clientes se logró por medio de la implementación de procedimientos almacenados, los cuales se crearon previamente al desarrollo de las distintas clases pertenecientes a las capas de la arquitectura del proyecto.

En la capa de negocio se creó una interfaz llamada “IVenta” para su respectivo formulario, en la cual se crearon distintos métodos que permitieron operar con los registros en la base de datos, dichos métodos permitieron la creación, consulta, modificación y eliminación de registros en la base de datos. Se pudo acceder a estos métodos desde distintas vistas en la capa de presentación, de igual manera, se crearon varios métodos de consulta de registros a la base de datos. Puesto que se requirió filtrar las búsquedas de ventas con el fin de obtener lo registros deseados de manera ordenada.

En la capa de acceso a datos se crearon distintos métodos los cuales recibieron como parámetro un objeto de tipo venta de la capa POCO, es en la capa de acceso a datos donde se agregaron todos los métodos que se utilizaron para el módulo de ventas, simultáneamente, se añadieron nuevos métodos a las interfaces en la capa de negocios y a las clases en la capa de acceso a datos de las clases de producto, stock, entre otras. Dado que algunas ventas influyeron en los registros de otras tablas, por lo tanto, se necesitó implementar nuevos procedimientos almacenados y métodos a los que se logró acceder desde el módulo de ventas.

Para el óptimo funcionamiento del formulario del módulo de ventas se crearon distintos procedimientos almacenados encargados de operar con los registros almacenados en la base de datos, a su vez, permitieron el filtrado de información para los diversos fines que se incluyeron en el proyecto.

Las vistas en cuadrícula implementadas en los formularios “VentaTraslado” y “VentaEspecial”, tuvieron como objetivo permitir la visualización de datos de tablas en memoria que se obtuvieron por medio de consultas a la base de datos que devolvieron los productos que se encuentren en la orden de dicha venta, la implementación de estas consultas se realizó por medio de distintos métodos desarrollados en la capa de acceso a datos mencionada anteriormente.

La programación del formulario “TypeSale” permitió el envío de información ingresada en sus controles a la base de datos por medio de métodos creados en una interfaz llamada “ITipoVenta” que se desarrolló en la capa de negocios, tales métodos se implementaron en una clase que se nombró como “TipoVenta” en la capa de acceso de datos. La información enviada del formulario a la base de datos fue de tipo objeto, perteneciente a la clase POCO. Los métodos en la clase TipoVenta se encargaron de tomar la información almacenada en los objetos y posteriormente se consumió mediante la implementación de los procedimientos almacenados mencionados anteriormente.

La programación en el back-end del formulario “Sales” permitió que las vistas en cuadrículas implementadas obtuvieran información mediante métodos en las capas de negocio y acceso a datos, que implementaron objetos de la capa POCO, dichos métodos realizaron la ejecución de los procedimientos almacenados creados para el módulo de garantías.

Al mismo tiempo, se implementó un filtrado de garantías, que facilitó la búsqueda de registros en la base de datos, dicho filtrado se basó en la información del folio, cliente, día y por técnicos certificados. Finalmente se añadió la opción de imprimir la información de la garantía como un documento, lo que permitió la facilidad de proporcionar garantías a los clientes.

Los métodos del CRUD implementados en el formulario en “SalesPedidos”, se implementaron en las clases de “Ventas” en las distintas capas de desarrollo de proyecto (DAL, Business). Cada una de las opciones desarrolladas para interactuar con la base de datos se encargaron de redireccionar al usuario al formulario “Ventas” detallado anteriormente.

El formulario “SalesPedidosEspeciales” se desarrolló con la intención de gestionar únicamente las ventas, para ello se incluyó la opción de almacenamiento y modificación de ventas especiales la redirección del usuario al formulario “VentasEspeciales”, en donde fue posible gestionar los productos que se encuentren en la orden de dicha venta.

En el formulario “SalesPedidosProveedor” se incluyó la funcionalidad de enviar el pedido al proveedor mediante un correo electrónico enviado automáticamente. Para ello se creó una clase llamada “FormatToPDF”, con la cual, empleando el uso de las librerías de terceros “iTextSharp” se logró convertir el contenido del formulario en un archivo de extensión PDF, esta

funcionalidad se implementó en un método que permitió guardar el contenido del pedido. Asimismo, se agregó la opción de imprimir los datos del pedido a realizar al vendedor.

En el formulario de confirmación del pedido (SalesPedidosProveedor) se cargan nuevos métodos desarrollados en la clase "PedidoProveedor" en la capa de acceso de datos (DAL), dicha clase importaron librerías propias de Microsoft, como los son *Mail* y servicios Exchange.

Por medio de dichas librerías fue posible llevar a cabo el envío de correos electrónicos al proveedor haciendo uso del servidor de correo con el que ya contaba la empresa Grupo Multiclimas, los parámetros requeridos por los métodos para enviar un correo electrónico, fueron un encabezado, el cual se creó automáticamente en la clase creada en la capa de acceso datos, el cuerpo del correo, que constó de la lista de productos creada en el formulario "SalesPedidosProveedor", la cual se incluye como tipo de dato HTML o de tipo string (cadena de texto), éste último tipo de dato se utilizó para la cancelación de pedidos, opción implementada en el formulario "SalesPedidosProveedorView" y un destino, el cual se obtiene de objetos de la capa POCO, puesto que los datos del proveedor se encuentran registrados en la base de datos, entre ellos su dirección de correo electrónico.

Para efectuar el envío de correos correctamente, se creó un método con el nombre de "SendEmail", el cual hizo uso de las librerías de Exchange, dicho método utilizó el cuerpo del correo con el tipo de dato HTML y se utilizó para enviar la lista de productos solicitados al proveedor. Por otro lado, el método "EnviarEmail" permitió el envío de distintos mensajes, es decir, si el correo se envió desde el formulario "SalesPedidosProveedor" el mensaje creado trata sobre una petición de equipos al proveedor, en caso contrario, si el correo se envió desde el formulario "SalesPedidosProveedorView" el mensaje creado trata sobre la cancelación de un pedido.

Los controles pertenecientes al formulario SalesPedidosSucursales permitieron al usuario elegir la sucursal en donde se encontrase el producto deseado, además del tipo de producto y cantidad. A su vez, se incluyó la opción de dejar una descripción o comentario adicionales, así como se agregó la opción de imprimir la orden de pedido.

El desarrollo de la segunda sección del formulario permitió la búsqueda de pedidos mediante un filtrado de datos, el cual constó tanto de fechas obtenidas de los controles correspondientes en el formulario, como de folio de pedido, el cual se pudo agregar por los controles designados para dicho filtro de búsqueda. Los pedidos obtenidos se almacenaron en una tabla en memoria la cual se pudo visualizar mediante la vista en cuadrícula implementada en el formulario.

En el formulario "Pre-VentasTraslado" se implementaron distintas funciones, tales como la edición de las preventas generadas, para ello se hizo uso de un redireccionamiento del usuario al formulario "Ventas" para completar tal acción, por otro lado, se incluyó la opción de eliminación, en la que se implementaron distintos métodos que se desarrollaron en la clase "Preventa" en la capa de acceso a datos, dichos métodos hicieron uso de los procedimientos almacenados correspondientes desarrollados anteriormente.

Para el desarrollo de formularios de venta de productos, se hizo uso de métodos desarrollados en las clases de stock, tanto de la capa de negocios, como de la capa de acceso de datos, por lo tanto, las funciones que se incluyeron en los formularios de ventas consistieron en extraer registros desde la base de datos, tales registros se seleccionan desde la interfaz de usuario, asimismo, también se incluyó la opción de seleccionarlos por medio de su código de identificación o del almacén desde el cual se lleva a cabo la venta; la información que se extrae de la base de datos se presentan en controles de tipo vista en cuadrícula implementados en la interfaz de usuario. Por otro lado, se encuentra la opción de reducir la cantidad de stock en la base de datos para cada producto seleccionado. De igual manera que en otros formularios se incluyó la opción de imprimir un reporte con la información de stock que se registró como salida de producto.

Como opciones complementarias se incluyeron las opciones de almacenamiento y actualización de registros, las cuales se llevaron a cabo mediante el redireccionamiento del usuario al formulario "TypeSale" el cual se detalló anteriormente.

De igual manera, se incluyeron las opciones de eliminación y consulta que se implementaron en el formulario SalesTypeVenta se agregaron en el código back-end de dicho formulario, tales métodos hicieron uso de los métodos desarrollados en la interfaz "ITipoVenta" mencionada

anteriormente, y, a su vez, dichos métodos se consumieron en la capa de acceso a datos, en donde se efectuó la comunicación con la base de datos.

3.2.3.4 Desarrollo de formularios de gestión del sistema

Como herramientas adicionales, se desarrollaron módulos que permitieron al cliente generar reportes, tanto en formato PDF como en hojas de cálculo. Para ello se crearon distintos formularios que se encargaron de obtener información tanto de módulos de ventas como de productos, y de esta manera, se logró exportar la información a los formatos mencionados anteriormente.

Los controles implementados en el formulario "SalesRepSales" permitieron filtrar información por medio de ventas ordenadas por mes, día, semana y/o usuario, la información obtenida por dicho filtrado se logró visualizar en controles de tipo vista en cuadrícula, donde, una vez cargada, se pudo procesar dicha información por medio de una librería de terceros llamada "ClosedXML" y posteriormente, se puede exportar como una hoja de cálculo, la cual se puede visualizar y editar con software de ofimática como Excel. De igual forma, la opción de impresión permitió que la información cargada en las vistas en cuadrícula se pueda exportar como formato PDF o simplemente, imprimir la información sin necesidad de conservar el archivo.

El formulario "SalesRepProd" permitió al usuario realizar una búsqueda de producto en función del almacén al que se encuentre asignado el usuario que realice la búsqueda, en el caso del segundo formulario, la información se filtró por producto y, al igual que el formulario anterior se utilizó almacén como filtro de búsqueda, los resultados obtenidos se almacenaron en una tabla en memoria que se pudo visualizar por medio de vistas en cuadrícula.

Una vez con datos, se pudo exportar y/o imprimir un reporte de los productos que se cargaron en su respectiva vista en cuadrícula, por medio de la librería "ClosedXML", tanto en la hoja de cálculo como en el documento PDF.

El formulario de inicio de sesión se desarrolló con métodos que permitieron obtener una lista de almacenes, la cual es posible visualizar por medio de controles de tipo lista desplegable.

El formulario de inicio de sesión se diseñó con la funcionalidad de extraer los datos del usuario que accede al sistema, los datos que se consiguen se envían como un objeto de tipo “usuario” perteneciente a la capa POCO a la capa de acceso de datos por medio de una interfaz en la que se implementó un método llamado “GetLogin”, los datos que se extraen se pueden utilizar en los formularios a los que el usuario tenga permiso de acceder en función de su rol en el sistema.

Se desarrollaron distintos métodos a implementar en el módulo de punto de venta, dichos métodos se desarrollaron con la finalidad de enviar la información por las distintas capas de la arquitectura del proyecto para los formularios de facturación y punto de ventas, los cuales se desarrollaron en la interfaz y la clase “Venta”, dichos métodos hicieron uso de los procedimientos almacenados que se desarrollaron especialmente para el módulo de punto de venta.

Al mismo tiempo, el formulario se diseñó para tomar automáticamente el número de identificación de usuario, lo que permitió llevar un control de los registros de venta ordenado, puesto que se guardan tanto los datos del usuario quien efectúa la venta, como los datos del cliente.

3.2.4 Fase de pruebas

Tras finalizar el desarrollo de cada procedimiento almacenado, métodos y formularios se realizaron distintas pruebas, en las que se evalúo tanto su correcto funcionamiento como su rendimiento, por lo tanto, se realizaban las correcciones y optimizaciones pertinentes a la brevedad posible.

Cabe destacar que ningún desarrollo se inició sin antes haber completado satisfactoriamente el desarrollo anterior, es decir, los errores y contratiempos que se presentaron se resolvieron antes de continuar con el desarrollo de los siguientes módulos. De esta manera fue posible reducir el periodo de pruebas y depuración optimizando los tiempos en su mayoría para el desarrollo del proyecto, todos.

No obstante, se delimitó un periodo de tiempo en el que se realizaron distintas pruebas en el sistema empleando la información recopilada de la base de datos, durante dichas pruebas se

encontraron diversas irregularidades, las cuales se depuraron y optimizaron de manera inmediata.

De igual manera, se realizaron diversas correcciones y ajustes para obtener un óptimo rendimiento del sistema, para, además de reducir tiempos de espera, obtener información concisa y de utilidad para el usuario final.

3.2.5 Implementación

La implementación del sistema se pensó llevar a cabo en dos fases, en primera instancia, se pensó en instalar la base de datos en el Data Center de la empresa Grupo Multiclimas, por otro lado, la instalación del sistema se planeó realizar en un servidor central, determinado por el cliente.

Es importante mencionar, que la idea de llevar a cabo la instalación del sistema en un servidor central fue para lograr la correcta unificación de la información registrada por las distintas sucursales de la empresa.

Capítulo 4: Resultados y conclusiones

4.1 Requerimientos (R)

De acuerdo a Sommerville, los requerimientos para un sistema son las descripciones sobre lo que el sistema debería realizar— los servicios que provee y las restricciones en su operación. Estos requerimientos reflejan las necesidades del cliente para un sistema que satisface ciertos propósitos como controlar un dispositivo, ordenar o encontrar información. El proceso de hallar, analizar, documentar y revisar estos servicios y restricciones se le conoce como ingeniería de requerimientos, de igual manera, indica que el término "requerimiento" no se usa frecuentemente en la industria del software (2007).

Al mismo tiempo, Sommerville (2005) menciona que, en algunos casos, un requerimiento es simplemente una declaración abstracta de alto nivel sobre un servicio que un sistema debería proporcionar o la restricción presente en un sistema.

De igual manera se presentan distintas definiciones de "Requerimiento" existentes en el glosario de la IEEE:

1) Una condición o capacidad necesaria por un usuario para solucionar un problema o alcanzar un objetivo (Std 610.12-1900, IEEE).

2) Una condición o capacidad que debe ser cumplida o poseída por un sistema o el componente de un sistema para satisfacer un contrato, estándar, especificación u otros documentos impuestos formalmente (Std 610.12-1900, IEEE).

Por lo tanto, a continuación, se presentan los requerimientos del sistema que se lograron obtener del análisis realizado, previamente detallado en la sección de Análisis en el capítulo de "Desarrollo". A continuación, se presentan los requerimientos (especificados con la abreviatura **R**) obtenidos, de los cuales se desglosan los distintos requerimientos funcionales (especificados con la abreviatura **RF**) que permitieron llevar acabo el desarrollo del proyecto en un correcto orden, permitiendo de esta manera su fácil escalabilidad y detección de errores, y, finalmente, se presentan los requerimientos no funcionales (especificados con la abreviatura **RNF**) que delimitan ciertos aspectos del sistema.

Requerimiento 1: Acceso al sistema.

R-1	**Acceso al sistema**
Requerimientos asociados	RF-1: Inicio de sesión
Descripción	Debe ser posible registrar las credenciales en el sistema para su posterior uso en los formularios correspondientes.
Datos específicos	▪ Nombre de usuario ▪ Contraseña

El Requerimiento 1 contiene las actividades y datos necesarios para cumplir la función que permite el acceso al sistema. Asimismo, se presentan los requerimientos funcionales que lo conforman.

Requerimiento 2: Catálogo de productos.

R-2	**Catálogo de productos**
Requerimientos asociados	▪ RF-2: Nuevo producto ▪ RF-3: Buscar producto ▪ RF-4: Modificar producto ▪ RF-5: Eliminar producto
Descripción	Debe ser posible manipular los registros correspondientes de cada producto.
Datos específicos	▪ ID ▪ Nombre ▪ Marca ▪ Tipo ▪ Código de barras ▪ Precio de compra ▪ Imagen ▪ ID proveedor ▪ Descripción ▪ Clave de unidad

En el Requerimiento 2 se describen los requerimientos funcionales que representan las actividades a desarrollar, así como los datos necesarios para lograr llevar a cabo una óptima gestión de productos.

Requerimiento 3: Catálogo de proveedores.

R-3	**Catálogo de proveedores**
Requerimientos asociados	▪ RF-6: Nuevo proveedor ▪ RF-7: Buscar proveedor ▪ RF-8: Modificar proveedor ▪ RF-9: Eliminar proveedor
Descripción	Debe ser posible manipular los registros correspondientes de cada proveedor.
Datos específicos	▪ ID ▪ Nombre ▪ Apellido ▪ Correo ▪ Dirección ▪ Estado ▪ Código postal ▪ Teléfono ▪ RFC ▪ ID de municipio ▪ ID de almacén

En el Requerimiento 3 se describen las actividades que contemplan la gestión de proveedores, así como la información correspondiente de cada uno que permitió mantener un orden de los proveedores de la empresa.

Requerimiento 4: Catálogo de stock.

R-4	**Catálogo de stock**
Requerimientos asociados	▪ RF-10: Nuevo stock ▪ RF-11: Buscar stock
Descripción	Debe ser posible manipular los registros correspondientes de cada producto en stock.
Datos específicos	▪ ID ▪ ID de almacén ▪ ID de producto ▪ Cantidad ▪ Fecha

En el Requerimiento 4 se especifican los requerimientos funcionales correspondientes al control de stock, es importante mencionar que algunos de los datos especificados se relacionan con los datos de otros requerimientos.

Requerimiento 5: Catálogo de clientes.

R-5	Catálogo de clientes
Requerimientos asociados	▪ RF-12: Nuevo cliente ▪ RF-13: Buscar cliente ▪ RF-14: Modificar cliente ▪ RF-15: Eliminar cliente
Descripción	Debe ser posible manipular los registros correspondientes de cada cliente.
Datos específicos	▪ ID ▪ Nombre ▪ Apellido ▪ Correo ▪ Dirección ▪ Estado ▪ Código postal ▪ Teléfono ▪ RFC ▪ ID de municipio

Con el fin de agilizar distintos procesos como los de ventas, se obtuvo el Requerimiento 5, en el cual se especifican los requerimientos funcionales y datos necesarios para llevar a cabo el óptimo control de los datos personales de cada cliente de la empresa Grupo Multiclimas.

Requerimiento 6: Catálogo de preventas.

R-6	Catálogo de preventas
Requerimientos asociados	▪ RF-16: Nueva preventa ▪ RF-17: Buscar preventa ▪ RF-18: Modificar preventa ▪ RF-19: Eliminar preventa
Descripción	Debe ser posible manipular los registros correspondientes de cada venta.
Datos específicos	▪ ID ▪ ID de cliente ▪ Total ▪ Fecha ▪ ID de tipo de venta ▪ ID de almacén ▪ Surtido ▪ Facturado ▪ Pedido de usuario ▪ Usuario de almacén ▪ Forma de pago ▪ IVA ▪ Total

Se obtuvo el Requerimiento 6, el cual se enfocó en la gestión de preventas de producto, de igual manera, se especifican los datos necesarios para un ordenado control de las ventas, es importante mencionar que los datos especificados se utilizan para la generación de facturas.

Requerimiento 7: Catálogo de garantías.

R-7	**Catálogo de garantías**
Requerimientos asociados	▪ RF-20: Nueva garantía ▪ RF-21: Buscar garantía
Descripción	Debe ser posible manipular los registros correspondientes de cada garantía.
Datos específicos	▪ ID ▪ ID de técnico ▪ ID de venta ▪ ID de producto ▪ ID de proveedor ▪ Fecha de venta ▪ Fecha de cambio ▪ ID de usuario ▪ Folio ▪ Descripción ▪ Almacén

En el Requerimiento 7 se detallan las actividades y campos necesarios para hacer válidas las garantías de producto, dado que se pueden presentar casos en donde sea necesario reparar o incluso cambiar un equipo que no se encuentre en condiciones estables para operar.

Requerimiento 8: Salida del sistema

R-8	**Salida del sistema**
Requerimientos asociados	▪ RF-22: Cerrar sesión ▪ RF-23: Salir del sistema
Descripción	Debe ser posible cerrar la sesión del usuario para evitar situaciones que comprometan la seguridad del sistema.
Datos específicos	Ninguno.

El Requerimiento 8 se obtuvo con el fin de sesión de usuario o la ejecución del sistema.

4.2 Requerimientos funcionales (RF)

Los requerimientos funcionales son los que definen las funciones que el sistema será capaz de realizar, describen las transformaciones que el sistema realiza sobre las entradas para producir salidas. Estos requerimientos al tiempo que avanza el proyecto de software se convierten en los algoritmos, la lógica y gran parte del código del sistema Chaves (2005).

Requerimiento funcional 1: Inicio de sesión.

RF-1	**Inicio de sesión**	
Requerimientos asociados	R-1: Acceso al sistema.	
Descripción	Debe ser posible registrar las credenciales en el sistema para su posterior uso en los formularios correspondientes.	
Precondición	La aplicación no se encuentra en ejecución.	
Secuencia normal	**Paso**	**Acción**
	1	Iniciar la aplicación.
	2	Se muestra al usuario el formulario de inicio de sesión solicitando nombre de usuario, contraseña y sucursal.
	3	El usuario ingresa sus credenciales y solicita el inicio de sesión.
	4	El sistema verifica las credenciales y obtiene el rol del usuario autenticado.
	5	El sistema notifica el inicio de sesión y carga el menú principal.
Postcondición	En función del rol del usuario serán las opciones que aparezcan en el menú.	
Excepciones	**Paso**	**Acción**
	3	El usuario puede cancelar el inicio de sesión y finalizar la ejecución de la aplicación.
	3	El sistema muestra alertas en función de la validez de los datos ingresados.

En el Requerimiento funcional 1 se especifica el procedimiento que se llevan a cabo, así como acciones que no completan las fases del requerimiento. De igual manera se presentan las condiciones a cumplir para iniciar con el procedimiento.

Requerimiento funcional 2: Nuevo producto.

RF-2	**Nuevo producto**	
Requerimientos asociados	R-2: Catálogo de productos.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Administrador”, en el cual la opción “Productos” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra la opción “Agregar” que muestra al usuario un formulario de registro.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para registrar productos.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Administrador” y se despliega un submenú con la opción “Productos”.
	2	Se da clic en la opción “Productos” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se da clic en el botón “Agregar”, el cual despliega una ventana emergente de registro.
	4	El usuario debe realizar el registro de cada producto llenando los campos solicitados.
	5	Los campos solicitados son: Marca, Selector de producto (SAT), Unidad de medida (SAT), Tipo de producto, Código de barras (se puede escanear), Precio, Descripción y una foto.
	6	Una vez capturados los datos, se da clic en el botón “Guardar”, el sistema muestra una alerta notificando el resultado de la operación.
	7	El sistema limpia los campos de registro y si es necesario registrar más productos se vuelve a iniciar en el Paso 4.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar el registro de producto y finalizar con el proceso actual.

En el Requerimiento funcional 2 se especifica el procedimiento con fases ordenadas de manera secuencial para llevar a cabo el registro de productos al sistema, el procedimiento finaliza cuando se deja de registrar productos.

Requerimiento funcional 3: Buscar producto.

RF-3	**Buscar producto**	
Requerimientos asociados	R-2: Catálogo de productos.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Administrador”, en el cual la opción “Productos” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre en donde se muestran los productos registrados.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para visualizar la opción de “Productos” en el apartado “Administrador”	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Administrador” y se despliega un submenú con la opción “Productos”.
	2	Se da clic en la opción “Productos” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se pueden visualizar los productos registrados.
Excepciones	**Paso**	**Acción**
	3	Si no se cargan productos en el formulario, se encuentra la opción de actualizar que reinicia la búsqueda.

El Requerimiento funcional 3 especifica una breve descripción sobre la funcionalidad del sistema que permite la búsqueda de producto, de igual manera se presenta el procedimiento para llevar a cabo dicha acción.

Requerimiento funcional 4: Modificar producto.

RF-4	**Modificar producto**	
Requerimientos asociados	R-2: Catálogo de productos.	
Descripción	El sistema muestra en el menú principal el apartado llamado "Administrador", en el cual la opción "Productos" aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra el botón de edición que muestra al usuario un formulario para editar los datos del producto.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para editar los datos de los productos.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción "Administrador" y se despliega un submenú con la opción "Productos".
	2	Se da clic en la opción "Productos" y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se muestran los productos registrados con un botón de edición en cada producto.
	4	El usuario da clic en el botón de edición y el sistema despliega una ventana emergente con los campos precargados del producto elegido.
	5	Los campos que se muestran son: Marca, Selector de producto (SAT), Unidad de medida (SAT), Tipo de producto, Código de barras (se puede escanear), Precio, Descripción y una foto.
	6	Una vez modificados los datos, se da clic en el botón "Actualizar", el sistema muestra una alerta notificando el resultado de la operación.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar la modificación del producto y finalizar con el proceso actual.

El Requerimiento funcional 4 es similar al requerimiento llamado Nuevo producto, dado que se utilizan los mismos datos, es importante mencionar que el requerimiento funcional de modificación finaliza al realizar cambios en el producto seleccionado.

Requerimiento funcional 5: Eliminar producto.

RF-5	**Eliminar producto**	
Requerimientos asociados	R-2: Catálogo de productos.	
Descripción	El sistema muestra en el menú principal el apartado llamado "Administrador", en el cual la opción "Productos" aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra el botón de eliminación de productos.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para dar de baja productos.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción "Administrador" y se despliega un submenú con la opción "Productos".
	2	Se da clic en la opción "Productos" y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se muestran los productos registrados con un botón de eliminación en cada producto.
	4	El usuario da clic en el botón de eliminar y el sistema muestra un mensaje de confirmación para eliminar el producto.
	5	La lista de productos mostrada se actualiza automáticamente.
Excepciones	**Paso**	**Acción**
	4	Si aún se encuentran existencias del producto a eliminar la operación finaliza sin éxito.
	4	El usuario puede cancelar la eliminación de un producto finalizando el proceso actual.

Para llevar a cabo la eliminación de productos, se obtuvo el Requerimiento funcional 5, en el cual se especifica el procedimiento por fases de manera secuencial, de igual manera, se presentan los casos donde se podría interrumpir dicho procedimiento.

Requerimiento funcional 6: Nuevo proveedor.

RF-6	**Nuevo proveedor**	
Requerimientos asociados	R-3: Catálogo de proveedores.	
Descripción	El sistema muestra en el menú principal el apartado llamado "Administrador", en el cual la opción "Proveedores" aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra la opción "Agregar" que muestra al usuario un formulario de registro	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para registrar proveedores.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción "Administrador" y se despliega un submenú con la opción "Proveedores".
	2	Se da clic en la opción "Proveedores" y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se da clic en el botón "Agregar", el cual despliega una ventana emergente de registro.
	4	El usuario debe realizar el registro de cada proveedor llenando los campos solicitados.
	5	Los campos solicitados son: opción de elegir entre persona moral y persona física, nombre y apellido (si se elige persona física), nombre de empresa (si se elige persona moral), correo, teléfono, dirección, código postal, estado, ciudad y RFC.
	6	Una vez capturados los datos, se da clic en el botón "Guardar", el sistema muestra una alerta notificando el resultado de la operación
	7	El sistema limpia los campos de registro y si es necesario registrar más proveedores se vuelve a iniciar en el Paso 4.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar el registro de proveedor y finalizar con el proceso actual.

Para registrar proveedores de producto desde el sistema se obtuvo el Requerimiento funcional 6, en donde se especifica de manera secuencial las fases del sistema para llevar a cabo dicho registro.

Requerimiento funcional 7: Buscar proveedor.

RF-7	**Buscar proveedor**	
Requerimientos asociados	R-3: Catálogo de proveedores.	
Descripción	El sistema muestra en el menú principal el apartado llamado "Administrador", en el cual la opción "Proveedor" aparece en el submenú, dicha opción muestra un formulario con el mismo nombre en donde se muestran los proveedores registrados.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para visualizar la opción de "Proveedores" en el apartado "Administrador".	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción "Administrador" y se despliega un submenú con la opción "Proveedores".
	2	Se da clic en la opción "Proveedores" y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se pueden visualizar los proveedores registrados.
Excepciones	**Paso**	**Acción**
	3	Si no se cargan proveedores en el formulario, se encuentra la opción de actualizar que reinicia la búsqueda.

El Requerimiento funcional 7 se obtuvo para satisfacer la necesidad de realizar búsquedas de proveedores, con el fin de agilizar diversos procesos y realizar peticiones de producto desde el sistema, para ello, se describe de manera secuencial las fases del requerimiento, además se detalla un caso particular donde no se completa la tarea de búsqueda.

Requerimiento funcional 8: Modificar proveedor.

RF-8	**Modificar proveedor**	
Requerimientos asociados	R-3: Catálogo de proveedores.	
Descripción	El sistema muestra en el menú principal el apartado llamado "Administrador", en el cual la opción "Proveedores" aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra el botón de edición que muestra al usuario un formulario para editar los datos del proveedor.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para editar los datos de los proveedores.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción "Administrador" y se despliega un submenú con la opción "Proveedores".
	2	Se da clic en la opción "Proveedores" y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se muestran los proveedores registrados con un botón de edición en cada proveedor.
	4	El usuario da clic en el botón de edición y el sistema despliega una ventana emergente con los campos precargados del proveedor elegido.
	5	Los campos solicitados son: opción de elegir entre persona moral y persona física, nombre y apellido (si se elige persona física), nombre de empresa (si se elige persona moral), correo, teléfono, dirección, código postal, estado, ciudad y RFC.
	6	Una vez modificados los datos, se da clic en el botón "Actualizar", el sistema muestra una alerta notificando el resultado de la operación.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar la modificación del proveedor y finalizar con el proceso actual.

Para realizar cambios en los datos de los proveedores se especifica de manera secuencial las acciones a seguir en el Requerimiento funcional 8, la especificación de este requerimiento es bastante similar al registro de proveedores, a su vez, es importante mencionar que algunos de sus datos no se pueden modificar.

Requerimiento funcional 9: Eliminar proveedor.

RF-9	**Eliminar proveedor**	
Requerimientos asociados	R-3: Catálogo de proveedores.	
Descripción	El sistema muestra en el menú principal el apartado llamado "Administrador", en el cual la opción "Proveedores" aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra el botón de eliminación de proveedor.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para dar de baja proveedores.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción "Administrador" y se despliega un submenú con la opción "Proveedores".
	2	Se da clic en la opción "Proveedores" y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se muestran los productos registrados con un botón de eliminación en cada proveedor.
	4	El usuario da clic en el botón de eliminar y el sistema muestra un mensaje de confirmación para eliminar el proveedor.
	5	La lista de proveedores mostrada se actualiza automáticamente.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar la eliminación de un proveedor finalizando el proceso actual.

Se incluyó la posibilidad de dar de baja a proveedores, para ello se obtuvo el Requerimiento funcional 9, en donde se especifican las fases del requerimiento, así como un caso de excepción en donde no se completa dicho requerimiento.

Requerimiento funcional 10: Nuevo stock.

RF-10	**Nuevo stock**	
Requerimientos asociados	R-4: Catálogo de stock.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Almacén”, en el cual la opción “Stock” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra la opción “Agregar” que muestra al usuario un formulario de registro.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para registrar stock.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Almacén” y se despliega un submenú con la opción “Stock”.
	2	Se da clic en la opción “Stock” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se da clic en el botón “Agregar”, el cual despliega una ventana emergente de registro.
	4	El usuario debe realizar el registro de cada producto en stock llenando los campos solicitados.
	5	Los campos solicitados son: producto y código de barras.
	6	Una vez capturados los datos, se da clic en el botón “Guardar”, el sistema muestra una alerta notificando el resultado de la operación
	7	El sistema limpia los campos de registro y si es necesario registrar más productos en stock se vuelve a iniciar en el Paso 4.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar el registro de nuevo stock y finalizar con el proceso actual.

El Requerimiento funcional 10 se obtuvo con el fin de registrar productos nuevos al stock, para ello se especifica el procedimiento para satisfacer dicho requerimiento. De igual manera se especifica un caso de excepción en el cual no se cumple la funcionalidad del requerimiento.

Requerimiento funcional 11: Buscar stock.

RF-11	**Buscar stock**	
Requerimientos asociados	R-4: Catálogo de stock.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Almacén”, en el cual la opción “Stock” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre en donde se muestran los productos en stock registrados.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para visualizar la opción de “Stock” en el apartado “Almacén”.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Almacén” y se despliega un submenú con la opción “Stock”.
	2	Se da clic en la opción “Stock” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se pueden visualizar los productos en stock registrados.
Excepciones	**Paso**	**Acción**
	3	Si no se cargan los productos en stock en el formulario, se encuentra la opción de actualizar que reinicia la búsqueda.

Con el fin de agilizar distintos procesos (por ejemplo, ventas de producto), se obtuvo el Requerimiento funcional 11, el cual especifica la secuencia a seguir, además de un caso de excepción en el cual no se cumple dicho requerimiento.

Requerimiento funcional 12: Nuevo cliente.

RF-12	**Nuevo cliente**	
Requerimientos asociados	R-5: Catálogo de clientes.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Ventas”, en el cual la opción “Clientes” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra la opción “Agregar” que muestra al usuario un formulario de registro.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para registrar clientes.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Ventas” y se despliega un submenú con la opción “Clientes”.
	2	Se da clic en la opción “Clientes” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se da clic en el botón “Agregar”, el cual despliega una ventana emergente de registro.
	4	El usuario debe realizar el registro de cada cliente llenando los campos solicitados.
	5	Los campos solicitados son: opción de elegir entre persona moral y persona física, nombre y apellido (si se elige persona física), nombre de empresa (si se elige persona moral), correo, teléfono, dirección, código postal, estado, ciudad y RFC.
	6	Una vez capturados los datos, se da clic en el botón “Guardar”, el sistema muestra una alerta notificando el resultado de la operación
	7	El sistema limpia los campos de registro y si es necesario registrar más clientes se vuelve a iniciar en el Paso 4.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar el registro de un cliente y finalizar con el proceso actual.

El Requerimiento funcional 12 satisface la necesidad de realizar el registro de clientes, mediante el cual se pueden agilizar distintos trámites en la empresa. Es importante mencionar que el procedimiento de registro finaliza cuando se deja de registrar clientes, para ello se especifica un caso de excepción el cual finaliza la ejecución de actividades del requerimiento.

Requerimiento funcional 13: Buscar cliente.

RF-13	**Buscar cliente**	
Requerimientos asociados	R-5: Catálogo de clientes.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Ventas”, en el cual la opción “Clientes” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre en donde se muestran los clientes registrados.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para visualizar la opción de “Clientes” en el apartado “Ventas”.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Ventas” y se despliega un submenú con la opción “Clientes”.
	2	Se da clic en la opción “Clientes” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se pueden visualizar los clientes registrados.
Excepciones	**Paso**	**Acción**
	3	Si no se cargan los clientes en el formulario, se encuentra la opción de actualizar que reinicia la búsqueda.

En el Requerimiento funcional 13 se especifica la secuencia de acciones para realizar la búsqueda de clientes, de igual manera se especifica un caso de excepción en el cual no se completa la acción de búsqueda.

Requerimiento funcional 14: Modificar cliente.

RF-14	**Modificar cliente**	
Requerimientos asociados	R-5: Catálogo de clientes.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Ventas”, en el cual la opción “Clientes” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra el botón de edición que muestra al usuario un formulario para editar los datos del cliente.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para editar los datos de los clientes.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Ventas” y se despliega un submenú con la opción “Clientes”.
	2	Se da clic en la opción “Clientes” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se muestran los clientes registrados con un botón de edición en cada cliente.
	4	El usuario da clic en el botón de edición y el sistema despliega una ventana emergente con los campos precargados del cliente elegido.
	5	Los campos solicitados son: opción de elegir entre persona moral y persona física, nombre y apellido (si se elige persona física), nombre de empresa (si se elige persona moral), correo, teléfono, dirección, código postal, estado, ciudad y RFC.
	6	Una vez modificados los datos, se da clic en el botón “Actualizar”, el sistema muestra una alerta notificando el resultado de la operación.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar la modificación del cliente y finalizar con el proceso actual.

Para realizar cambios en los datos de los clientes, se obtuvo el Requerimiento funcional 14, en él se detallan de manera secuencial las acciones necesarias para realizar dicha actividad, es importante mencionar que algunos datos de los clientes no se pueden modificar.

Requerimiento funcional 15: Eliminar cliente.

RF-15	**Eliminar cliente**	
Requerimientos asociados	R-5: Catálogo de clientes.	
Descripción	El sistema muestra en el menú principal el apartado llamado "Ventas", en el cual la opción "Clientes" aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra el botón de eliminación de clientes.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para dar de baja a clientes.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción "Ventas" y se despliega un submenú con la opción "Clientes".
	2	Se da clic en la opción "Clientes" y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se muestran los productos registrados con un botón de eliminación en cada cliente.
	4	El usuario da clic en el botón de eliminar y el sistema muestra un mensaje de confirmación para eliminar el cliente.
	5	La lista de clientes mostrada se actualiza automáticamente.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar la eliminación de un cliente finalizando el proceso actual.

Al igual que con los proveedores, se incluyó la opción de dar de baja un cliente, para ello se obtuvo el Requerimiento funcional 15, en el cual se especifican las acciones a realizar. De igual manera se incluye un caso de excepción en el cual no se completa dicha operación.

Requerimiento funcional 16: Nueva preventa.

RF-16	**Nueva preventa**	
Requerimientos asociados	R-6: Catálogo de preventas.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Ventas”, en el cual la opción “Preventas” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra la opción “Agregar” que muestra al usuario un formulario de registro.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para registrar preventas.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Ventas” y se despliega un submenú con la opción “Preventas”.
	2	Se da clic en la opción “Preventas” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se da clic en el botón “Agregar”, el cual despliega una ventana emergente de registro.
	4	El usuario debe realizar el registro de cada preventa llenando los campos solicitados.
	5	Los campos solicitados son: cliente, producto, cantidad, código, forma de pago y tipo de venta.
	6	Una vez capturados los datos, se da clic en el botón “Guardar”, el sistema muestra una alerta notificando el resultado de la operación
	7	El sistema limpia los campos de registro y si es necesario registrar más preventas se vuelve a iniciar en el Paso 4.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar el registro de preventas y finalizar con el proceso actual.

En el Requerimiento funcional 16 se especifica de manera secuencial el proceso para realizar preventas, es importante mencionar que dicha operación finaliza en el caso de excepción especificado.

Requerimiento funcional 17: Buscar preventa.

RF-17	**Buscar ventana**	
Requerimientos asociados	R-6: Catálogo de preventas.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Ventas”, en el cual la opción “Preventas” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre en donde se muestran las preventas registradas.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para visualizar la opción de “Preventas” en el apartado “Ventas”.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Ventas” y se despliega un submenú con la opción “Preventas”.
	2	Se da clic en la opción “Preventas” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se pueden visualizar las preventas registradas.
Excepciones	**Paso**	**Acción**
	3	Si no se cargan preventas en el formulario, se encuentra la opción de actualizar que reinicia la búsqueda.

En caso de realizar una búsqueda de preventas, se obtuvo el Requerimiento funcional 17, en el cual se especifica el procedimiento a realizar para llevar a cabo dicha operación, de igual manera, se puede presentar el caso de excepción especificado.

Requerimiento funcional 18: Modificar preventa.

RF-18	**Modificar preventa**	
Requerimientos asociados	R-6: Catálogo de preventas.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Ventas”, en el cual la opción “Preventas” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra el botón de edición que muestra al usuario un formulario para editar los datos de preventas.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para editar los datos de las preventas.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Ventas” y se despliega un submenú con la opción “Preventas”.
	2	Se da clic en la opción “Preventas” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se muestran las preventas registradas con un botón de edición en cada preventa.
	4	El usuario da clic en el botón de edición y el sistema despliega una ventana emergente con los campos precargados de la preventa elegida.
	5	Los campos solicitados son: cliente, producto, cantidad, código, forma de pago y tipo de venta.
	6	Una vez modificados los datos, se da clic en el botón “Actualizar”, el sistema muestra una alerta notificando el resultado de la operación.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar la modificación de la preventa y finalizar con el proceso actual.

En el Requerimiento funcional 18 se especifica una secuencia de acciones la cual es similar al requerimiento funcional que permite registrar preventas (ver Requerimiento funcional 16: Nueva preventa), asimismo, se especifica un caso de excepción en el que no es posible completar dicha operación.

Requerimiento funcional 19: Eliminar preventa.

RF-19	**Eliminar preventa**	
Requerimientos asociados	R-6: Catálogo de preventas.	
Descripción	El sistema muestra en el menú principal el apartado llamado “Ventas”, en el cual la opción “Preventas” aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra el botón de eliminación de preventa.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para dar de baja preventas.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción “Ventas” y se despliega un submenú con la opción “Preventas”.
	2	Se da clic en la opción “Preventas” y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se muestran los productos registrados con un botón de eliminación en cada preventa.
	4	El usuario da clic en el botón de eliminar y el sistema muestra un mensaje de confirmación para eliminar la preventa;
	5	La lista de preventas mostrada se actualiza automáticamente.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar la eliminación de una preventa finalizando el proceso actual.

Como se contempló incluir en el sistema la posibilidad de cancelación, se obtuvo el Requerimiento funcional 19, en el cual se especifica el procedimiento a realizar para llevar a cabo dicha operación. Por otro lado, es posible que se presente un caso de excepción, mediante el cual no es posible cancelar una preventa.

Requerimiento funcional 20: Nueva garantía.

RF-20	**Nueva garantía**	
Requerimientos asociados	R-7: Catálogo de garantías.	
Descripción	El sistema muestra en el menú principal el apartado llamado "Administrador", en el cual la opción "Garantías" aparece en el submenú, dicha opción muestra un formulario con el mismo nombre, en él se encuentra la opción "Agregar" que muestra al usuario un formulario de registro.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para registrar garantías.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción "Administrador" y se despliega un submenú con la opción "Garantías".
	2	Se da clic en la opción "Garantías" y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se da clic en la pestaña "Agregar".
	4	El usuario debe realizar el registro de cada garantía llenando los campos solicitados.
	5	Los campos solicitados son: tipo de técnico, producto, unidad que falla, y una breve descripción del problema.
	6	Una vez capturados los datos, se da clic en el botón "Guardar", el sistema muestra una alerta notificando el resultado de la operación
	7	El sistema limpia los campos de registro y si es necesario registrar más garantías se vuelve a iniciar en el Paso 4.
Excepciones	**Paso**	**Acción**
	4	El usuario puede cancelar el registro de garantías y finalizar con el proceso actual.

El Requerimiento funcional 20 permitió hacer válida la garantía de un producto en cual no funcione adecuadamente, para ello, se especifican las acciones ordenadas de manera secuencial a realizar para llevar a cabo dicha operación. En caso de ocurrir un evento inesperado, se presenta un caso de excepción en el cual no se registra la garantía.

Requerimiento funcional 21: Buscar garantía.

RF-21	**Buscar garantía**	
Requerimientos asociados	R-7: Catálogo de garantías.	
Descripción	El sistema muestra en el menú principal el apartado llamado "Administrador", en el cual la opción "Garantías" aparece en el submenú, dicha opción muestra un formulario con el mismo nombre en donde se muestra un formulario de búsqueda de garantías.	
Precondición	La aplicación debe estar ejecutándose y el usuario tener los permisos para visualizar la opción de "Garantías" en el apartado "Administrador"	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la opción "Administrador" y se despliega un submenú con la opción "Garantías".
	2	Se da clic en la opción "Garantías" y se carga el formulario que lleva el mismo nombre.
	3	En dicho formulario se ingresan los datos requeridos para realizar una búsqueda de las garantías existentes.
Excepciones	**Paso**	**Acción**
	3	El usuario puede cancelar la búsqueda finalizando así el proceso actual.

Para comprobar la existencia de una determinada garantía, se obtuvo el Requerimiento funcional 21, en el cual se especifica una condición a cumplir, y la secuencia de acciones a seguir para llevar a cabo dicha operación. De igual manera se puede presentar un caso de excepción por el cual no se completa la búsqueda.

Requerimiento funcional 22: Cerrar sesión.

RF-22	**Cerrar sesión**	
Requerimientos asociados	R-8: Salida del sistema	
Descripción	El sistema debe cerrar la sesión activa cuando el usuario deba terminar su turno.	
Precondición	La aplicación se debe encontrar en ejecución.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario da clic en la etiqueta de su perfil y se despliega un menú.
	2	Se da clic en opción “Logout”.
	3	El sistema cierra la sesión activa del usuario.

Para finalizar una sesión de trabajo, se obtuvo el Requerimiento funcional 22, el cual se especifican las acciones a realizar para completar dicha operación.

Requerimiento funcional 23: Salir del sistema.

RF-23	**Salir del sistema**	
Requerimientos asociados	R-8: Salida del sistema	
Descripción	El sistema termina la ejecución cuando el usuario desee salir.	
Precondición	La aplicación debe estar en tiempo de ejecución.	
Secuencia normal	**Paso**	**Acción**
	1	Sólo es necesario cerrar la pestaña en el navegador.
	2	El sistema se encuentra a la espera de ser consumido desde el servidor.

Para finalizar la ejecución del sistema se obtuvo el Requerimiento funcional 23 en el cual se describe la secuencia a seguir para salir del sistema.

4.3 Requerimientos no funcionales (RNF)

Los requerimientos no funcionales tienen que ver con características que de una u otra forma puedan limitar el sistema, como, por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad, estándares, etc. (Chávez, 2005).

Requerimiento no funcional 1: Interfaz del sistema.

Identificación del requerimiento	RNF-1
Nombre del requerimiento	Interfaz del sistema.
Características	El sistema debe presentar una interfaz de usuario sencilla que permita la facilidad de manejo a los usuarios.
Descripción del requerimiento	El sistema debe tener una interfaz sencilla e intuitiva.
Prioridad del requerimiento Alta	

El Requerimiento no funcional 1 representa un conjunto de descripciones y características acerca del apartado visual del sistema, es decir, el diseño de la interfaz de usuario.

Requerimiento no funcional 2: Desempeño.

Identificación del requerimiento	RNF-2
Nombre del requerimiento	Desempeño
Características	El sistema garantiza a los usuarios un óptimo rendimiento en cuestión de la información almacenada en el sistema, ofreciendo datos confiables.
Descripción del requerimiento	Garantiza el desempeño del sistema informático a los diferentes usuarios. En este sentido la información almacenada o registros realizados se pueden actualizar y consultar permanente y simultáneamente, sin afectar el tiempo de respuesta.
Prioridad del requerimiento Alta	

El Requerimiento no funcional 2 se obtuvo con la finalidad de llevar a cabo las optimizaciones pertinentes del sistema para un óptimo funcionamiento.

Requerimiento no funcional 3: Rol de usuario.

Identificación del requerimiento	RNF-3
Nombre del requerimiento	Rol de usuario
Características	Garantiza al usuario el acceso a la información en función del rol que posee.
Descripción del requerimiento	Facilidad y controles para permitir el acceso a la información al personal autorizado, de esta manera es posible consultar y cargar información pertinente para cada una de ellas.
Prioridad del requerimiento Alta	

En el Requerimiento no funcional 3 se declara la capacidad del sistema para otorgar distintos permisos y restricciones a los usuarios en función del rol asignado, con el fin de optimar la seguridad del sistema.

Requerimiento no funcional 4: Confiabilidad continua del sistema.

Identificación del requerimiento	RNF-4
Nombre del requerimiento	Confiabilidad continua del sistema
Características	El sistema se debe encontrar activo las 24 horas del día los siete días de la semana.
Descripción del requerimiento	La disponibilidad del sistema debe ser continua con un nivel de servicio para usuarios de tiempo completo.
Prioridad del requerimiento Alta	

En el Requerimiento no funcional 4 se declara la capacidad de acceder al sistema con frecuencia, e incluso, el tiempo en el que se encuentre operando, con la finalidad de hacer uso del sistema en cualquier momento.

Requerimiento no funcional 5: Seguridad en información.

Identificación del requerimiento	RNF-5
Nombre del requerimiento	Seguridad en información
Características	Garantiza al usuario final seguridad en cuanto a la información manipulada por el sistema.
Descripción del requerimiento	Garantiza la seguridad del sistema con respecto a la información y datos que se manejan tales sean documentos, contraseñas, etc.
Prioridad del requerimiento Alta	

El Requerimiento no funcional 5 contiene un conjunto de características y descripciones sobre temas relacionados con privacidad y seguridad de los datos que se pueden manipular en el sistema.

4.4 Interfaces de usuario

Por medio de imágenes, se presenta el diseño de la aplicación ERP, las imágenes que se muestran a continuación son de los formularios más relevantes a utilizar por el usuario final.

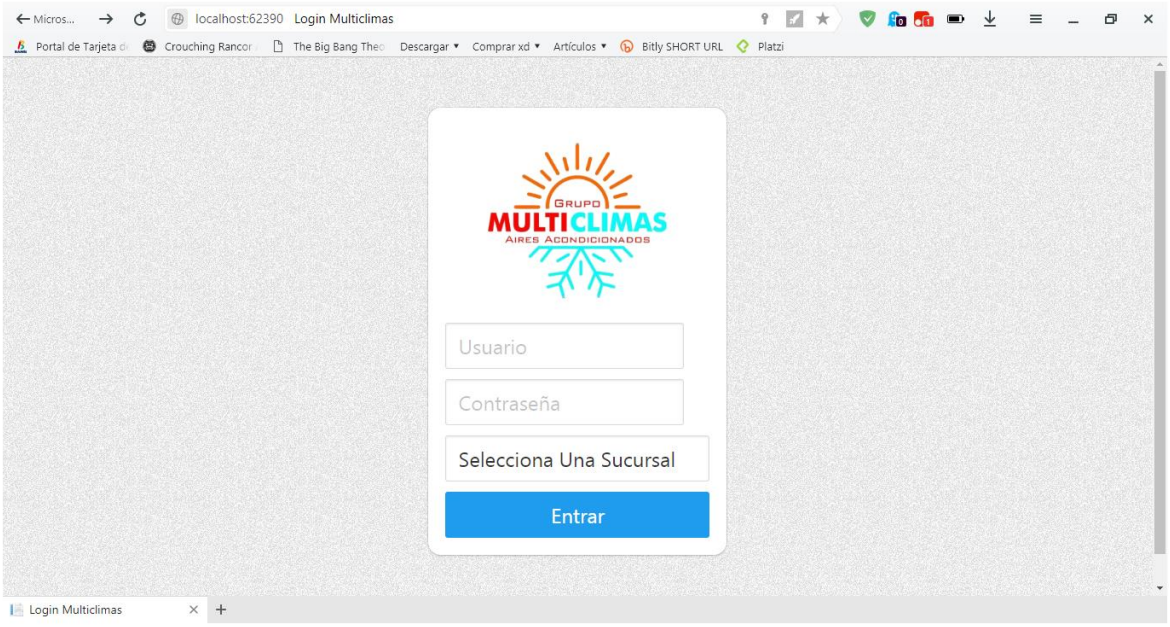

Figura 4.1: Formulario de inicio de sesión.

Se obtuvo un formulario de inicio de sesión que permite el acceso al sistema por sucursal (ver Figura 4.1).

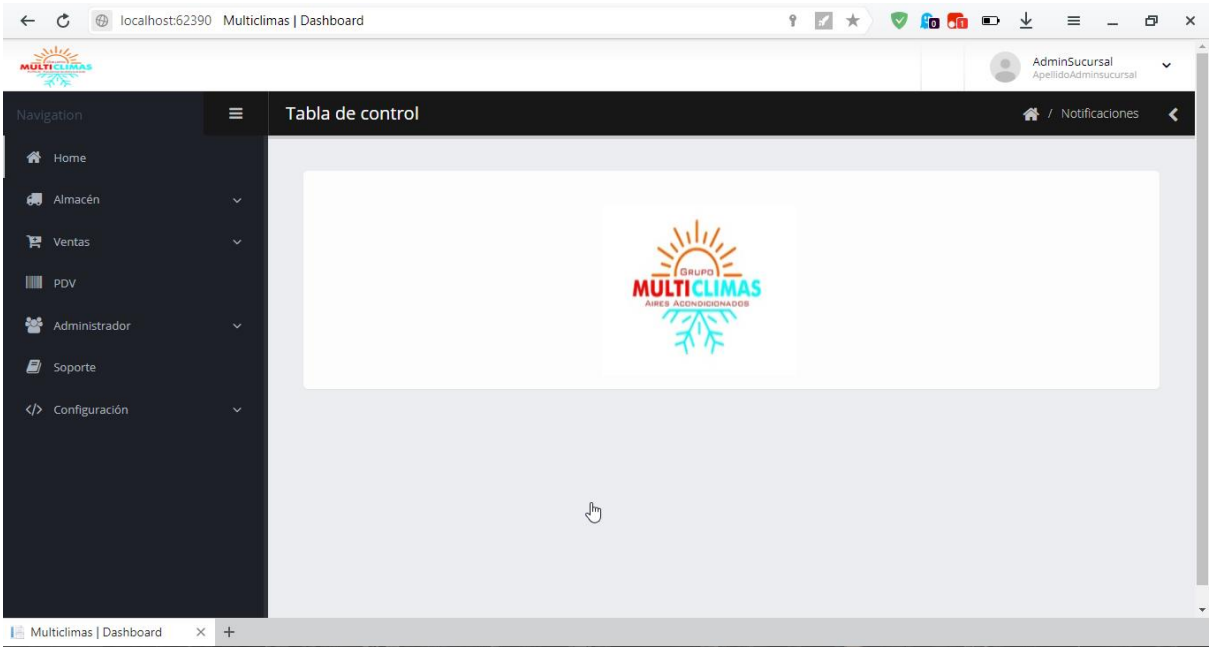

Figura 4.2: Formulario inicial.

Se obtuvo una interfaz de usuario en la que se muestra el menú principal de estilo DashBoard (ver Figura 4.2).

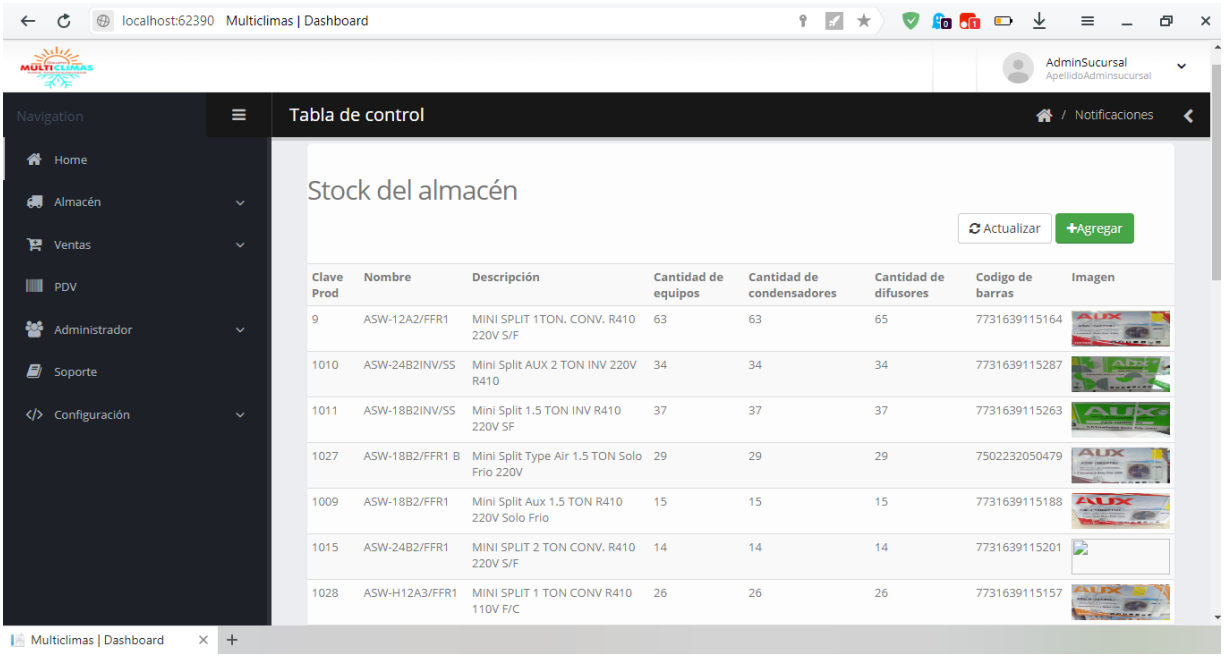

Figura 4.3: Formulario de stock.

Formulario en donde se puede visualizar el stock existente por sucursal (ver Figura 4.3).

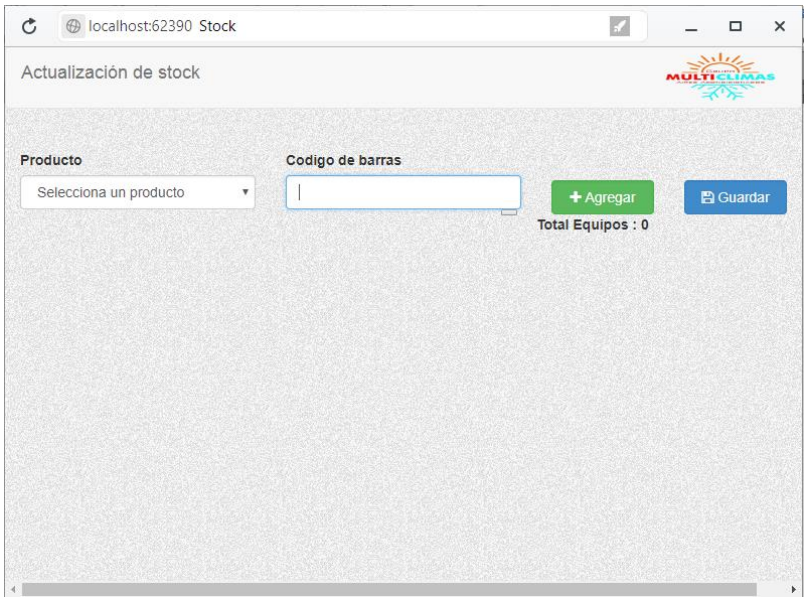

Figura 4.4: Formulario emergente (Pop-Up) de actualización de stock.

Se obtuvo la ventana emergente (ver Figura 4.4) que permite agregar existencias de productos al stock de la sucursal.

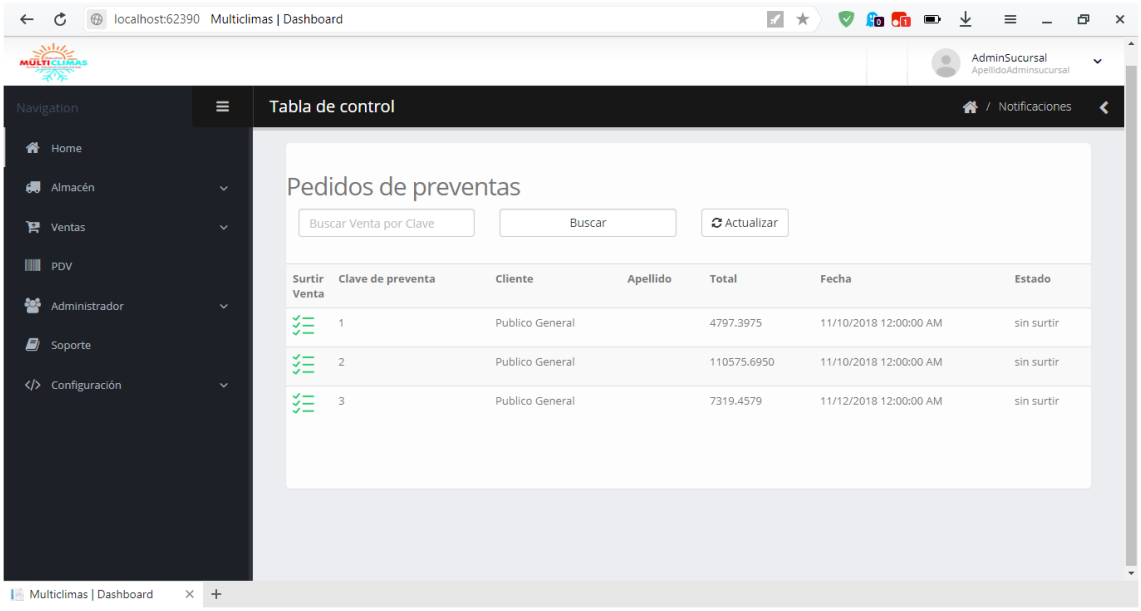

Figura 4.5: Formulario de preventas.

En la Figura 4.5 se muestra un formulario que permite la confirmación de pedidos para posteriormente abastecerlos de productos.

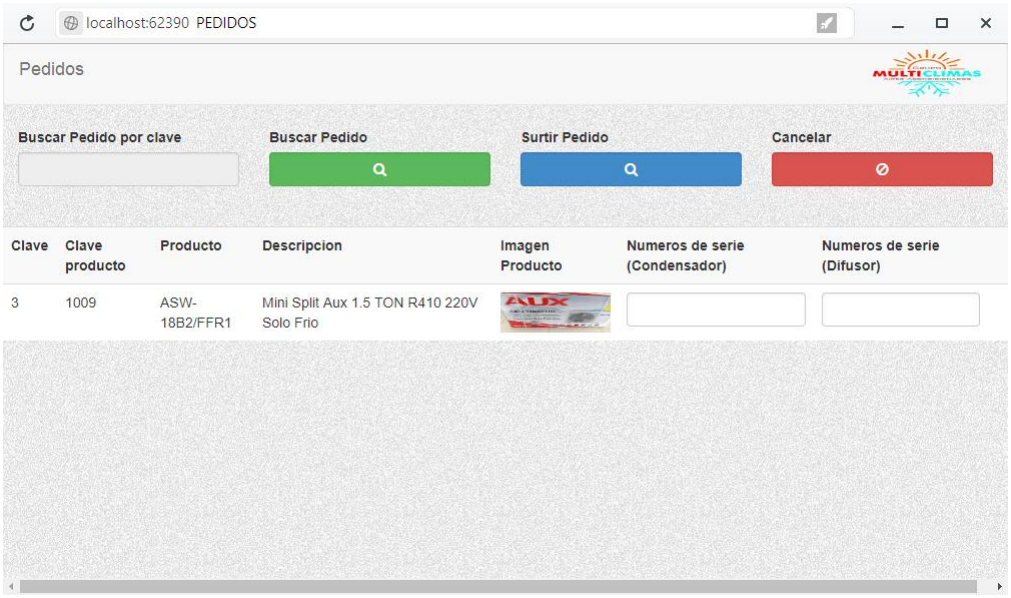

Figura 4.6: Formulario emergente de pedidos.

El formulario emergente (Pop-Up) que se muestra en la Figura 4.6 permite llevar a cabo el abastecimiento de productos para las preventas realizadas.

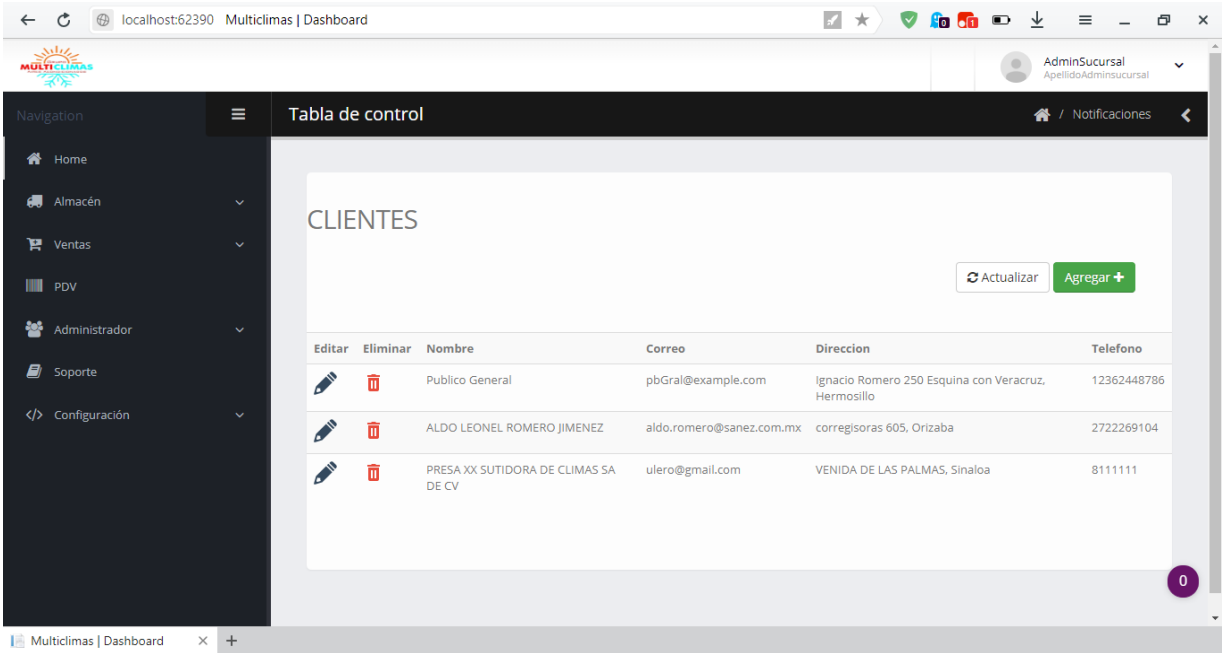

Figura 4.7: Formulario de clientes.

Se logró un formulario (ver Figura 4.7) que permite visualizar, editar y dar de baja a clientes previamente registrados.

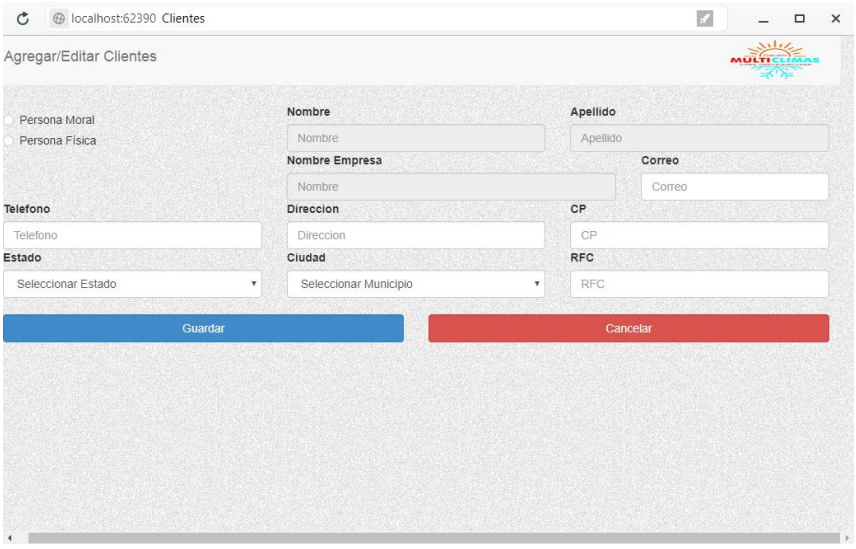

Figura 4.8: Formulario emergente de registro para clientes.

Formulario emergente cuya función consiste en dar de alta a distintos clientes (ver Figura 4.8).

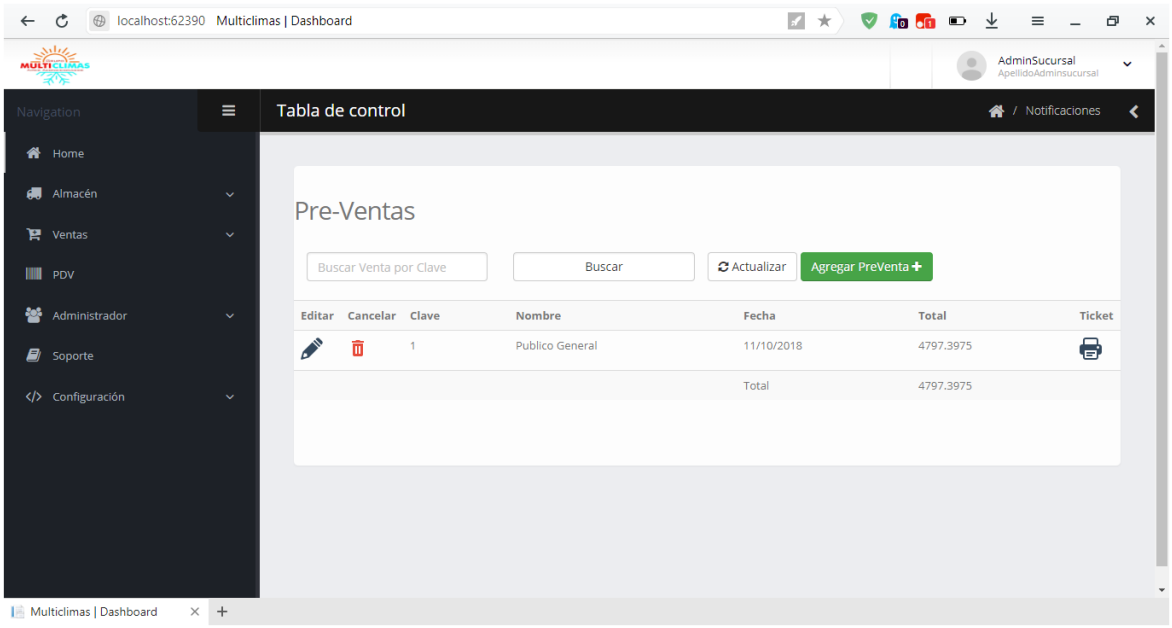

Figura 4.9: Formulario de enlistado de preventas.

Se obtuvo una interfaz de usuario (ver Figura 4.9) que permite la visualización, edición y cancelación de preventas, además de que es posible imprimir los correspondientes documentos.

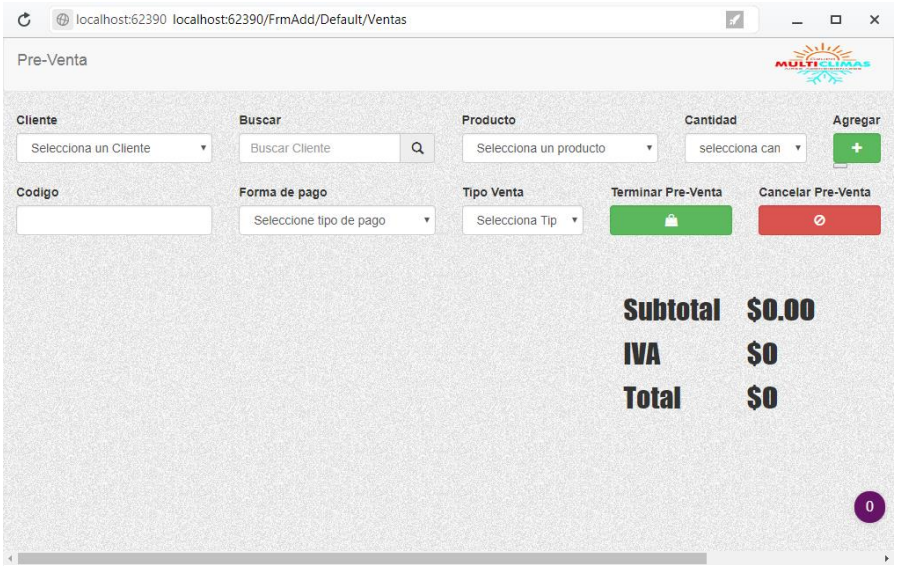

Figura 4.10: Ventana emergente de preventas.

Se consiguió un formulario emergente que permite el registro o edición de preventas (ver Figura 4.10).

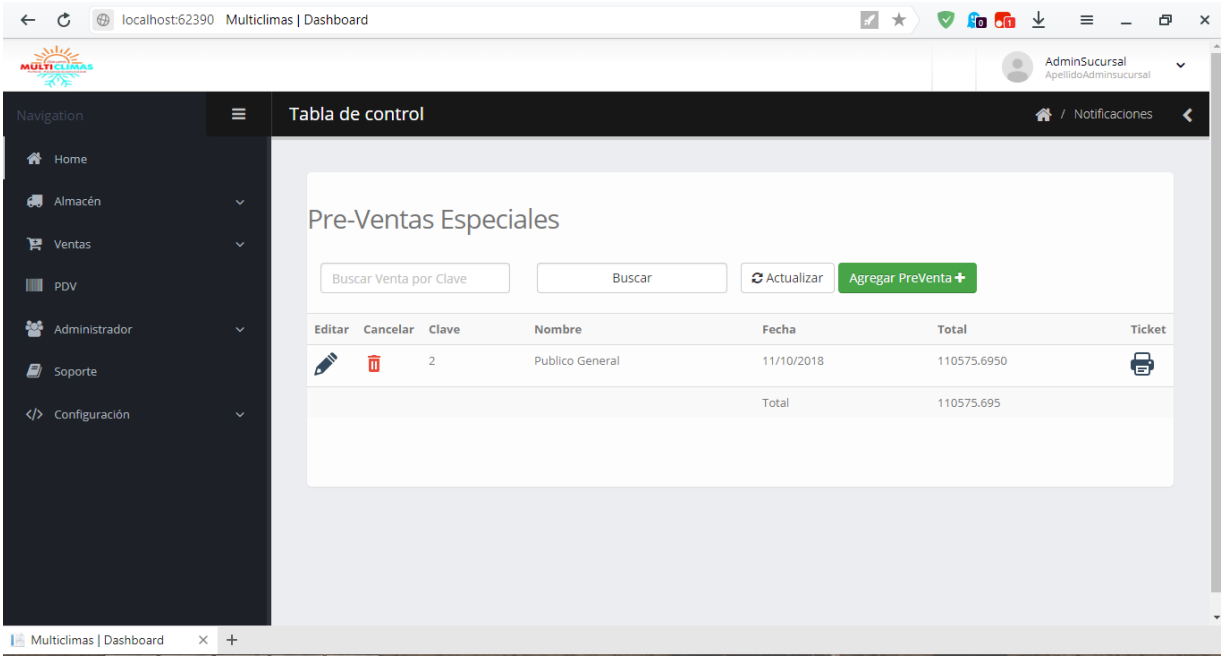

Figura 4.11: Formulario de preventas especiales.

Formulario de control (ver Figura 4.11) en el cual es posible editar, cancelar, e imprimir las preventas especiales registradas.

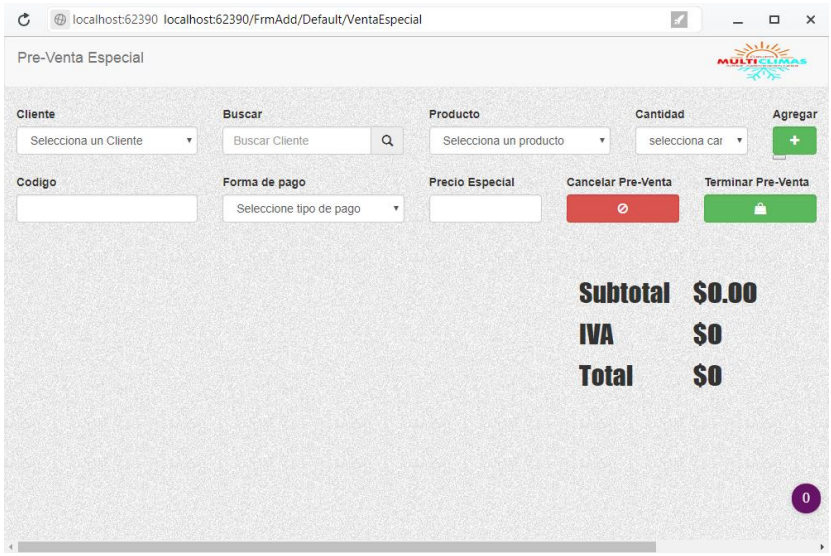

Figura 4.12: Ventana Pop-Up de preventas especiales.

Se obtuvo un formulario emergente que permite la edición o registro de preventas especiales (ver Figura 4.12).

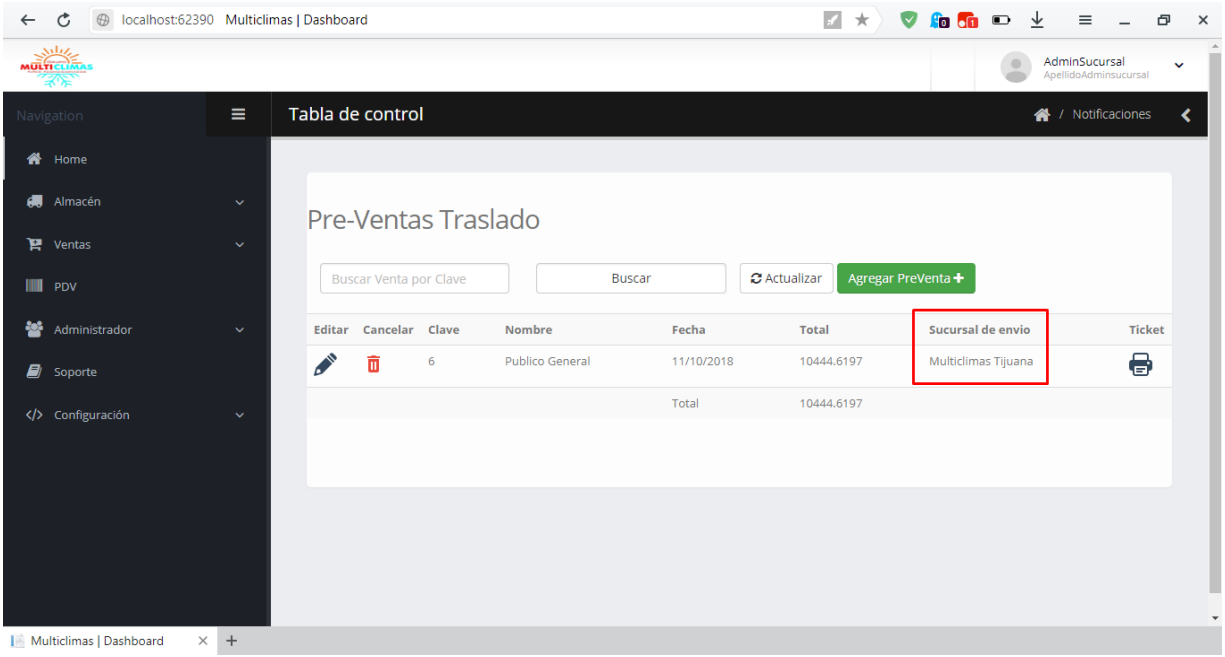

Figura 4.13: Preventas para traslado.

Formulario cuya finalidad consta de permitir la visualización, edición, cancelación e impresión de preventas realizadas a otras sucursales, de igual manera se especifica la sucursal de envío (ver Figura 4.13).

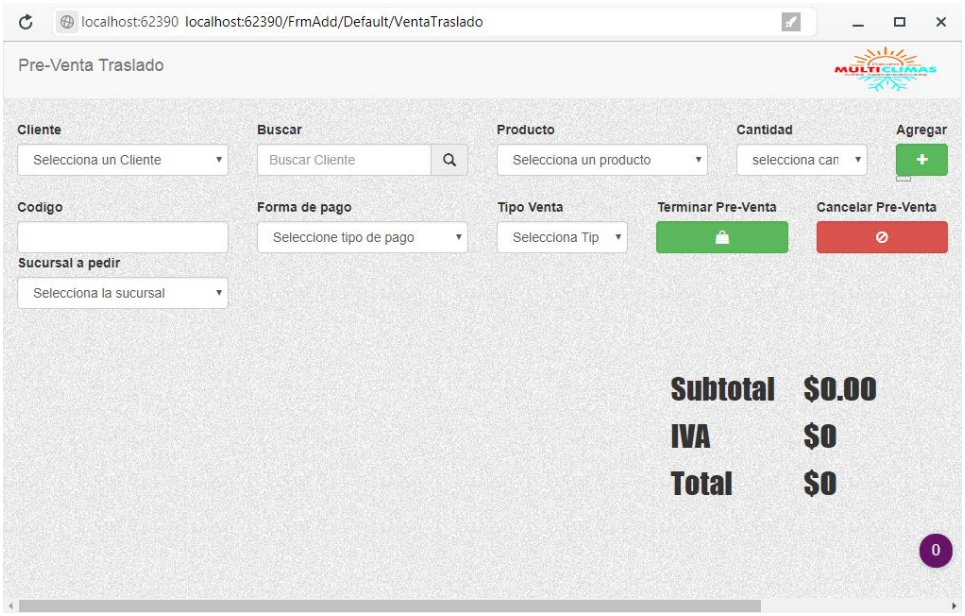

Figura 4.14: Ventana emergente de creación y edición de preventas para traslado.

El formulario emergente que se muestra en la Figura 4.14 permite dar de alta preventas a otras sucursales (Figura 4.14).

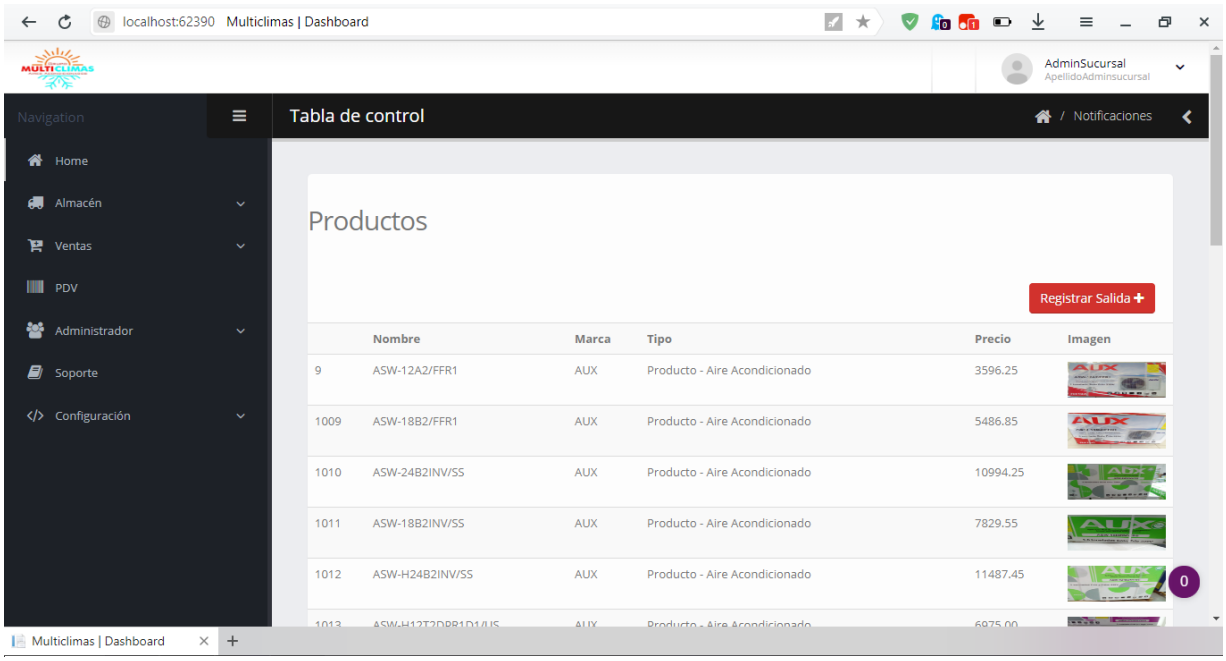

Figura 4.15: Formulario de productos.

Formulario (ver Figura 4.15) que permite visualizar los productos adquiridos por la empresa.

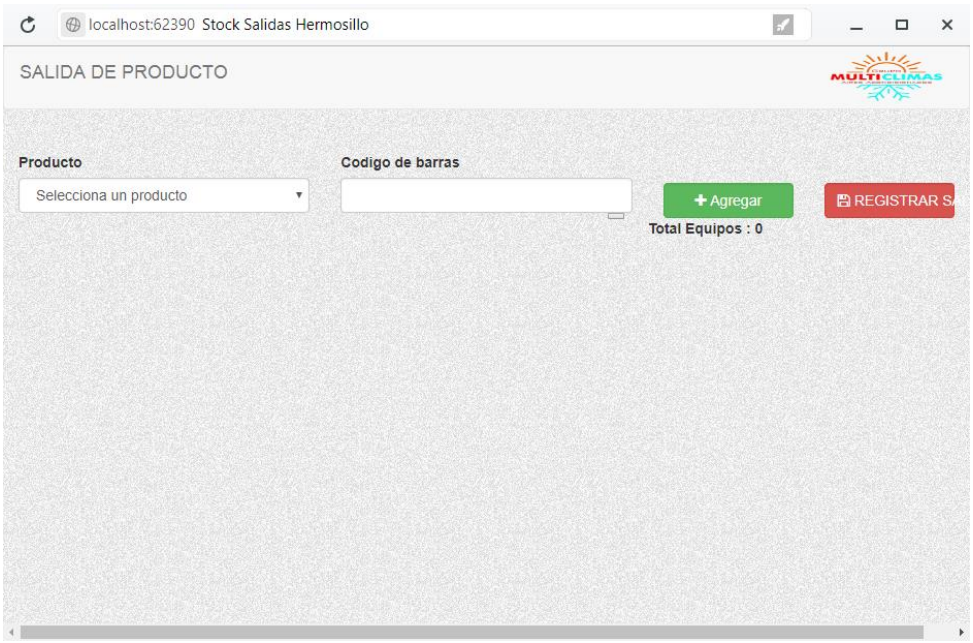

Figura 4.16: Formulario para el registro de salida de productos.

Formulario emergente cuyo objetivo consta de descontinuar productos ofrecidos por la empresa (ver Figura 4.16).

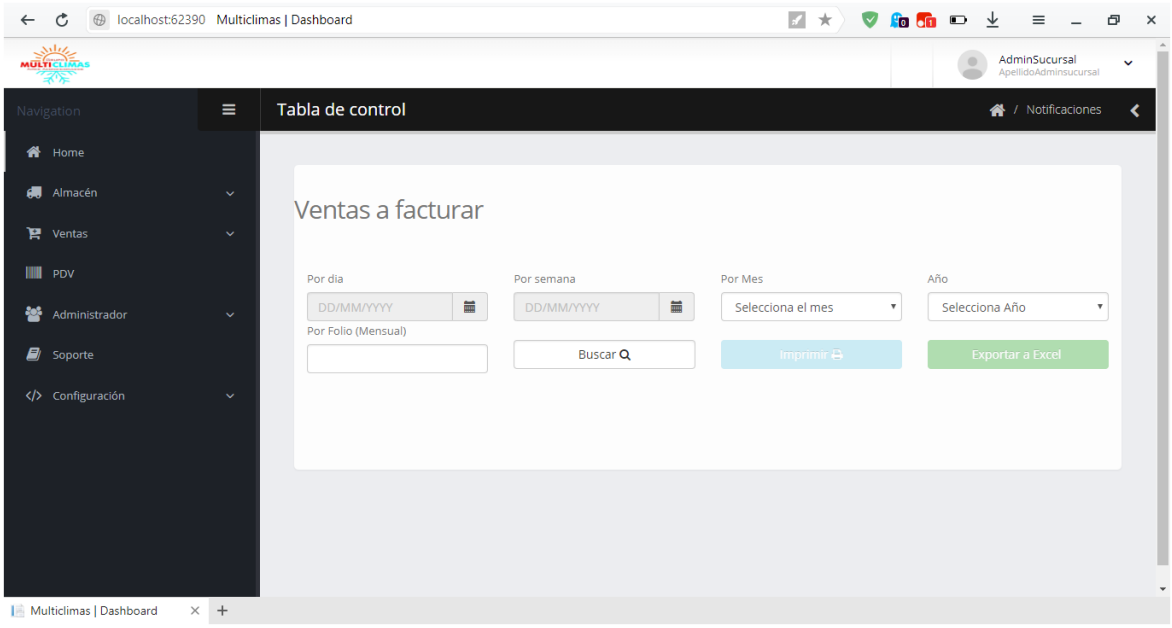

Figura 4.17: Formulario para la facturación de ventas.

Se obtuvo una interfaz de usuario que permite la búsqueda de facturas para posteriormente imprimirlas o exportarlas como hojas de cálculo (ver Figura 4.17).

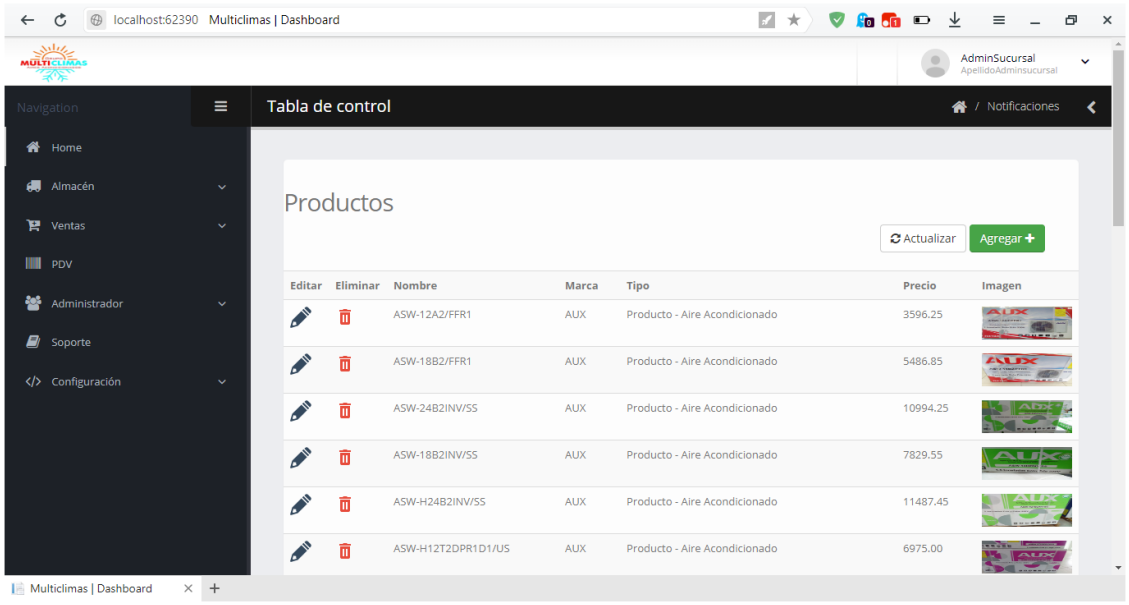

Figura 4.18: Formulario de productos.

Se logró desarrollar un formulario de control que permite la edición, descontinuación y visualización de los productos adquiridos por la empresa (ver Figura 4.18).

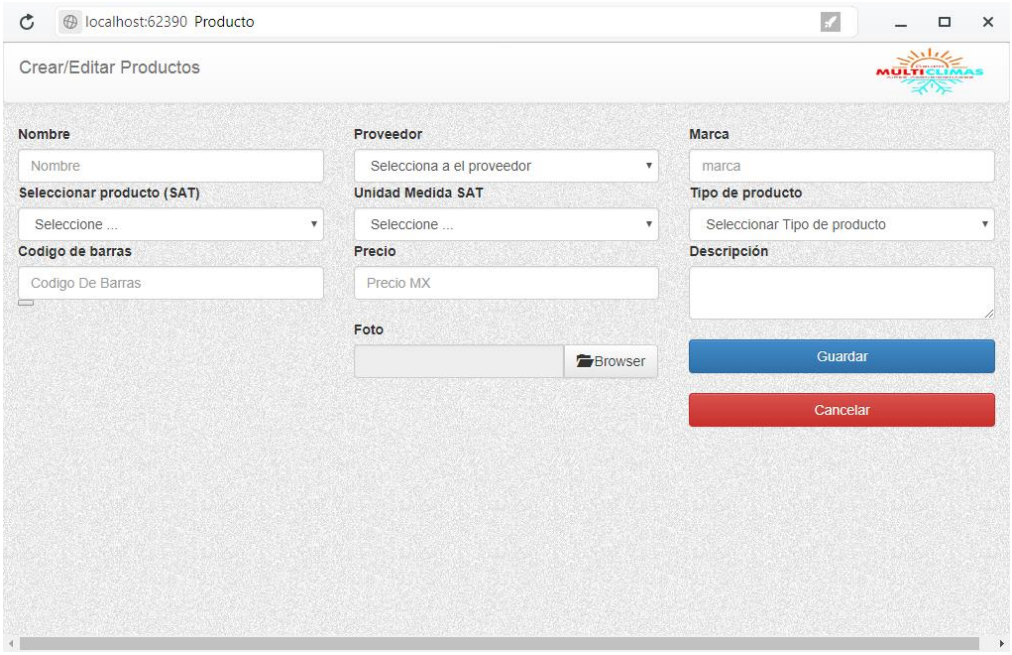

Figura 4.19: Formulario PopUp para dar de alta y editar productos.

Se obtuvo un formulario emergente (ver Figura 4.19) que permite dar de alta nuevos productos, asimismo, es posible editar los datos correspondientes a cada producto.

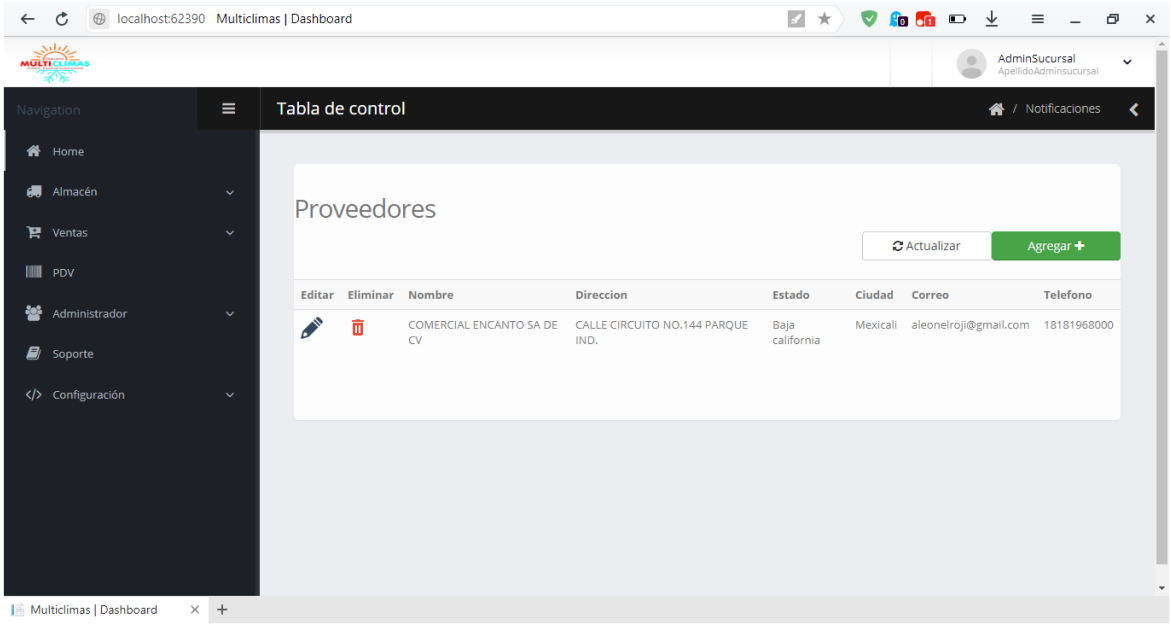

Figura 4.20: Formulario de proveedores.

Se logró desarrollar un formulario de control (ver Figura 4.20) el cual permite visualizar, dar de baja y modificar los datos de proveedores.

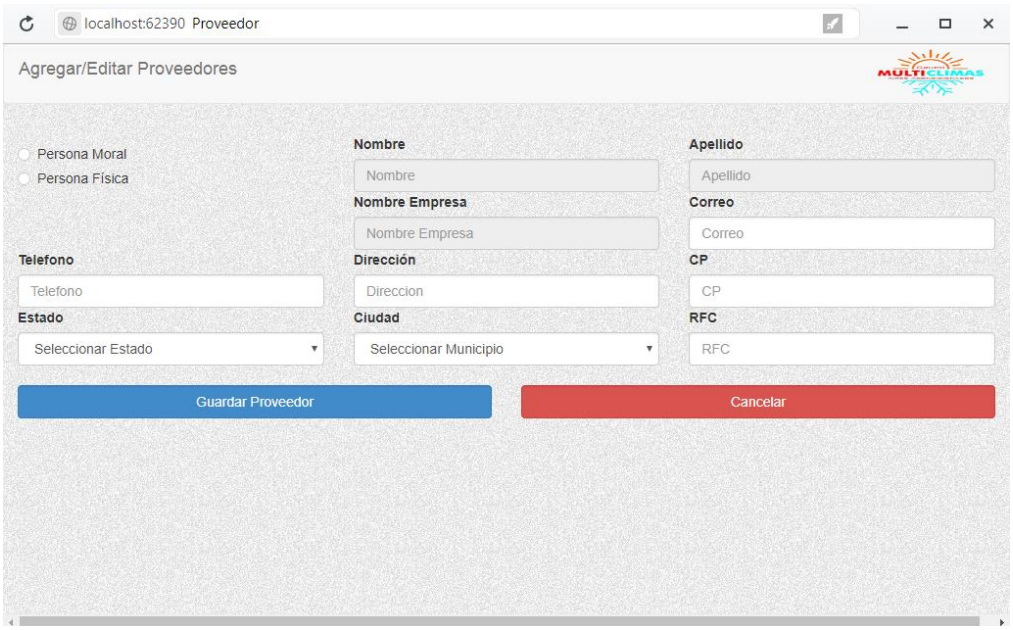

Figura 4.21: Ventana emergente para registro y edición de proveedores.

Formulario que permite dar de alta o cambiar datos sobre los proveedores de producto (ver Figura 4.21).

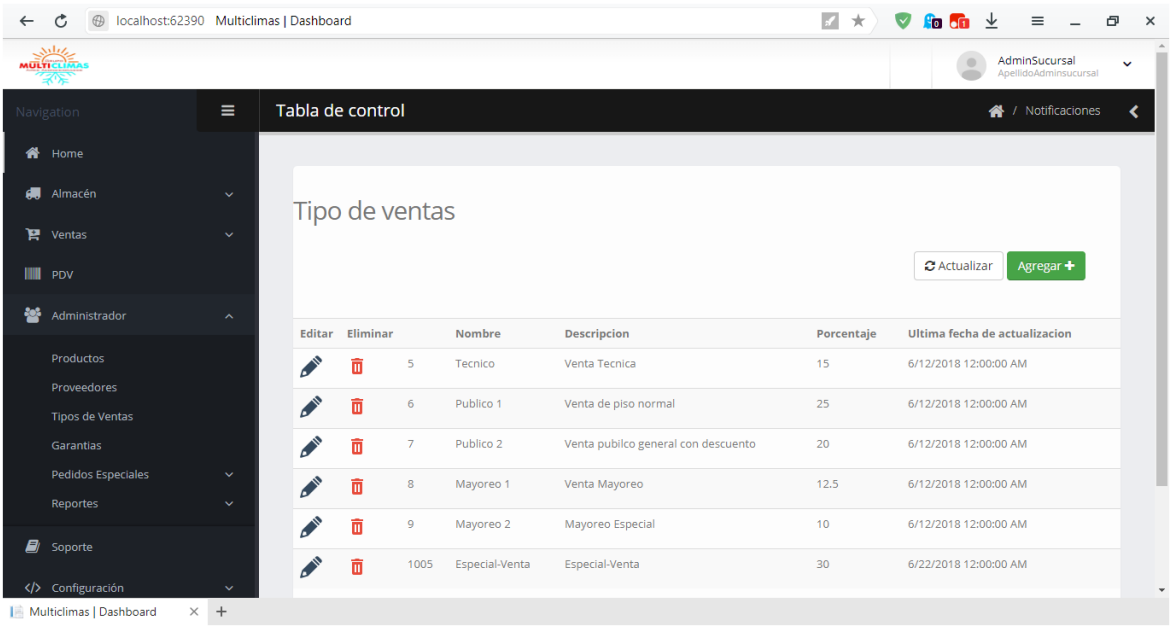

Figura 4.22: Formulario de tipo de ventas.

Se logró desarrollar un formulario de control que permite la gestión de los tipos de venta que existen en la empresa (ver Figura 4.22).

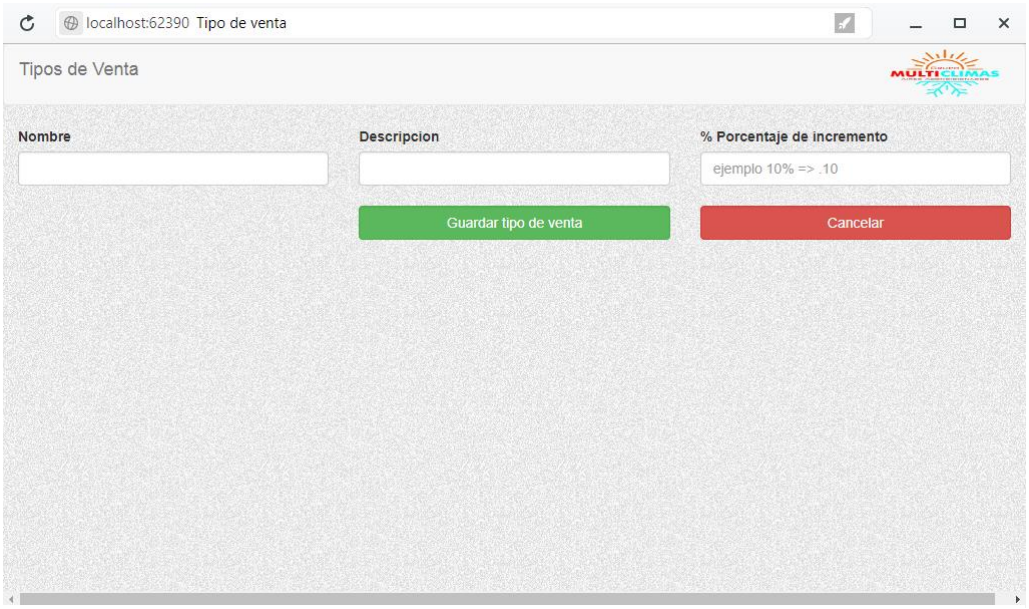

Figura 4.23: Formulario emergente para registro o edición de tipos de venta.

Se obtuvo el formulario emergente (ver Figura 4.23) que permite tanto el registro de nuevos tipos de venta, como la edición de los ya existentes.

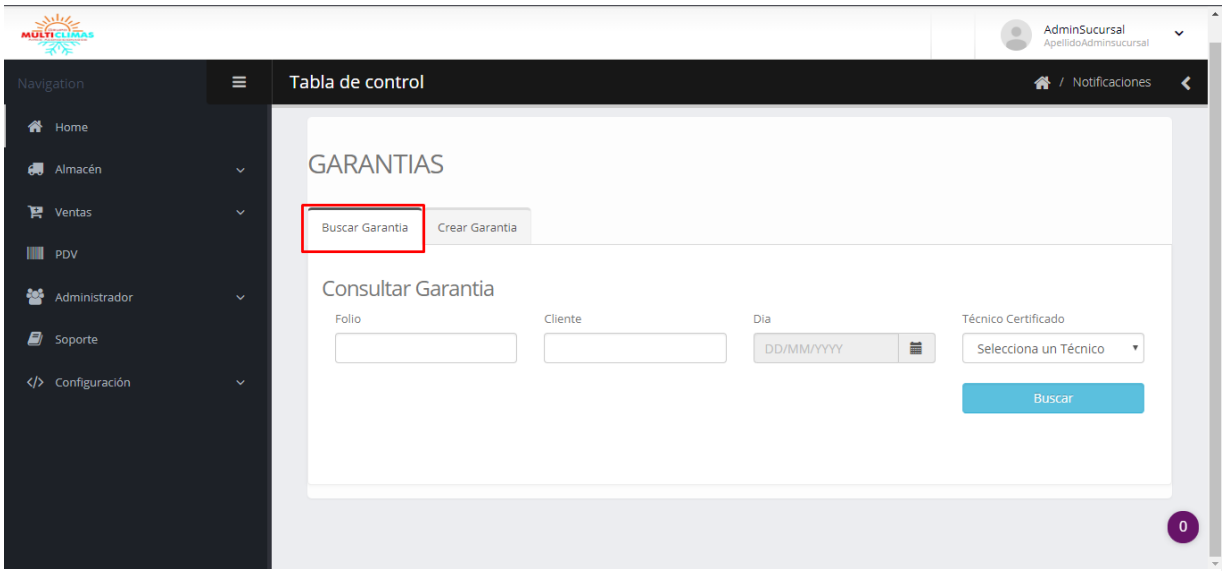

Figura 4.24: Pestaña de búsqueda de garantías.

La pestaña de búsqueda en el formulario de garantías (ver Figura 4.24) tiene como función devolver las garantías sobre productos defectuosos para hacerlas válidas.

Figura 4.25: Pestaña de creación de garantía.

La pestaña de creación en el formulario de garantías (ver Figura 4.25) tiene como finalidad validar la garantía a determinado producto que presente problemas.

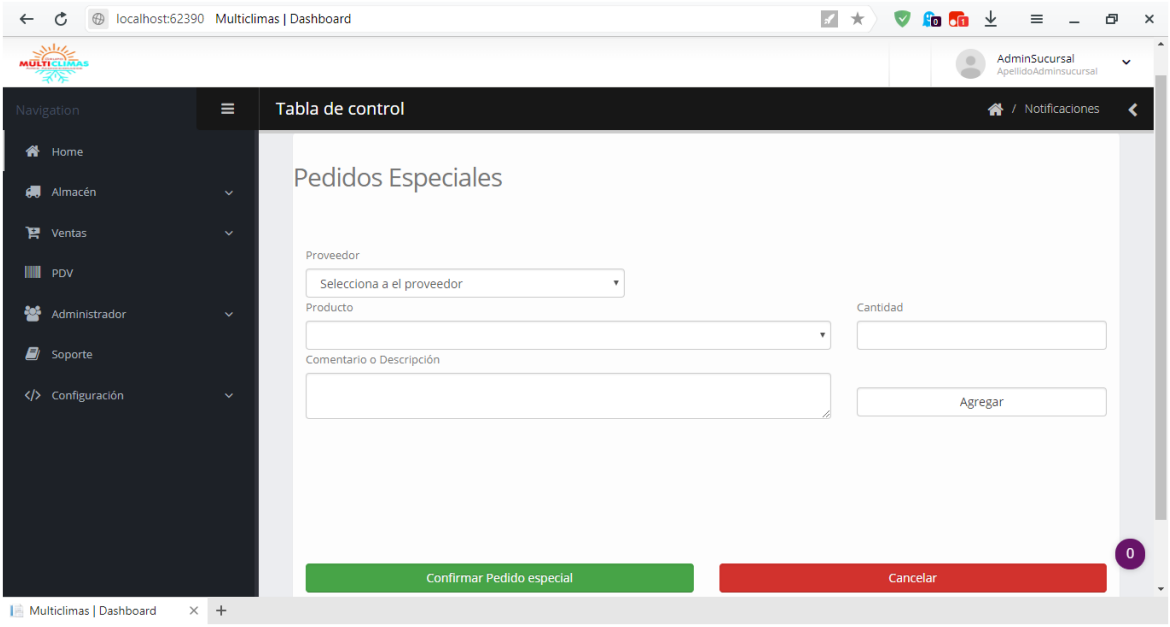

Figura 4.26: Formulario de pedidos especiales.

Se obtuvo un formulario (ver Figura 4.26) en el cual mediante convenios con el proveedor se realizan solicitudes de producto.

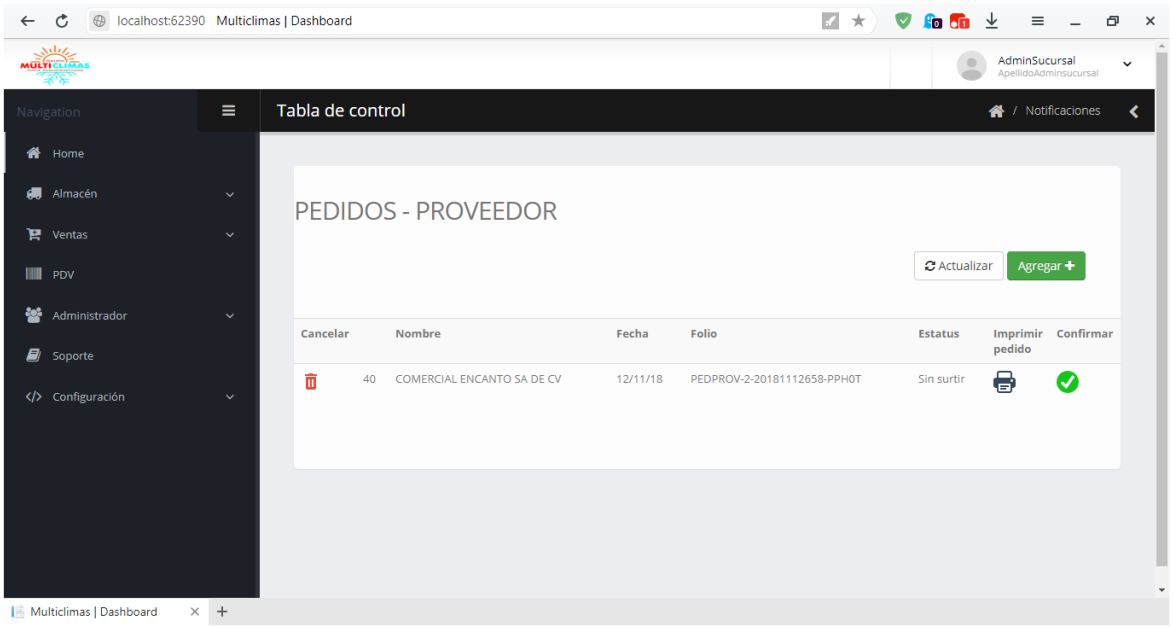

Figura 4.27: Formulario de pedidos a proveedores.

Se logró desarrollar un formulario de control (ver Figura 4.27) que permite gestionar solicitudes de producto a proveedores, en el cual es posible cancelar pedidos o confirmar pedidos, de igual manera es posible imprimir los documentos correspondientes.

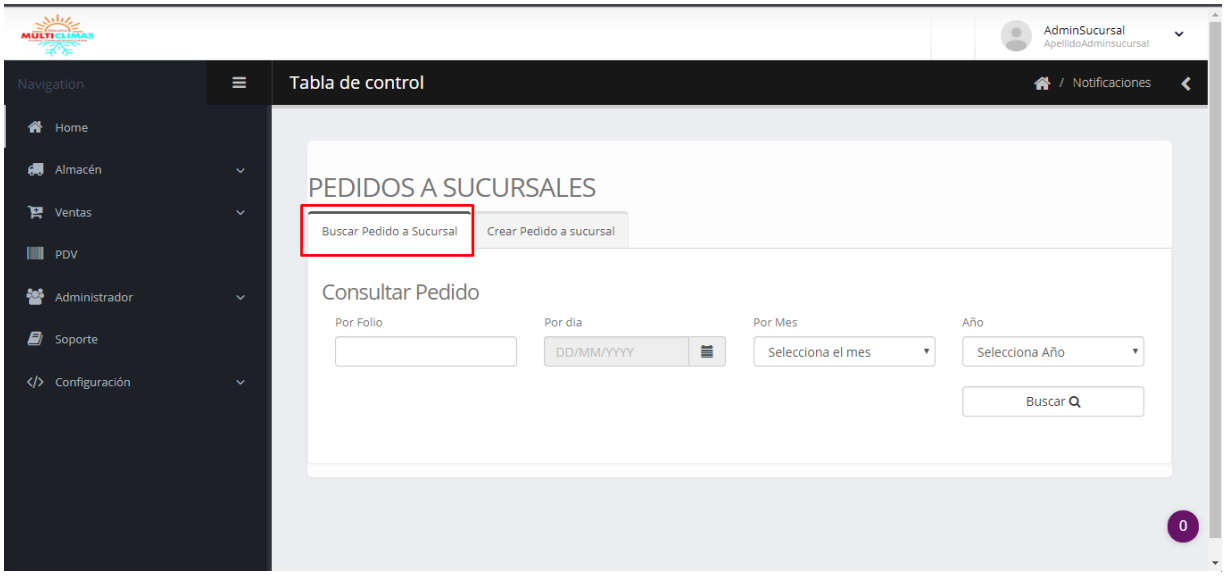

Figura 4.28: Pestaña de búsqueda de pedidos a sucursales.

Pestaña en formulario de pedidos a sucursales (ver Figura 4.28) que permite la búsqueda de solicitudes de producto realizadas entre sucursales pertenecientes a la empresa grupo Multiclimas.

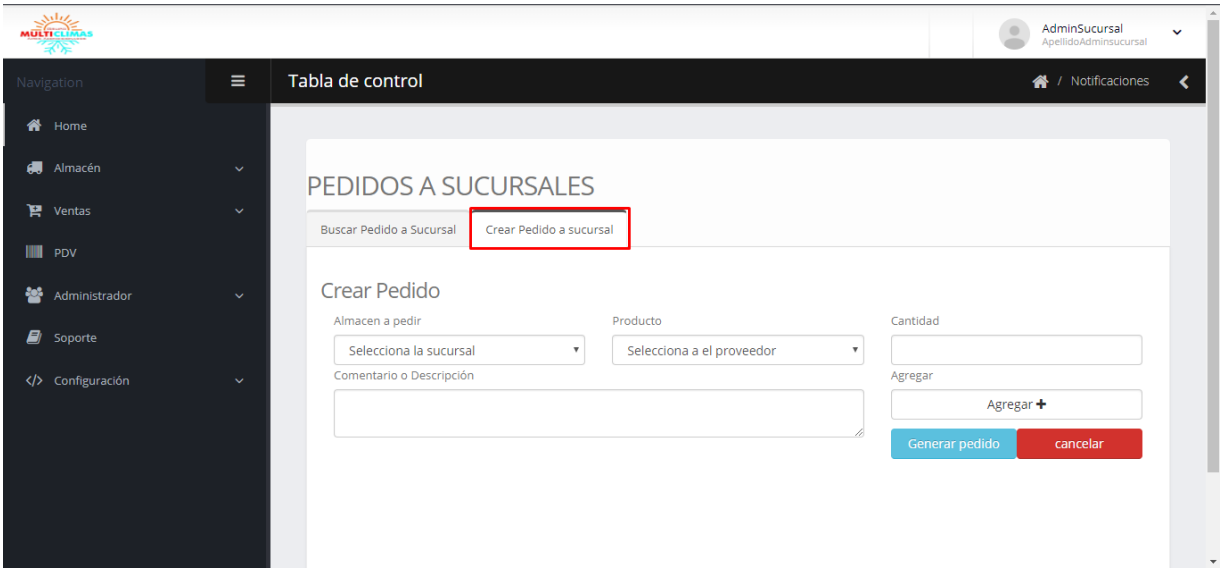

Figura 4.29: Pestaña de creación de pedidos a sucursales.

Pestaña en formulario de pedidos a sucursales (ver Figura 4.29) que permite generar solicitudes de producto a otras sucursales pertenecientes a Multiclimas.

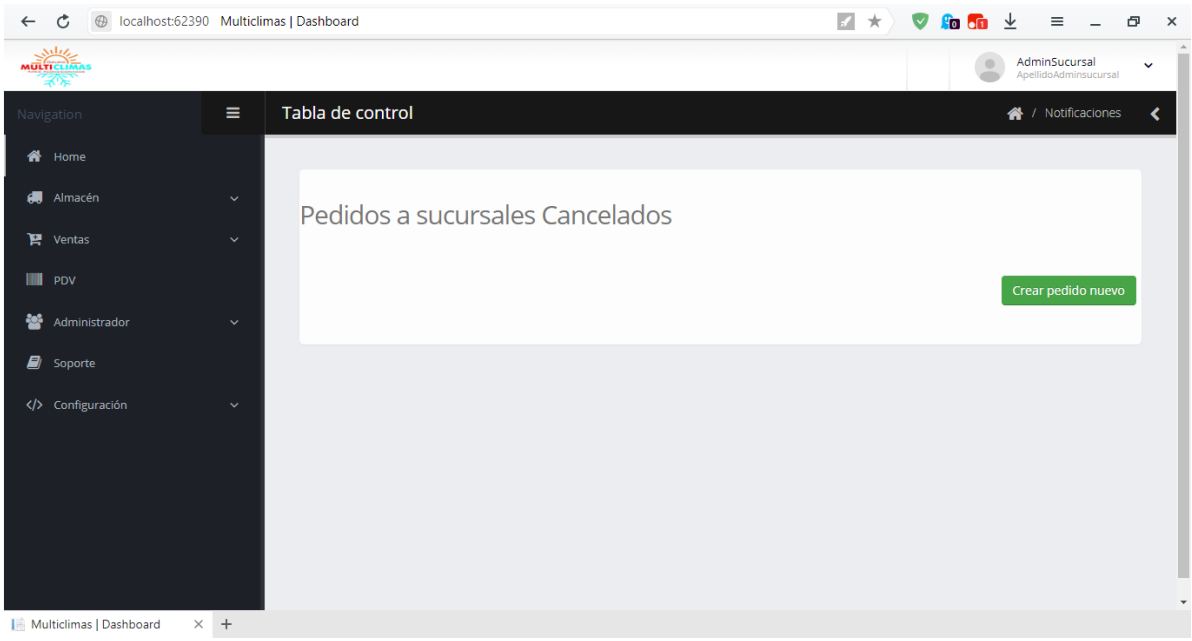

Figura 4.30: Formulario de pedidos cancelados.

Se obtuvo un formulario que permite visualizar los pedidos entre sucursales que fueron cancelados (ver Figura 4.30).

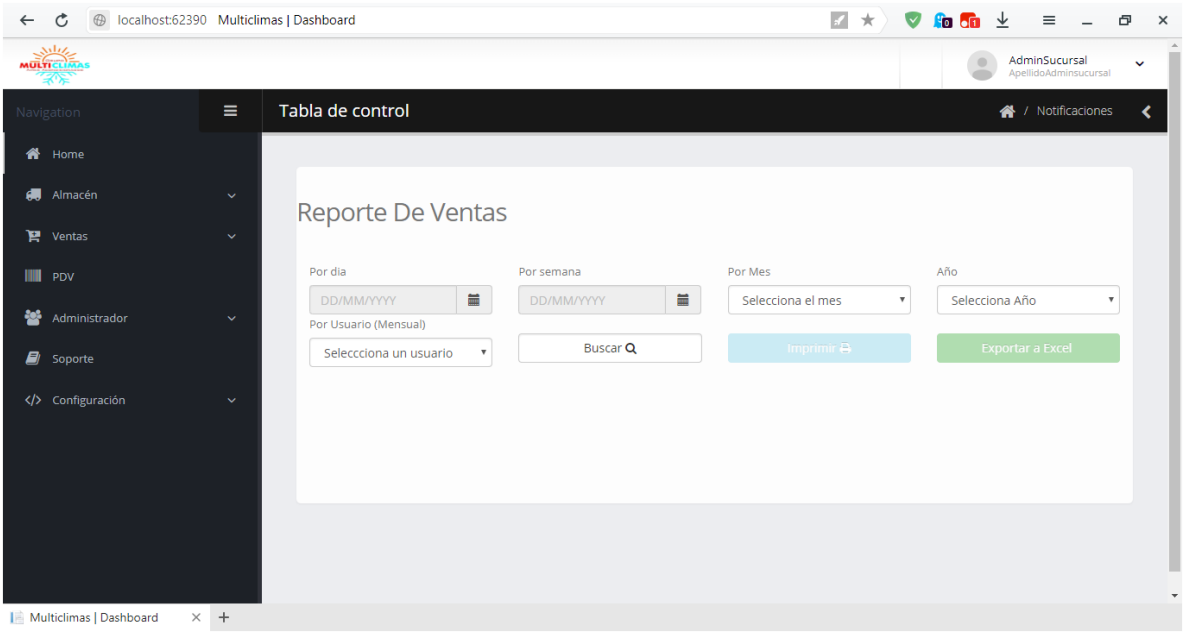

Figura 4.31: Formulario de reporte de ventas.

Se logró desarrollar un formulario que se muestra en la Figura 4.31 el cual permite la búsqueda de reportes de venta para posteriormente imprimirlos o exportarlos como hojas de cálculo.

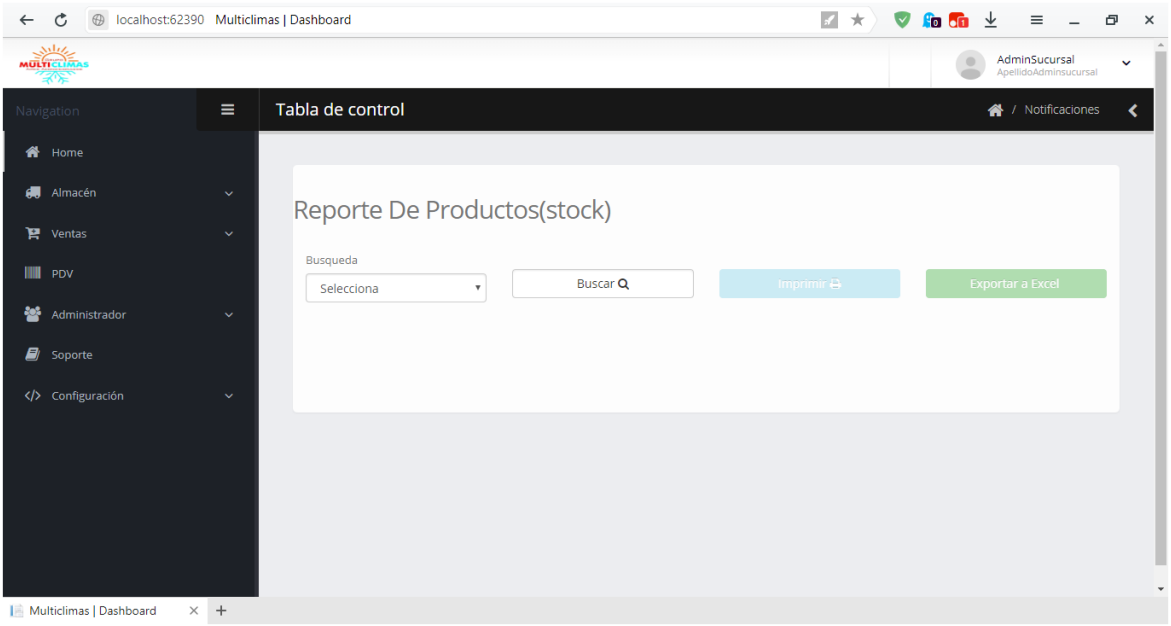

Figura 4.32: Formulario de reporte de productos en stock.

Se obtuvo un formulario (ver Figura 4.32) que permite generar un reporte de inventario para posteriormente imprimirlo o exportarlo como hoja de cálculo.

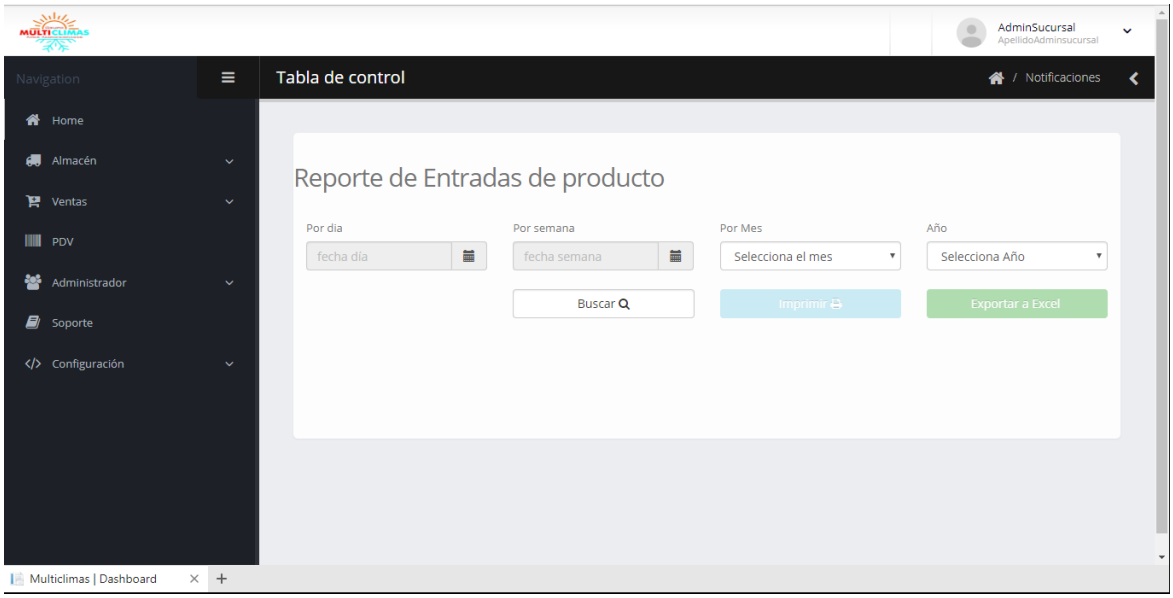

Figura 4.33: Formulario de reporte de entradas de producto.

Se desarrolló un formulario (ver Figura 4.33) que permite generar un reporte sobre entradas de producto al almacén, para posteriormente imprimirlo o exportarlo como hoja de cálculo.

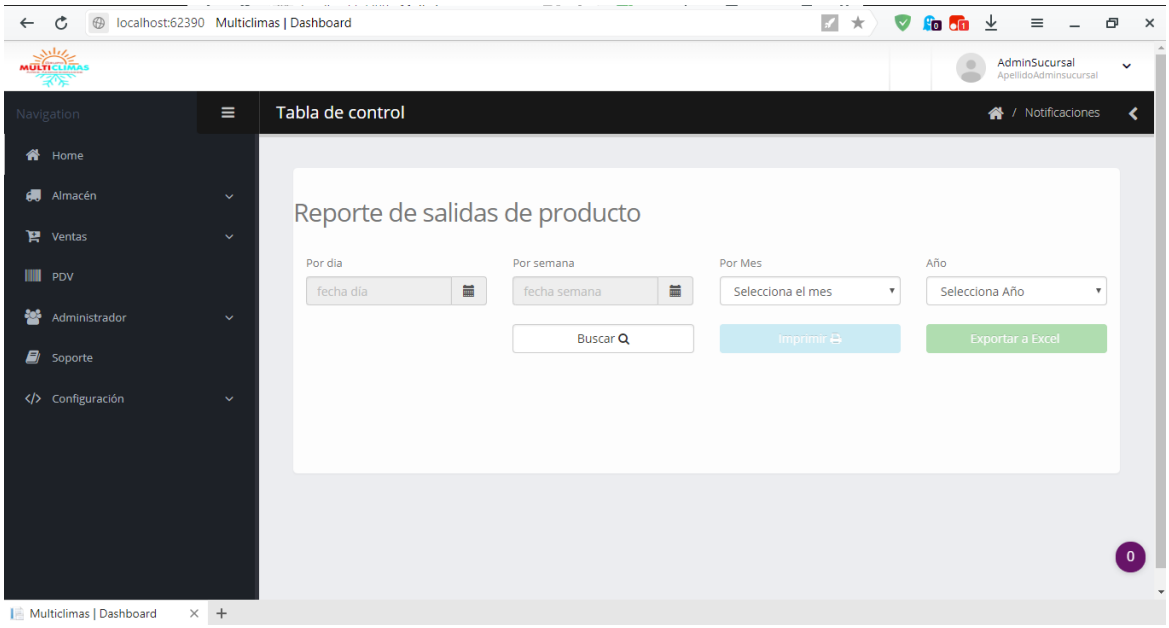

Figura 4.34: Formulario de reporte de salidas de producto.

Se obtuvo un formulario (ver Figura 4.34) que permite generar un reporte de salidas de producto, para posteriormente imprimirlo o exportarlo como hoja de cálculo.

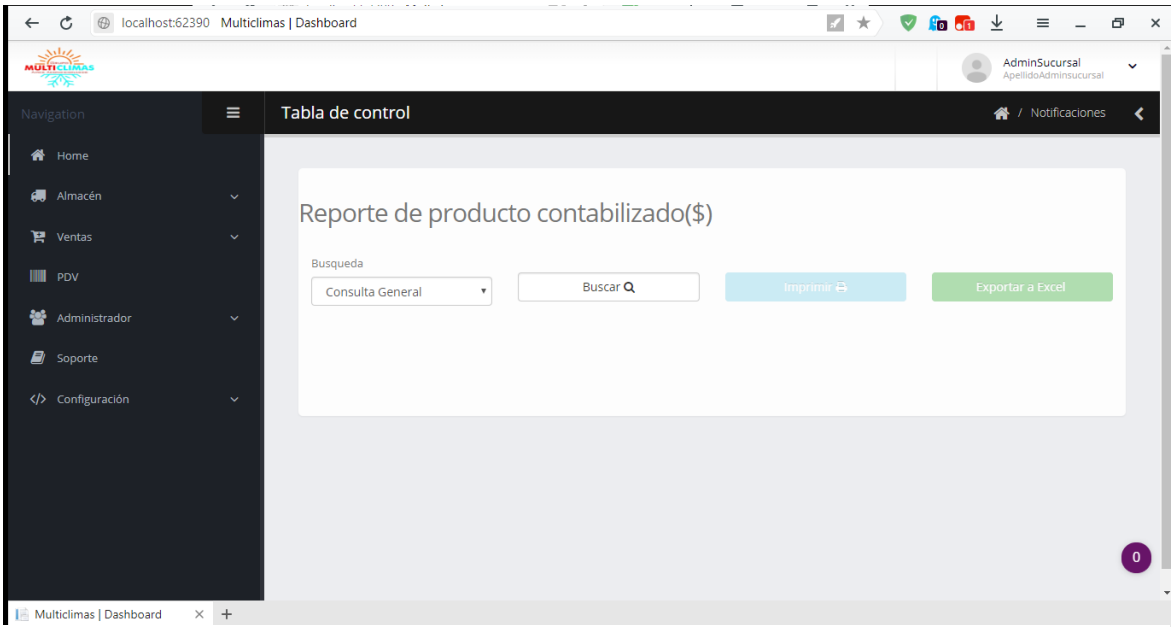

Figura 4.35: Formulario de reporte de inversión neta.

Se logró desarrollar un formulario que permite generar un reporte del producto contabilizado monetariamente, es decir, el monto invertido en el producto existente en inventario, para posteriormente imprimirlo o exportarlo como hoja de cálculo.

4.5 Evaluación del producto

El producto final obtenido tras el análisis y desarrollo previamente descritos representa un sistema robusto y completamente funcional el cual cumple satisfactoriamente los objetivos planteados.

Dicho sistema se diseñó con la particularidad de poseer un sencillo manejo de errores, además de contar con una alta escalabilidad, lo que permite realizar a futuro diversas correcciones, modificaciones o mejoras, sin mencionar la posibilidad de agregar nuevos módulos en el sistema para un completo funcionamiento.

Cada uno de los módulos que conforman al sistema general fueron desarrollados y adecuados de acuerdo a las necesidades y requerimientos especificados por el cliente, sin limitar la facilidad de operación por el usuario final.

Entre la gran variedad de funciones implementadas en el sistema, las más destacadas, son en primer lugar: la correcta gestión de almacén, dicha funcionalidad representa una reducción en tiempos de espera para completar las actividades relacionadas con el manejo de producto existente en almacén, conjuntamente se obtienen resultados correctos y precisos para llevar en un futuro un control ordenado del inventario registrado por almacén en cada sucursal.

También es importante destacar la correcta operatividad del módulo de ventas, dentro del cual se procesan las operaciones generadoras de utilidad para la empresa Grupo Multiclimas, dado que es posible efectuar una escrupulosa administración de la información registrada por cada cliente, proveedor e inclusive las transacciones que se llevan cabo tanto entre clientes como entre proveedores.

Colectivamente, la satisfactoria finalización del desarrollo del proyecto se debe a la estructura concreta de la base de datos, la cual mediante el modelo entidad relación permitió conservar integridad y consistencia en el flujo de información que existente entre el sistema y la base de datos.

4.6 Conclusiones

En base a los resultados obtenidos en el presente trabajo, es posible afirmar que los objetivos propuestos se alcanzaron satisfactoriamente.

El manejo del sistema representa una serie de ventajas tecnológicas frente al creciente y cambiante mercado meta presente en la actualidad.

Algunos de los beneficios adquiridos son los siguientes:

- La aplicación ERP hace posible llevar un correcto y estricto orden de los registros de inventario.
- Permite consultar información tanto de clientes como de proveedores sin importar la hora y lugar de acceso al sistema.
- Reduce los tiempos de espera en la realización de diversas operaciones, tales como ventas o adquisición de producto.
- Datos exactos y confiables en la creación de los distintos tipos de reportes que el sistema puede generar.
- Unificación y consistencia de la información que se registre en el sistema.
- Representa una mejora de servicio al cliente tanto en calidad como en eficiencia.

Por otro lado, se presentan diversos inconvenientes, algunos de los más importantes son los siguientes:

- Dado que es un sistema basad en web, se requiere en todo momento conexión a internet.
- El administrador del sistema es el único encargado de gestionar las cuentas del resto de usuario registrados en el sistema.

El correcto uso de la aplicación permitirá mantener una integridad y persistencia en los registros almacenados, de esta manera se facilita la gestión de información consultando datos relevantes, reduciendo así los tiempos de espera entre operaciones.

Cabe destacar que tanto la arquitectura con la que se diseñó el sistema, como su proceso de desarrollo de la aplicación, se llevaron a cabo con la finalidad de ser fácilmente escalable, por lo que es posible, además de realizar modificaciones y llevar a cabo corrección de posibles

errores que puedan presentarse, agregar nuevas funcionalidades que llegue a ser requeridas, al mismo tiempo es posible rediseñar módulos ya existentes, al igual que realizar mejoras en el rendimiento del sistema, volviéndolo aún más eficaz frente a los requerimientos que lleguen a presentarse con el paso del tiempo.

Bibliografía

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2017). Agile software development methods: Review and analysis. arXiv preprint arXiv:1709.08439.

Arias Chaves, M. (2005). La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. InterSedes: Revista de las Sedes Regionales, VI (10), 1-13.

Cadavid, A. N., Martínez, J. D. F., & Vélez, J. M. (2013). Revisión de metodologías ágiles para el desarrollo de software. Prospectiva, 11(2), 30-39.

Castro, J. (21 de octubre de 2014). Beneficios de un sistema de control de inventarios. Obtenido en la Red Mundial el 22 de septiembre de 2018, http://bit.ly/2QTigLh

Chiesa, F. (2004). Metodología para selección de sistemas ERP. Reportes técnicos en ingeniería del software, 6(1), 17-37.

Cobo, Á. (2005). PHP y MySQL: Tecnología para el desarrollo de aplicaciones web. Ediciones Díaz de Santos.

De la Torre Llorente, C., Castro, U. Z., Barros, M. A. R., & Nelson, J. C. (2010). Guía de Arquitectura N-Capas orientada al Dominio con .NET.

Figueroa, R. G., Solís, C. J., & Cabrera, A. A. (2008). Metodologías tradicionales vs. metodologías ágiles. Universidad Técnica Particular de Loja, Escuela de Ciencias de la Computación.

Forcier, J., Bissex, P., & Chun, W. J. (2008). Python web development with Django. Addison-Wesley Professional.

Gallardo López, D., & Pomares Puig, C. (2008). Programación orientada a objetos. Lenguajes y Paradigmas de Programación.

Gallego, M. T. (2012). Metodología Scrum. Universitat Oberta de Catalunya.

Gupta, P., & Govil, M. (2010). Spring Web MVC Framework for rapid open source J2EE application development: a case study. International Journal of Engineering Science and Technology, 2(6), 1684-1689.

Halterman, R. L. (2011). LEARNINGTO PROGRAMWITH PYTHON.

Henríquez, S. D. M., Huerta, H. V., & Grados, L. A. G. (2010). Programación en N capas. Revista de investigación de Sistemas e Informática, 7(2), 57-67.

Holland, C. P., & Light, B. (1999). A critical success factors model for ERP implementation. IEEE software, (3), 30-36.

Hong, K. K., & Kim, Y. G. (2002). The critical success factors for ERP implementation: an organizational fit perspective. Information & management, 40(1), 25-40.

IEEE Std 610.12-1990, "IEEE Standard Glossary of Software Engineering Terminology"

Joyanes Aguilar, L. (1996). Programación orientada a objetos (No. 004.652. 5). McGraw-Hill Interamericana.

Kotonya, G., & Sommerville, I. (1998). Requirements engineering: processes and techniques. Wiley Publishing.

Lerdorf, R., Tatroe, K., & MacIntyre, P. (2006). Programming Php. " O'Reilly Media, Inc.".

Liskov, B., & Guttag, J. (1986). Program development in JAVA: abstraction, specification, and object-oriented design. Pearson Education.

Luján Mora, S. (2002). Programación de aplicaciones web: historia, principios básicos y clientes web. Editorial Club Universitario.

Mahnic, V., & Drnovscek, S. (2005, June). Agile software project management with scrum. In EUNIS 2005 Conference-Session papers and tutorial abstracts (p. 6).

Mateu, C. (2012). Desarrollo de aplicaciones web.

Matsumoto, Y., & Ishituka, K. (2002). Ruby programming language.

Mikkonen, T., & Taivalsaari, A. (2007). Using JavaScript as a real programming language.

Nielson, J., Williamson, C., & Arlitt, M. (2008, October). Benchmarking modern web browsers. In 2nd IEEE Workshop on Hot Topics in Web Systems and Technologies.

Odoo, S.A.(2018, 10 15). OpenERP Documentation v6.1. Retrieved from Odoo: https://doc.odoo.com/v6.1/index.html/

Palazón, F. J. (2018, 03 02). Microsoft Dynamics 365. Retrieved from byteTI: https://www.revistabyte.es/analisis-byte-ti/microsoft-dynamics-365/

Palazón, F. J. (2018, 06 11). Aqua eBS 2018. Retrieved from byteTI: https://www.revistabyte.es/analisis-byte-ti/erp-aqua-ebs-2018/

Palazón, F. J. (2018, 06 11). Oracle ERP Cloud 2018. Retrieved from byteTI: https://www.revistabyte.es/analisis-byte-ti/oracle-erp-cloud-2018/

Plekhanova, J. (2009). Evaluating web development frameworks: Django, Ruby on Rails and CakePHP. Institute for Business and Information Technology.

Richter, J. (2002). Applied Microsoft. NET framework programming (Vol. 1). Redmond: Microsoft Press.

Rodríguez, C. V. (2011). Paradigmas de programación. Curso 2012-13.

Ruby, S., & Thomas, D. (2009). Agile web development with rails. Raleigh, NC: Pragmatic Bookshelf.

Samtani, G., & Sadhwani, D. (2002). Web services and application frameworks (. NET and J2EE). In Web Services Business Strategies and Architectures (pp. 273-289). Apress, Berkeley, CA.

Sanders, W., & Cumaranatunge, C. (2007). ActionScript 3.0 Design Patterns: Object Oriented Programming Techniques. " O'Reilly Media, Inc.".

Smith, B. (2011). Object-oriented programming. In AdvancED ActionScript 3.0: Design Patterns (pp. 1-25). Apress.

Sommerville, I. (2005). Ingeniería del Software, Séptima edición, México DF, Editorial Pearson.

Sommerville, I. (2007). Software engineering (pp. I-XXIII). Addison-wesley.

Sommerville, I., & Sawyer, P. (1997). Requirements engineering: a good practice guide. John Wiley & Sons, Inc.

Tucker, A. B. 1., & Noonan, R. 1. Lenguajes de programación: Principios y paradigmas. Madrid: McGraw-Hill/Interamericana de España

Van Rossum, G. (2007, June). Python Programming Language. In USENIX Annual Technical Conference (Vol. 41, p. 36)

Vera, Á. B. (2006). Implementación de sistemas ERP, su impacto en la gestión de la empresa e integración con otras TIC. CAPIC REVIEW, (4), 3.

Williams, G. (4 de octubre de 2006). Controlling Your Inventory. Obtenido en la Red Mundial el 22 de septiembre de 2018, http://bit.ly/2O5B9Me