EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Tecnológico Nacional de México

Instituto Tecnológico Superior de Guasave

Tesis de Licenciatura

Mejoras a la gestión del proceso de producción de una empresa acuícola mediante tecnologías computacionales

presentada por

Miguel Alonso López Peñuelas

como requisito para la obtención del grado de

Licenciado en Sistemas Computacionales

Director de tesis

M.C. Raúl Loredo Medina

Codirector de tesis

ING. Graciela Lugo Rubio

Guasave, Sinaloa, México. Febrero de 2025.

Instituto Tecnológico Superior de Guasave

RESUMEN

Este trabajo tiene como objetivo mejorar el proceso de producción de una empresa acuícola a través del desarrollo e implementación de una aplicación móvil y de escritorio, ambas conectadas a una base de datos implementada en una Raspberry Pi. Esta solución hace más eficiente la captura, almacenamiento y consulta de datos, minimizando errores humanos y reduciendo los tiempos de búsqueda.

Los resultados muestran una notable disminución en los tiempos de captura y almacenamiento de datos, una generación más ágil de reportes en formatos PDF y Excel, y un acceso inmediato a la información en tiempo real. Además, se ha fortalecido la integridad y trazabilidad de los datos, lo que facilita la toma de decisiones informadas y confiables.

En conclusión, la adopción de tecnologías computacionales no solo optimiza las operaciones diarias, sino que también mejora la competitividad y sostenibilidad de la empresa.

Palabras clave: Digitalización, Optimización de procesos, Aplicación móvil, Raspberry Pi, Gestión de datos.

ABSTRACT

This work aims to improve the production process of an aquaculture company through the development and implementation of a mobile and desktop application, both connected to a database implemented on a Raspberry Pi. This solution makes data capture, storage, and retrieval more efficient, minimizing human errors and reducing search times.

The results show a significant decrease in data capture and storage times, a more agile generation of reports in PDF and Excel formats, and immediate access to real-time information. Additionally, data integrity and traceability have been strengthened, facilitating informed and reliable decision-making.

In conclusion, the adoption of computational technologies not only optimizes daily operations but also improves the company's competitiveness and sustainability.

Keywords: Digitization, Process Optimization, Mobile Application, Raspberry Pi, Data Management.

INDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En la actualidad, las empresas acuícolas enfrentan el desafío de gestionar grandes volúmenes de datos generados durante sus procesos de producción. La correcta administración de esta información es esencial para la toma de decisiones estratégicas y la optimización de recursos. Sin embargo, muchas empresas aún dependen de métodos manuales, como el uso de formatos en papel, lo que no solo incrementa la posibilidad de errores humanos, sino que también ralentiza el acceso y análisis de la información.

Ante esta problemática, el presente plantea mejoras para la gestión de procesos mediante tecnologías computacionales. Esta mejora permitirá digitalizar la captura y gestión de datos, reduciendo los tiempos de respuesta, minimizando errores y facilitando la consulta de información en tiempo real.

El documento se estructura de la siguiente manera: en primer lugar, se presentan los antecedentes y fundamentos teóricos que sustentan la propuesta. Posteriormente, se describe la metodología utilizada en el desarrollo del sistema, seguida de los resultados obtenidos tras su implementación. Finalmente, se incluyen conclusiones y recomendaciones para futuras mejoras del sistema.

PLANTEAMIENTO DEL PROBLEMA

En una acuícola en la ciudad de Guasave, se cuenta con el proceso de control de calidad de los productos por lo que para hacer esto, se cuenta con diversos formatos físicos como área de frezado, área de proceso, temperatura de producto, temperatura de agua y producto y detección de metales para la recolección de datos, este proceso de recolección se hace de manera manual, donde el/la responsable se traslada de un área a otra con los documentos de manera expuesta. Este método no solo es propenso a errores humanos, sino que también a que la información se pierda, se dañen los documentos, se mojen o cuando sea necesario obtenerlos exista pérdida de tiempo, equivocaciones en los documentos tomados y se vuelva a buscar lo cual vuelve ineficiente el trabajo, así como, también se vuelven tardías la toma de decisiones o no se llegan a tomar decisiones completas o correctas a base de documentos con información dañada o perdida.

La oportunidad que se presenta es mejorar todo este proceso mediante el uso de las tecnologías informáticas para que permitan digitalizar y organizar la información en tiempo real. Con esta solución, la empresa podrá acceder fácilmente a los datos y tener un mayor control sobre sus operaciones, mejorando así su productividad y calidad.

JUSTIFICACIÓN

Actualmente la empresa utiliza formatos de papel llenados a mano, lo cual puede ser propenso a una alta probabilidad de confusión de datos requeridos, una alta probabilidad a perdida de información o de sufrir algún daño como derrame de líquidos o caigan al suelo y se ensucien o lleguen a rasgarse lo cual provoque que se vuelva complicada o imposible de leer la información o los datos en los documentos, cabe destacar que el proceso actual puede volver tardía la toma de decisiones o la búsqueda de información, así como también un documento mal organizado o guardado incorrectamente puede conllevar a una confusión de documentos ya sea que se deba a un formato de fecha diferente al necesario o de un área no buscada.

Al adoptar las mejoras de tecnologías informáticas para la gestión de información, llegara a mejorar el registro, almacenamiento y búsqueda de la información, la gestión de los datos puede ser en tiempo real, además de, dar una mayor seguridad a los datos evitando así de esta manera las principales preocupaciones o las más básicas sobre los documentos en papel.

Este avance moderniza las operaciones, optimiza la toma de decisiones y adaptada a las tecnologías actuales y próximas por venir. No realizar este proyecto mantendría un sistema de gestión atrasado trabajo, limitando la eficiencia y el avance futuro de la empresa.

HIPÓTESIS

La implementación de un software de escritorio junto con una aplicación móvil, ambos conectados a una base de datos alojada en una Raspberry Pi, representa una solución innovadora y eficiente para la gestión de los procesos en una empresa acuícola. Este sistema permite digitalizar la captura, almacenamiento y consulta de información clave, optimizando las actividades operativas y reduciendo significativamente la dependencia de formatos en papel.

Al automatizar estos procesos, se minimizarían los errores humanos derivados de registros manuales y se mejora la precisión de los datos recopilados. Además, al centralizar la información en una base de datos accesible desde diferentes dispositivos, se agilizan los tiempos de consulta y análisis, permitiendo una toma de decisiones más informada y oportuna. Esto no solo contribuye a una mayor eficiencia operativa, sino que también facilita la trazabilidad y el monitoreo de los parámetros críticos del proceso productivo, aspectos fundamentales para garantizar la calidad y sostenibilidad en la acuicultura.

OBJETIVOS

Objetivo general

Optimizar la gestión del proceso de producción en una empresa acuícola mediante el desarrollo e implementación de soluciones tecnológicas, mejorando la eficiencia en el manejo de la información y facilitando la toma de decisiones.

Objetivos específicos

- **Tomar y analizar los requisitos** necesarios para identificar las necesidades y estructura para su desarrollo.
- **Desarrollar una aplicación móvil** que permita el registro eficiente de datos en las áreas de producción de la empresa acuícola.
- **Implementar una base de datos en una Raspberry Pi** para almacenar y consultar los datos registrados por la aplicación móvil.
- **Desarrollar una aplicación de escritorio** para la generación automática de reportes en formatos PDF y Excel.
- **Analizar el impacto de la mejora** de la solución tecnológica en la eficiencia de los procesos.

MARCO TEÓRICO

Introducción

Para mejorar la eficiencia en cualquier proceso, es fundamental contar con fundamentos teóricos que brinden una idea del desarrollo de soluciones efectivas. En este trabajo, el marco teórico cumple la función de sentar las bases conceptuales necesarias para entender la problemática de la gestión de datos en una empresa acuícola y cómo las tecnologías computacionales pueden ofrecer una solución eficiente.

Actualmente, muchas empresas acuícolas siguen dependiendo de métodos manuales para registrar información clave de sus procesos de producción. Este enfoque, aunque funcional, tiene limitaciones evidentes, como, por ejemplo: errores humanos, la pérdida, y el daño de datos, y la dificultad para acceder a la información en tiempo real. Aquí es donde la tecnología entra en juego, ofreciendo herramientas que permiten digitalizar, organizar y analizar los datos de manera rápida y precisa.

Este marco teórico es clave porque permite comprender la importancia de cada componente del sistema propuesto. Se explorarán conceptos como la gestión eficiente de datos, el uso de tecnologías de bajo costo como la Raspberry Pi, y el impacto de la digitalización en la toma de decisiones. Todo esto no solo ayudará a justificar la solución planteada, sino que también demostrará cómo la tecnología puede transformar la manera en que las empresas acuícolas administran su información y optimizan sus procesos.

Bases de datos con MySQL/MariaDB

¿Qué es una Base de datos?

Una base de datos es un apartado para almacenar datos de gran tamaño se reconoce generalmente por su clásico icono que aparenta un cilindro dividido en 3 secciones (Véase figura 1), explicado más claramente, es una herramienta para recopilar y organizar información. Las bases de datos pueden almacenar información sobre personas, productos, pedidos u otras cosas. Muchas bases de datos comienzan como una lista en una hoja de cálculo o en un programa de procesamiento de texto. A medida que la lista aumenta su tamaño, empiezan a aparecer redundancias e inconsistencias en los datos. Cada vez es más difícil comprender los datos en forma de lista y los métodos de búsqueda o extracción de subconjuntos de datos para revisión son limitados. Una vez que estos problemas empiezan a aparecer, es una buena idea transferir los datos a una base de datos creada por un sistema de administración de bases de datos (Soporte técnico de Microsoft, s/f).

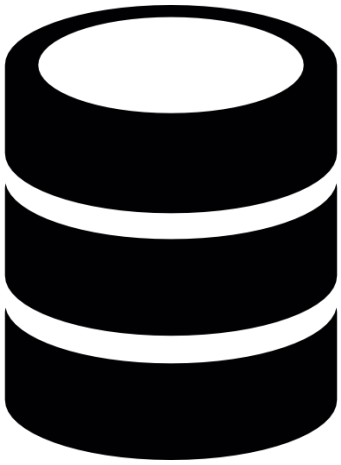


Figura 1: Icono representativo *de las* bases *de datos*

¿Qué es un motor de base de datos?

Un motor de base de datos es el agente principal en los sistemas de gestión de datos, Su papel fundamental es gestionar la interacción entre las aplicaciones y los datos almacenados, proporcionando una infraestructura sólida para la gestión de información en entornos diversos, desde aplicaciones empresariales hasta plataformas web. Los motores de bases de datos son la fuerza motriz que le permite a las organizaciones gestionar grandes volúmenes de datos de manera eficiente y escalable, con el fin de asegurar la integridad y disponibilidad de la información crítica para la toma de decisiones y el funcionamiento diario de sistemas informáticos (Navarro, 2024).

MySQL y MariaDB

MySQL

MySQL es la base de datos de código abierto más popular del mercado, además de tener un logo único e inconfundible (Véase figura 2). MySQL potencia muchas de las aplicaciones más accesibles, como Facebook, Twitter, Netflix, Uber, Airbnb, Shopify y Booking.com, además la base de datos MySQL es un sistema cliente/servidor que consta de un servidor SQL multihilo que admite diferentes Back-End, varios programas y bibliotecas cliente diferentes, herramientas administrativas y una amplia gama de interfaces de programación de aplicaciones (oracle, s/f-b).

Figura 2: Logo de MySQL

MariaDB

MariaDB Server es un sistema de gestión de bases de datos relacionales de código abierto (Véase figura 3). Es uno de los servidores de bases de datos más populares del mundo, con usuarios notables como Wikipedia, WordPress.com y Google. MariaDB es el predecesor de MySQL después de ser comprado por Oracle en 2009, además tiene una gran compatibilidad con MySQL, PostgREST, MongoDB y Oracle, sobre todo MySQL ya que la mayoría de las aplicaciones populares que utilizan MySQL funcionan sin problemas con MariaDB (MariaDB.org, 2023).

Figura 3: Logo de MariaDB

Uso de MySQL/MariaDB

Funciones en MySQL/MariaDB

En la tabla 1 se puede apreciar la sintaxis de algunas funciones básicas para el uso de MySQL o MariaDB.

Tabla 1: Sintaxis de las funciones básicas en MySQL/MariaDB

Función	Sintaxis
Crear base de datos	Create database <Nombre de la base de datos>;
Crear tabla	Create Table <Nombre de la tabla> (<Nombre del campo> <Tipo de dato>); Si se desea agregar más campos se pone una ‘,’ al final del campo puesto y se escribe el siguiente.
Mostrar	Select <Campos> From <Tabla> Se usa * en lugar de campos si deseas que te devuelva información de todos los campos. En caso de querer que se devuelvan datos específicos se usa Where <Condicional>
Insertar	Insert into <tabla> (<Campos separados por ‘,’>) Values (<Valor de los campos, separados por ‘,’>) Ejemplo: INSERT INTO clientes (id, nombre, email, telefono) VALUES (1, 'Pedro', 'pedro@gmail.com', '123456789');
Eliminar	DELETE FROM <tabla> WHERE <condición>; El where no es obligatorio, pero en caso de no estar se eliminará toda la tabla.
Actualizar	Update <Tabla> Set <Campo> = ‘<Nuevo valor>’ where <Condicional> En caso de requerir cambiar varios campos se separa con ‘,’ entre cada cambio, el where es opcional.
Condicional	La condicional se utiliza entre un campo y un valor, por ejemplo: edad = 15, Nombre = ‘Javier’ o Fecha = ‘10/05/2015’.

Tipos de datos MySQL/MariaDB

Dentro de MySQL/MariaDB existen diferentes tipos de datos para los datos que serán registrados dentro de las tablas de la base de datos, a continuación estos son los diferentes tipos de datos (Véase tablas 2-8) (Platzi, s/f).

Tabla 2: Tipo de datos numéricos enteros

Tipo	Tamaño	Valor mínimo	Valor máximo
INTEGER	32 bits	-2^31	2^31 - 1
UNSIGNED INTEGER	32 bits	0	2^32 - 1
BIT (M)	M	1 bit	64 bits
TINYINT	8 bits	-2^7	2^7 - 1
UNSIGNED TINYINT	8 bits	0	2^8-1
SMALLINT	16 bits	-2^15	2^15 - 1
UNSIGNED SMALLINT	16 bits	0	2^16 - 1
MEDIUMINT	24 bits	-2^23	2^23 - 1
UNSIGNED MEDIUMINT	24 bits	0	2
BIGINT	64 bits	-2^63	2^63 - 1
UNSIGNED BIGINT	64 bits	0	2^64 - 1

Tabla 3: Tipo de datos Numéricos reales

Tipo	Tamaño	Valor mínimo	Valor máximo
FLOAT	32 bits	-3.402823466E+38 / 1.175494351E-38	-1.175494351E-38 / 3.402823466E+39
DOUBLE	64 bits	-1.7976931348623157E+308 / 2.2240738585072014E-308	-2.2240738585072014E-308 / 1.7976931348623157E+308

Tabla 4: Tipo de datos de texto

Tipo	Tamaño mínimo	Tamaño máximo	Descripción
CHAR	0	255	Cadena de texto rellena con espacios a la derecha hasta cumplir el tamaño configurado
VARCHAR	0	2^16 *	El tamaño máximo puede variar de acuerdo al charset configurado. El tamaño almacenado es variable
BINARY	0	255	Similar a CHAR, pero almacena cadenas binarias en lugar de caracteres
VARBINARY	0	2^16 *	Similar a VARCHAR, pero almacena cadenas binarias en lugar de caracteres
TINYBLOB	0	2^8 - 1	Almacena datos binarios
TINYTEXT	0	2^8 - 1	Almacena caracteres
BLOB	0	2^16 - 1	Almacena datos binarios
TEXT	0	2^16 - 1	Almacena caracteres
MEDIUMBLOB	0	2^24 - 1	Almacena datos binarios
MEDIUMTEXT	0	2^24 - 1	Almacena caracteres
LONGBLOB	0	2^32 - 1	Almacena datos binarios
LONGTEXT	0	2^32 - 1	Almacena caracteres
ENUM	2^3 - 1	2^8 - 1	Permite contener una cadena de caracteres de una lista predefinida
SET	2^3 - 1	2^64 - 1	Similar a ENUM, pero permite contener uno o más valores, hasta 64.

Tabla 5: Tipos de datos de Fechas

Tipo	Valor mínimo	Valor máximo	Descripción
DATE	1000-01-01	9999-12-31	El formato de almacenamiento es YYYY-MM-DD
DATETIME	1000-01-01 00:00:00.000000	9999-12-31 23:59:59.999999	La precisión de los segundos fraccionales es configurable, por defecto es 0
TIMESTAMP	1970-01-01 00:00:01.000000	2038-01-19 03:14:07.999999	Utiliza el EPOCH de Unix. El valor 0 está reservado para representar 0000-00-00 00:00:00
TIME	-838:59:59.000000	838:59:59.000000	
YEAR	1901	2155	El formato de almacenamiento es YYYY. El valor nulo es representado por 0000

Tabla 6: Alias de tipos de datos

Tipo	Alias de	Descripción
BOOL	TINYINT (1)	Un valor de 0 es considerado falso, los demás son considerados verdaderos
BOOLEAN	TINYINT (1)	Un valor de 0 es considerado falso, los demás son considerados verdaderos
SERIAL	BIGINT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE	N/A
DEC	DECIMAL	N/A
FIXED	DECIMAL	N/A
DOUBLE_PRECISION	DOUBLE	N/A
REAL	DOUBLE	N/A

Tabla 7: Tipo de dato Geoespaciales

Tipo	Descripción
POINT (x, y)	Una columna especificando coordenadas
LINESTRING (x1 y1, x2 y2, …)	Una columna especificando una línea. Cada par de coordenadas está separado por una coma
POLYGON ((x1 y1, x2 y2, …), (x1 y1, x2 y2, …))	Una columna especificando un polígono.
MULTIPOINT	Una columna especificando diferentes puntos. A diferencia de LINESTRING, estos no forman una línea.

Tabla 8: Tipo de dato Json

Tipo	Descripción
JSON	Almacena documentos de JSON en la columna

Raspberry pi

¿Qué es una Raspberry pi?

La Raspberry Pi es una computadora de bajo costo y con un tamaño compacto, del porte de una tarjeta de crédito, puede ser conectada a un monitor de computador o un TV, y usarse con un mouse y teclado estándar. Es un pequeño computador que correo un sistema operativo Linux capaz de permitirle a las personas de todas las edades explorar la computación y aprender a programar lenguajes como Scratch y Python. Es capaz de hacer la mayoría de las tareas típicas de un computador de escritorio, desde navegar en internet, reproducir videos en alta resolución, manipular documentos de ofimática, hasta reproducir juegos (Paguayo, 2022).

Usos para la Raspberry pi

En cuanto al uso que se le puede dar a la RPi, existen diversos trabajos realizados en los últimos años, entre ellos se pueden mencionar aplicaciones dedicadas a la Domótica, Monitoreo para Seguridad, Streaming de Video/Audio, Estación meteorológica, Agro, salud entre otras (Rodriguez et al., 2017).

Existe una gran variedad de usos para la Raspberry Pi ya que al ser prácticamente una computadora y al ser muy portátil, económica y de fácil utilización la hace una herramienta muy eficiente como servidor donde se pueden comunicar y consultar datos las aplicaciones dentro de la misma red (Véase figura).

Figura 4: Uso de Raspberry pi como servidor

Aplicaciones móviles para la gestión de datos

¿Qué son las aplicaciones móviles?

Dentro del mundo moderno algo como los dispositivos móviles es algo tan común como autos en la calle, incluso los jóvenes y niños gozan de uno para su comunicación o mero entretenimiento, era lógico que tras la aparición de semejantes aparatos también vendrían nuevas oportunidades de avance, como serían las mismas aplicaciones para esos dispositivos, también conocidas como **aplicaciones móviles**; estas se encuentran dentro del margen de tabletas, teléfonos inteligentes y entre otros dispositivos que tienen la facilidad de movilidad con el usuario, que a diferencia de las computadoras de escritorio y portátiles que necesitan más espacio, algunos ejemplos de aplicaciones móviles serian:

Las redes sociales

Estas aplicaciones tienen la tarea o mejor dicho están diseñadas para uso exclusivo de comunicarse o estar comunicadas con otras personas compartiendo textos, imágenes, videos, tu vida personal, cosas que te pasan día a día, etc. Tales aplicaciones como las más conocidas como serian: Facebook, Instagram, YouTube y WhatsApp (Véase figura 5).

Figura 5: Aplicaciones móviles - redes sociales

Los Videojuegos

Estas aplicaciones son para único y mero entretenimiento dentro del mundo de la tecnología pero que son muy buenas para pasar el tiempo pero que son una muestra de las posibilidades graficas en las aplicaciones para móviles.

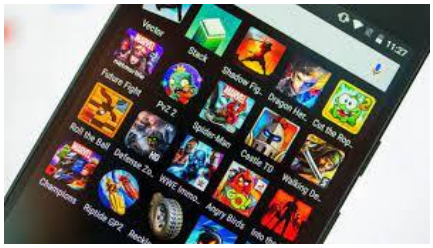


Figura 6: Aplicaciones móviles - videojuegos

Aplicaciones Bancarias

Dentro del mundo empresarial el tiempo es oro y los bancos supieron aprovechar la falta de tiempo que tienen los inversionistas y los mismos trabajadores brindándoles la facilidad de hacer transferencias, revisar su saldo, comprobar movimientos incluso mantener un token dinámico para más seguridad en sus compras en línea, de esta forma las cajas están más libres, y las personas no necesitan ir hasta el banco para saber cuánto saldo tienen, de igual manera se evita la utilización de papel para cada ocasión que se requiere saber el informe del estado de cuenta, ya que la aplicación puede consultarse, algunos bancos que cuentan con su aplicación móvil son: el nuevo banco del bienestar o BBVA.

Figura 7: Aplicaciones móviles - Banco Bienestar

Figura 8: Aplicaciones móviles - Banco BBVA

¿Qué es la gestión de datos?

La gestión de datos consiste en recopilar, mantener y utilizar datos de manera segura, eficiente y rentable. El objetivo de la gestión de datos es ayudar a las personas, las organizaciones y las cosas conectadas a optimizar el uso de los datos dentro de los límites de las políticas y normativas, para que puedan tomar decisiones y tomar medidas que maximicen el beneficio para la empresa (oracle, s/f-a).

Los creadores de aplicaciones de Android facilitan la gestión de bases de datos incorporando un enfoque visual para el diseño y la integración de bases de datos. Esto significa que los usuarios pueden crear estructuras de datos, definir relaciones y configurar ajustes mediante una interfaz gráfica de usuario, evitando la necesidad de conocimientos profundos de codificación (Gonzalez, 2023).

Ventajas del uso de tecnologías computacionales en una empresa

En una era donde la tecnología es el pilar de cientos de actividades empresariales, la informática se ha convertido en un elemento clave para el éxito. Desde la mejora de la eficiencia operativa hasta la facilitación de la toma de decisiones estratégicas, la informática ofrece numerosas ventajas que pueden transformar la forma en que los negocios operan y compiten en un mercado globalizado. Implementar sistemas informáticos permite a las empresas optimizar sus procesos operativos, desde la gestión de inventarios hasta la relación con los clientes, mejorando la eficiencia y reduciendo costos (Godoy, 2024).

En la era digital actual, la tecnología está revolucionando la manera en que las empresas realizan sus operaciones del día a día. Sin embargo, actualizarse tecnológicamente puede ser todo un desafío, sobre todo para las organizaciones que realizan sus procesos de forma tradicional o manual, en cuanto a costos, adaptación del personal y la integración con nuevas tecnologías. La implementación de estrategias de tecnología en una empresa puede generar una reducción de costos operativos, liberar tiempo y recursos monetarios que pueden ser destinados para otras áreas o fines de la organización (CoveríX, 2024).

La digitalización está transformando el panorama empresarial, ofreciendo múltiples beneficios que impulsan la eficiencia y competitividad de las organizaciones. La automatización de procesos reduce la carga de tareas repetitivas, minimiza errores humanos y optimiza recursos. Hay un aumento en la productividad y gracias a eso, los empleados pueden realizar sus tareas de manera más efectiva y autónoma. Al optimizar los recursos, el equipo se libera de ciertas responsabilidades, permitiéndoles concentrarse en actividades más importantes (Servicios Empresariales, 2024).

La digitalización de una empresa implica la adopción y el uso de tecnologías digitales para mejorar sus procesos, operaciones y servicios. Desde la gestión de datos hasta la automatización de tareas la digitalización facilita la comunicación y la colaboración entre diferentes departamentos dentro de una empresa al eliminar almacenes de información y permitir un flujo de trabajo más fluido y eficiente (InnoQubit, s/f).

Framework .NET MAUI para el desarrollo de aplicaciones móviles

¿Qué es un Framework?

Un framework es una estructura tecnológica utilizada por desarrolladores para el desarrollo de aplicaciones, ya sean escritorio, móviles o web. Estas estructuras las componen un lenguaje de programación, una colección de librerías entre otras cosas que las hacen especiales para su utilización en ese tipo de desarrollo

Es un conjunto de reglas y convenciones que se usan para desarrollar software de manera más eficiente y rápida. Estos marcos de trabajo se emplean para ahorrar tiempo y esfuerzo en el desarrollo de aplicaciones, ya que proporcionan una estructura básica que se puede utilizar como punto de partida. Además, los frameworks también ofrecen soluciones a problemas comunes en el desarrollo de software, lo que significa que los desarrolladores pueden centrarse en las funcionalidades específicas de su aplicación en lugar de perder tiempo resolviendo problemas técnicos (Lucena, 2023).

¿Qué es .NET MAUI?

Es un marco multiplataforma que permite el desarrollo tanto de aplicaciones móviles en IOS y Android, como de escritorio en Windows y macOS, y también desarrollar en WEB (Véase figura 9).

.NET MAUI es una evolución de código abierto de Xamarin.Forms, extendida desde escenarios móviles a escritorio, con controles de interfaz de usuario recopilados para mejorar el rendimiento y la extensibilidad. Mediante .NET MAUI, puede crear aplicaciones multiplataforma con un solo proyecto, pero puede agregar recursos y código fuente específicos de la plataforma si es necesario. Uno de los objetivos clave de .NET MAUI es permitirle implementar la mayor parte de la lógica de la aplicación y el diseño de la interfaz de usuario en una única base de código (britch, Rastegarinia, et al., 2024).

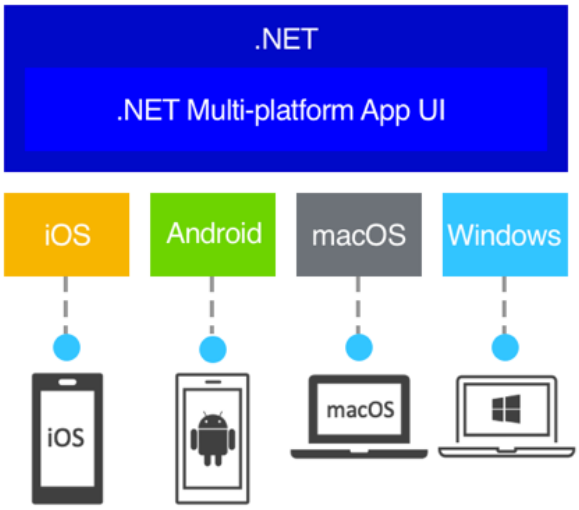


Figura 9: .NET MAUI

En el desarrollo móvil de .NET MAUI se utilizan dos tipos de lenguaje, uno de diseño o mejor conocido como Etiquetas y uno de programación que al ser un Framework de Microsoft se utiliza el lenguaje c#, el lenguaje de diseño es aquel que le da forma a la interfaz de usuario mediante etiquetas predefinidas en el framework, el diseño en general, la distribución y visualización es lo que maneja ese lenguaje, el lenguaje de programación se encarga del comportamiento de la aplicación al ejecutar ciertas acciones, su funcionalidad lógica o al manejo de datos e información.

Los diseños de .NET MAUI se usan para componer controles de interfaz de usuario en estructuras visuales, y cada diseño normalmente contiene varias vistas. Las clases de diseño suelen contener lógica para establecer la posición y el tamaño de los elementos secundarios (britch, Bughiu, et al., 2024).

El manejo de etiquetas que son lo que les dan diseño a las aplicaciones móviles es algo indispensable, y dentro de .NET MAUI existen una gran cantidad de las cuales algunas son bastante básicas y comunes dentro de este marco de desarrollo que te permiten hacer una gran cantidad de cosas (Véase tabla 9).

Tabla 9: Etiquetas más comunes de .NET MAUI

Etiqueta	Descripción
<Label/>	Proporciona texto dentro de la aplicación
<Grid></Grid>	Proporciona una división de tabla a la pantalla de la aplicación para un mejor manejo de la estructura, mediante la utilización de sus propiedades de asignación de filas y columnas.
<Entry/>	Proporciona un cuadro donde el usuario puede escribir texto
<Button/>	Proporciona un botón
<Image/>	Proporciona una imagen dentro de la aplicación asignándole la dirección de la imagen dentro de la aplicación.
<Borde></Borde>	Aplica un contenedor que da opciones de utilización de bordes.
<Frame></Frame>	Se usa para una vista o diseño que se le puede configurar color, sombra y otras opciones graficas

Principios de diseño UI/UX para el desarrollo de aplicaciones móviles

Diseño

Dentro del mundo del diseño la elección correcta de colores es de suma importancia, no es únicamente escoger colores que tengan estilo con el tema de lo que harás, tienes que utilizarlos de una forma correcta para que el usuario que se encuentre usando la aplicación o que se encuentre visualizándola no le moleste y tampoco le incomode su uso, esto se debe a que ciertos colores dentro de otros colores fuerzan la vista, o de igual manera usar ciertos colores en mayor medida provocan mareos o incomodidad extrema, por ejemplo: amarillo, rojo oscuro, negro, generalmente los colores super brillosos, o colores ultra oscuros.

Aunque UX (Experiencia de Usuario) y UI (Interfaz de Usuario) son conceptos diferentes, están estrechamente relacionados y dependen el uno del otro. El diseño de experiencia de usuario (UX) se enfoca en el recorrido del usuario en una aplicación o proceso, y en la forma y función general de un producto o tecnología. En cambio, el diseño de interfaz de usuario (UI) se centra en cómo se ve y funciona el exterior de un producto (los elementos tangibles del proceso). La UX se centra en la experiencia general del usuario final, incluidas sus percepciones, emociones y respuestas al producto, sistema o servicio. Se define por criterios que incluyen la facilidad de uso, la accesibilidad y la conveniencia. La UX engloba dos conceptos claves, el de Usabilidad, es decir la facilidad con la que un usuario puede usar un producto digital y la Accesibilidad que abarca aspectos de diseño responsivo e inclusividad. Es decir, toma en consideración desde qué dispositivo lo hacen (móvil, laptop, tablet, etc.) y si tienen una discapacidad (física, visual, motriz, auditiva o cognitiva) para adaptar sus canales (Bidart, 2023).

Que una aplicación móvil tenga un color determinado no es casualidad o simple estética, los colores también son una forma de comunicar, por lo que la psicología del color aporta un significado diferente a cada uno de ellos (Psicologia del color, 2020).

Rojo. Es un color que llama a la acción, a hacer algo. Simboliza pasión e invita a ser impulsivo ya que crea cierta sensación de urgencia (Véase figura 10).

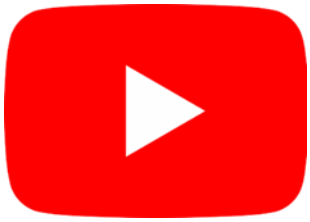


Figura 10: Ejemplo del color - Rojo - Logo de YouTube

Verde. Este color siempre ha estado asociado con la naturaleza, el dinero o la tranquilidad, pero en el mundo del marketing, el color verde se utiliza para serenar y dar confianza al cliente (Véase figura 11).

Figura 11: Ejemplo del color - Verde - Logo de Airhopping

Azul. Este color siempre está relacionado con la seguridad y elegancia, pero en el mundo de las apps, se utiliza para aportar paz al usuario, es decir, al utilizar las apps que tienen este color se sienten tranquilos y seguros de compartir en ellas lo que quieran (Véase figura 12).

Figura 12: Ejemplo del color - Azul – Logo de Skype

Naranja. Refleja emoción y al igual que el rojo, este color tienda a la llamada a la acción para generar sobre todo leads o compras impulsivas (Véase figura 13).

Figura 13: Ejemplo del color - Naranja – Logo de Tinder

Morado. Va siempre asociado a marcas creativas y originales ya que representa el éxito (Véase figura 14).

Figura 14: Ejemplo del color - Morado – Logo de Cabify

Amarillo. Representa optimismo y felicidad por eso, casi todas las compañías aéreas tienen la representación de este color en sus logos ya que viajar es una sensación de felicidad al conocer nuevos destinos y también intentan hacer ver que no hay que tener miedo a volar (Véase figura 15).

Figura 15: Ejemplo del color - Amarillo - Logo de Vueling

El color es un tema importante al momento de diseñar una aplicación, porque eso definirá si es usada o si es desechada, pero, un factor que llega a importar muchísimo en esos aspectos es la estructura de la aplicación, que puede incomodar o molestar al usuario inconscientemente (Véase figura 16)

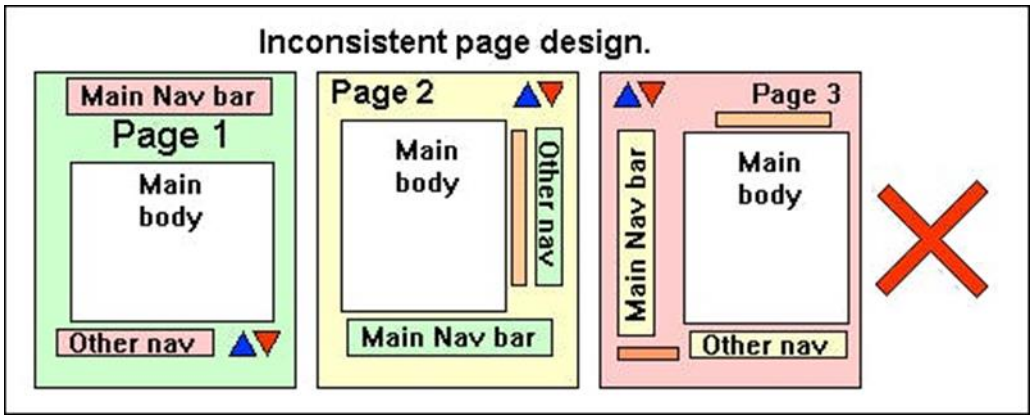


Figura 16: Diseño de interfaz molesto

En la figura 17 podemos apreciar una interfaz más amigable y cómoda para el usuario.

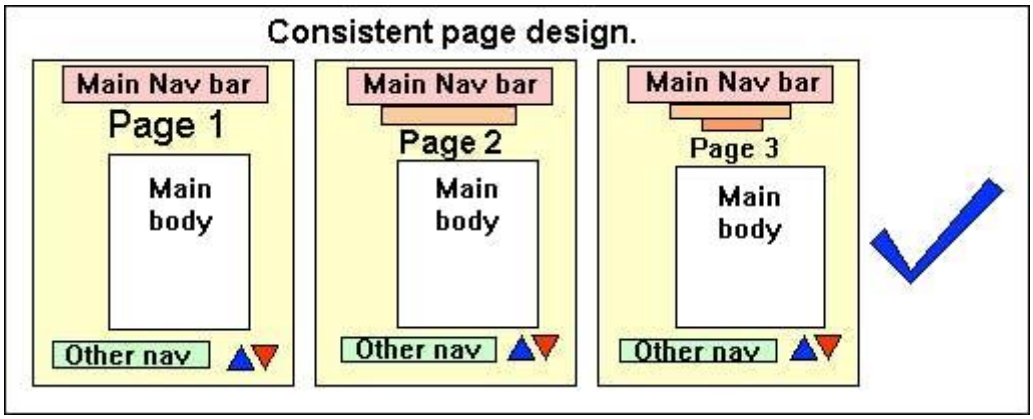


Figura 17: Diseño de interfaz cómodo

El diseño visual en el desarrollo de aplicaciones cumple un papel transversal en el diseño de la identidad de marca en el plano estético como actividad que se encarga de la definición de vocabularios visuales para el manejo de la información presentada, el color y la tipografía como elementos esenciales para transmitir con eficacia y singularidad unos valores estéticos muy específicos para cada software (Serna, 2016). Diseñar con la mente en una minoría hará que te alejes de la mayoría, dentro de la publicidad una campaña triunfa mientras a más gente llega, Y Algunos consejos que ayudan mucho para un diseño inclusivo, agradable y eficiente son los siguientes:

- Darles una jerarquía a las cosas, los accesos más importantes colocarlos arriba o abajo en la aplicación, y las que tengan la misma importancia a un lado de otra.
- El texto no justificarlo porque eso puede confundir a la gente con dislexia.
- La relación entre un color y su fondo varía de 1 a 21 dependiendo la luz. Un tamaño de texto pequeño requiere mayor contraste. Se recomienda una relación de al menos 4.5:1 entre el color del texto y el color de fondo, disminuyendo a un mínimo de 3:1 cuando la fuente es de 14 pt en negrita o 18+ pt.
- El color debe utilizarse como complemento no como el medio de transmisión principal, apoyando a los textos, iconos y texturas.
- En los apartados de rellenado deben destacar y no ocultarse ante el fondo.

(Caballero, 2023)

Toma de decisiones basada en datos

La toma de decisiones basada en datos (DDDM) es un enfoque que enfatiza el uso de datos y análisis en lugar de la intuición para fundamentar las decisiones comerciales. Implica aprovechar fuentes de datos como los comentarios de los clientes, las tendencias del mercado y los datos financieros a fin de guiar el proceso de toma de decisiones. Al recopilar, analizar e interpretar datos, las organizaciones pueden tomar mejores decisiones que se alineen más estrechamente con las metas y objetivos del negocio (Mucci, s/f).

Beneficios

La toma de decisiones basada en datos representa una variedad de beneficios, IBM en su blog ‘Toma de decisiones basada en datos’ nos da algunos ejemplos, como serian: Compromiso y satisfacción del cliente, Aumento de la retención de clientes, Prácticas empresariales proactivas, Mejor planeación estratégica, Oportunidades de crecimiento, Gestión estratégica de inventarios y Protección contra sesgos.

Análisis de datos en el proceso de producción

El análisis de datos impulsa la toma de decisiones, mejora la eficiencia operativa, ahorra costos y brinda a las organizaciones una ventaja competitiva. Según PwC, el análisis de datos puede mejorar el tiempo de actividad de fabricación al 9%, disminuir los costos en 12%, mitigar los riesgos de seguridad, salud, medio ambiente y calidad mediante 14%y prolongar la vida útil de los activos antiguos mediante 20% (Khan, 2025).

Uso de inteligencia artificial y machine learning en la producción acuícola.

El aprendizaje automático y aprendizaje profundo son los métodos más utilizados en investigaciones realizadas en temas de gestión inteligente de granjas acuícolas, apoyo más dinámico para toma de decisiones en acuicultura e identificación de áreas de acuicultura. La tendencia que existe entre el crecimiento de los peces en función del tiempo debe ser optimizada en acuicultura, así como la combinación de tratamientos que optimizan el crecimiento a través del tiempo. En la acuicultura, la predicción precisa de la ingesta de alimento para los peces del grupo se considera crucial a cualquier sistema de alimentación. Por otro lado, la sostenibilidad de la acuicultura depende también del aspecto económico, siendo el costo de alimentación el que influye de manera significativa y en mayor proporción que determina cuan rentable es la actividad para mantenerse activo. El nivel de alimentación está directamente relacionado con la eficiencia de producción y costo de reproducción (Vásquez-Quispesivana et al., 2022).

Materiales y Métodos

Toma y análisis de requisitos

Como primer paso, se tomaron y analizaron los requisitos para el sistema, para cumplir el objetivo se usó como base y referencia los formatos de papel utilizados para el registro de los datos dentro de la acuícola (Véase figura 18) para identificar los campos que se deben ingresar en el sistema, además, mediante reuniones también se tomaron en cuenta otros requisitos para el sistema como la implementación de exportar los datos en PDF y hojas de cálculo de Excel.

En este paso también se decidió en que sería hecho el sistema y se optó por utilizar el IDE de Visual Studio, con los frameworks .NET MAUI para el desarrollo de la aplicación móvil y para la aplicación de escritorio se utilizaría el framework Windows Forms, también se decidió utilizar MySQL por su sencilles con conexiones remotas.

En este paso también se decidió en que sería hecho el sistema y se optó por utilizar el IDE de Visual Studio, con los frameworks .NET MAUI para el desarrollo de la aplicación móvil y para la aplicación de escritorio se utilizaría el framework Windows Forms, también se decidió utilizar MySQL por su sencilles con conexiones remotas.

Figura 18: Formatos utilizados

Desarrollo de una aplicación móvil

Para el diseño de la aplicación móvil eran requeridas 13 pantallas, donde 6 serán utilizadas para el registro de datos, 5 pantallas serán utilizadas para poder observar los datos registrados, 1 pantalla seria utilizada para el menú que dará la opción de entrar en alguna de las diferentes áreas para registrar los datos, 1 pantalla para el Login y 1 pantalla para registrar usuario (véase la figura 19).

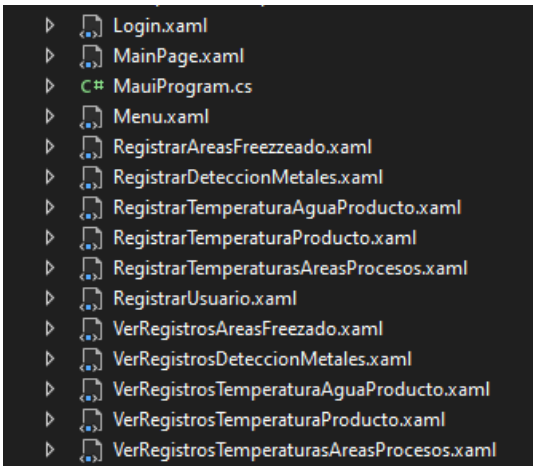


Figura 19. Pantallas requeridas

Para la estructuración de las pantallas se optó por la utilización de la etiqueta ***<Grid>*** para una previsualización de donde se pondrían los componentes además de tener una forma prevista de la pantalla en diseño simple y en líneas (Véase figura 20 y 21).

```
<Grid RowDefinitions="30*,60*,10*"
      ColumnDefinitions="25*,20*,10*,20*,25*">
    <Border Grid.Row="0"
            Grid.ColumnSpan="5"
            Margin="5">
        <Grid RowDefinitions="30*,30*,30*"
              ColumnDefinitions="50*,50*">
            <Border Grid.Row="0"
                    Grid.Column="0"
                    Grid.RowSpan="3"></Border>
            <Border Grid.Row="0"
                    Grid.Column="1"
                    Grid.ColumnSpan="2"></Border>
            <Border Grid.Row="1"
                    Grid.Column="1"
                    Grid.ColumnSpan="2"></Border>
            <Border Grid.Row="2"
                    Grid.Column="1"
                    Grid.ColumnSpan="2"></Border>
        </Grid>
    </Border>
    <Border Grid.Row="1"
            Grid.ColumnSpan="5"
            Margin="5">
        <Grid RowDefinitions="25*,20*,15*,15*,15*,10*">
            <Border Grid.Row="0">
            </Border>
            <Border Grid.Row="1">
            </Border>
            <Border Grid.Row="2">
            </Border>
            <Border Grid.Row="3">
            </Border>
            <Border Grid.Row="4">
            </Border>
            <Border Grid.Row="5">
            </Border>
        </Grid>
    </Border>
    <Border Grid.Row="2"
            Grid.ColumnSpan="5">
        <Grid ColumnDefinitions="10*,30*,20*,30*,10*">
            <Border Grid.Column="1"
                    VerticalOptions="Center"
                    HeightRequest="60">
            </Border>
            <Border Grid.Column="3"
                    VerticalOptions="Center"
                    HeightRequest="60">
            </Border>
        </Grid>
    </Border>
</Grid>
```

Figura 20: Ejemplo de código de la estructura con etiqueta <Grid>

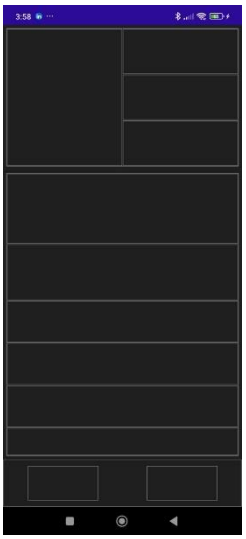


Figura 21: Ejemplo visual de la estructura con etiqueta <Grid>

Para el diseño del Login se optó por dos Entry para ingresar Correo electrónico y contraseña, siendo que la contraseña aparezca oculta o con un símbolo, y dos botones, uno para registrar usuario y otro para ingresar al menú.

En la pantalla para registrar usuario se pedirán datos clave como nombre, apellido, teléfono y contraseña, la contraseña tendrá un botón para poder ver y ocultar la contraseña en caso de querer verificar.

En el menú se manejarán solo los botones para entrar a cada área para registrar los datos, cada botón con el nombre y una imagen referente al área.

En la pantallas de registrar datos tendrán un título del área en la parte superior, cada pantalla tendrán colores referentes al área, en la parte superior se tendrá un botón para entrar a una pantalla donde se podrán ver los registros, cada pantalla para ver los registros tendrán textos que serán el nombre del campo para rellenar y se usaran entradas de texto para registrar el valor del campo, en caso de ser hora o fecha se usaran reloj o calendario respectivamente, en caso de existir opciones limitadas ya establecidas en los formatos se usara componente con los valores ya definidos y con la opción de seleccionar otro valor.

En las pantallas para ver los datos manejará colores iguales a la pantalla anterior, un componente para seleccionar la fecha en la parte superior, y en la parte del cuerpo

una lista para mostrar los datos, para darle contraste y margen a la lista se usará un frame.

Para la programación de la aplicación primero se programó una clase para el manejo de la base de datos, la clase la nombramos ManipulacionDatos y contiene los métodos para registrar y para obtener datos de cada tabla de la base de datos (Véase figura 22).

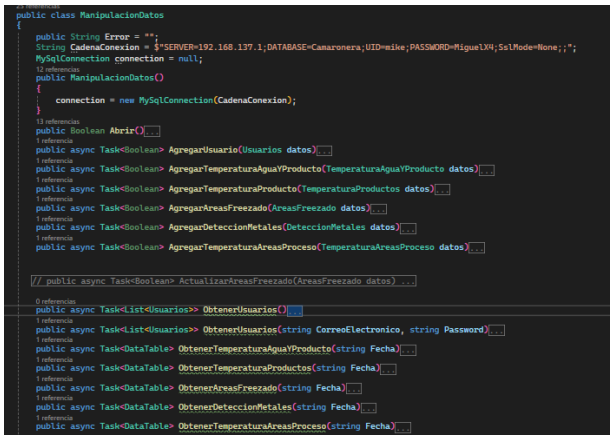


Figura 22. Clase ManipulacionDatos

Hay una función de abrir que se encarga de verificar que la conexión este abierta (Véase figura 23).

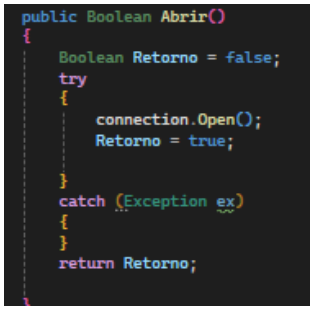


Figura 23. Función Abrir

Para las funciones de registrar datos utilizan la misma sintaxis todas exceptuando por la parte del comando y del parámetro que es donde cada función es diferente, las funciones comienzan con una variable booleana en false, dentro del try de un try catch se pone un if que pregunta si la conexión está abierta, en caso de no estar abierta la conexión la variable de error será igual al mensaje deseado y retorna la variable booleana, en caso de estar abierta la conexión se escribirá en una variable String el comando SQL y se ejecutara, se cambiara la variable booleana a true, dentro del catch se escribe en la variable de error el mensaje de la excepción, finalmente la función termina retornando la variable booleana (Véase figura 24) el resto funciones son idénticas lo único que cambia es el comando SQL utilizado para la consulta.

```
public async Task<Boolean> AgregarTemperaturaAguaYProducto(TemperaturaAguaYProducto datos)
{
    Boolean Retorno = false;
    try
    {
        if (!Abrir()) { Error = "No se pudo comunicar con el servidor, verifique la conexion"; return Retorno; }
        String comando = "INSERT INTO TemperaturasAguaProducto(Fecha, Hora, Agua, Producto, Granja, NoTina, KG_NA2s205) " +
                    $"VALUES('{datos.Fecha}', '{datos.Hora}', '{datos.TemperaturaAgua}','{datos.TemperaturaProducto}'," +
                    $"'{datos.Granja}','{datos.Tina}','{datos.KGNA2S205}');";
        MySqlCommand cmd = connection.CreateCommand();
        cmd.CommandText = comando;
        await cmd.ExecuteNonQueryAsync();
        Retorno = true;
    }
    catch (Exception ex)
    {
        Error = ex.Message;
    }
    return Retorno;
}
```

Figura 24: Ejemplo funciones registrar

Las funciones de obtener datos funcionan de la misma manera, solo que estas comparten de parámetro la fecha, y a diferencia de las funciones anteriores, estas devuelven un DataTable o una tabla con los datos tomados, la función comienza inicializando el DataTable que se retornara después, dentro del Try del Try catch se pregunta si la conexión está abierta, en caso de no estar abierta entrara al catch donde se escribirá el mensaje de error en la variable error, en caso de estar abierta se creara la variable para ejecutar el comando SQL, en una Variable String se guarda la consulta SQL y después se ejecuta el comando, después se guardan dentro del DataTable creado los datos obtenidos y al final de la función se retorna la tabla (Véase figura 25) el resto funciones son idénticas lo único que cambia es el comando SQL utilizado para la consulta.

```
public async Task<DataTable> ObtenerTemperaturaAguaYProducto(string Fecha)
{
    DataTable dataTable = new DataTable();
    try
    {
        Abrir();
        MySqlCommand cmd = connection.CreateCommand();
        String Comando = "SELECT Fecha, Hora, Agua, Producto, Granja, NoTina, WG_NA2s2O5 FROM TemperaturasAguaProducto " +
                         $"WHERE Fecha = '{Fecha}';";
        cmd.CommandText = Comando;
        MySqlDataAdapter adapter = new MySqlDataAdapter(cmd);
        adapter.Fill(dataTable);
    }
    catch (Exception ex)
    {
        Error = ex.Message;
    }
    return dataTable;
}
```

Figura 25: Ejemplo funciones obtener

La función de ObtenerUsuarios es diferente a las anteriores ya que esta pide un correo electrónico y contraseña como parámetros en la función, además que en lugar que devuelva una tabla devuelve una lista de la clase Usuarios para ser más fácil su uso, se comienza inciializando una variable DataTable y otra de List<Usuarios>, dentro del Try del Try catch se pregunta si la conexión está abierta, en caso de no estar abierta entrara al catch donde se escribirá el mensaje de error en la variable error, en caso de estar abierta se creara la variable para ejecutar el comando SQL, en una Variable String se guarda la consulta SQL y después se ejecuta el comando, se guarda en el DataTable, después del try catch por medio de un foreach se recorren las filas de la tabla y se van insertando en la lista, al finalizar se devuelve la lista (véase figura 26).

```
public async Task<List<Usuarios>> ObtenerUsuarios(string CorreoElectronico, string Password)
{
    DataTable dataTable = new DataTable();
    List<Usuarios> Lista = new List<Usuarios>();
    try
    {
        Abrir();
        MySqlCommand cmd = connection.CreateCommand();
        String comando = "Select Correo, Contra, Nombres, AP_Paterno, AP_Materno, Telefono from Usuarios " +
                         $"Where Correo = '{CorreoElectronico}' and Contra = '{Password}';";
        cmd.CommandText = comando;
        MySqlDataAdapter adapter = new MySqlDataAdapter(cmd);
        adapter.Fill(dataTable);
    }
    catch (Exception ex)
    {
        Error = ex.Message;
    }
    foreach (DataRow item in dataTable.Rows)
    {
        Usuarios objeto = new Usuarios();
        objeto.CorreElectronico = item[0].ToString();
        objeto.Password = item[1].ToString();
        objeto.Nombre_s = item[2].ToString();
        objeto.ApellidoPaterno = item[3].ToString();
        objeto.ApellidoMaterno = item[4].ToString();
        objeto.Celular = item[5].ToString();
        Lista.Add(objeto);
    }

    return Lista;
}
```

Figura 26. Función ObtenerUsuarios

También se utilizan clases para recolectar los datos de forma más ordenada y limpia (Véase Figura 27-32).

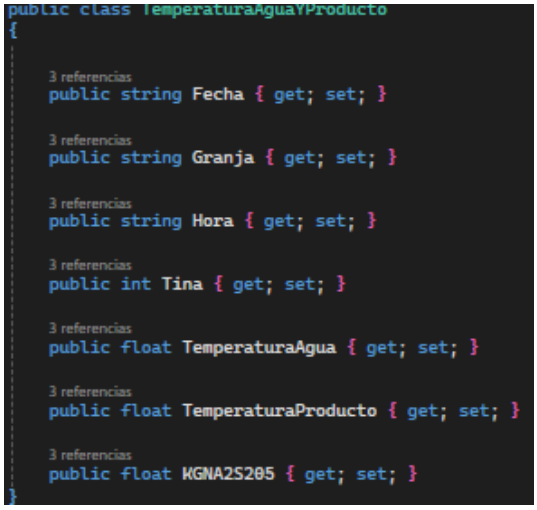


Figura 27. Clase TemperaturaAguaYProducto

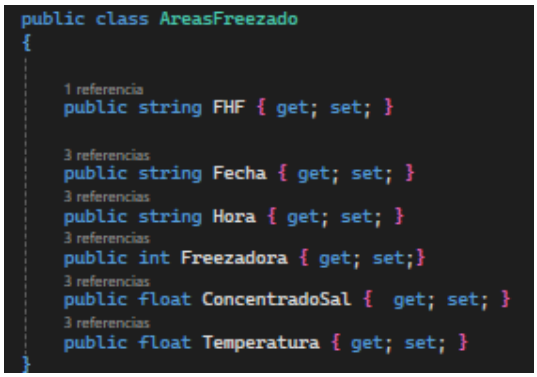


Figura 28. Clase AreasFreezado

```
public class DeteccionMetales
{

    0 referencias
    public long ID { get; set; }

    4 referencias
    public string Fecha { get; set; }

    4 referencias
    public string Hora { get; set; }

    4 referencias
    public string Tipo { get; set; }

    5 referencias
    public string FE { get; set; }

    5 referencias
    public string NonFE { get; set; }

    5 referencias
    public string Sos { get; set; }

    3 referencias
    public string Monitoreo110 { get; set; }

    3 referencias
    public string Monitoreo210 { get; set; }
}
```

Figura 29. Clase DeteccionMetales

```
public class TemperaturaProductos
{

    public string Fecha { get; set; }

    public string Hora { get; set; }

    public string Granja { get; set; }
    3 referencias
    public int TinaRMP { get; set; }

    3 referencias
    public float PesoLote { get; set;}
    3 referencias
    public string Mesa { get; set; }
    3 referencias
    public string Bascula { get; set; }
    3 referencias
    public string ClaseCarne {  get; set; }

}
```

Figura 30. Clase TemperaturaProductos

```
public class TemperaturaAreasProceso
{

    1 referencia
    public string FHA { get; set; }
    3 referencias
    public string Fecha { get; set; }
    3 referencias
    public string Hora { get; set; }
    3 referencias
    public string Area { get; set; }
    3 referencias
    public float Temperatura { get; set; }


}
```

Figura 31. Clase TemperaturaAreasProceso

```
public class Usuarios
{
    6 referencias
    public string CorreElectronico { get; set; }

    5 referencias
    public string Nombre_s { get; set; }

    5 referencias
    public string ApellidoPaterno { get; set; }

    5 referencias
    public string ApellidoMaterno { get; set; }

    5 referencias
    public string Celular { get; set; }

    6 referencias
    public string Password { get; set; }
}
```

Figura 32. Clase Usuarios

El Login, en el inicio del código se inicializa un objeto donde se encuentra los métodos para la comunicación con la base de datos (Véase figura 33), al momento darle al botón registrar abre una pantalla para registrar usuarios nuevos (Véase figura 34), y al momento de darle al botón de ingresar, validara que los datos de correo electrónico y contraseña estén ingresados, luego en la base de datos buscara el usuario en la base de datos, mandara un mensaje de error en caso de no existir y en caso de existir entra al menú (Véase figura 35), también antes de verificar la existencia envía la contraseña encriptada ya que así es como se guarda en la base de datos, el método de encriptación consiste en convertir la cadena en un arreglo de bytes utilizando "*System.Text.Encoding.ASCII.GetBytes(<String>)*" (Véase figura 36).

```
4 referencias
public partial class Login : ContentPage
{
    ManipulacionDatos manejoDatos = new ManipulacionDatos();
    0 referencias
    public Login()
    {
        InitializeComponent();
    }
```

Figura 33. Código objeto de la clase para verificación de existencia de usuario

```
0 referencias
private void btnRegistrar_Clicked(object sender, EventArgs e)
{
    Navigation.PushModalAsync(new RegistrarUsuario());
}
```

Figura 34. Código abrir registrar usuario

```
0 referencias
private async void btnIngresar_Clicked(object sender, EventArgs e)
{
    if (CorreoElectronico.Text != null && CorreoElectronico.Text.Trim() != "" && Contraseña.Text != null && Contraseña.Text.Trim() != "")
    {
        DatosUsuario datos = null;
        EncriptarDesencriptar en = new EncriptarDesencriptar();
        string contraseñaEN = en.Encriptar(Contraseña.Text.Trim());
        List<Usuarios> usuario = await manejoDatos.ObtenerUsuarios(CorreoElectronico.Text.ToLower().Trim(), contraseñaEN);
        if (usuario.Count == 1)
        {
            Navigation.PushModalAsync(new Menu(datos));
        }
        else
        {
            DisplayAlert("Alerta", "Usuario no registrado", "OK");
        }
    }
    else
    {
        DisplayAlert("Alerta","Agregar Correo Electronico/Contraseña","OK");
    }
}
```

Figura 35. Botón Ingresar

```
2 referencias
public string Encriptar(string clave)
{
    string clave2 = string.Empty;

    byte[] encrypted = System.Text.Encoding.ASCII.GetBytes(clave);
    clave2 = Convert.ToBase64String(encrypted);

    return clave2;
}
```

Figura 36. Método de Encriptar

En el menú puedes elegir una de las 5 áreas para registrar datos y te enviaran a esa pantalla (Véase figura 37).

Figura 37: Funciones de los eventos Clicked de los botones

Todas las pantallas tienen un botón para entrar a su respectiva pantalla para ver los datos ya registrados dentro de sistema (Véase figura 38 y 39).

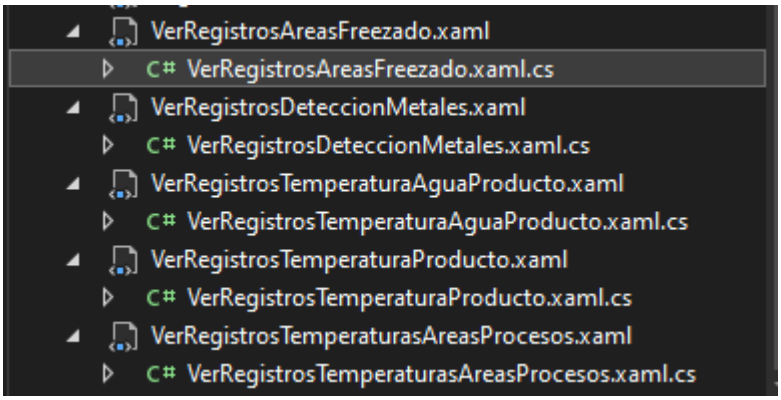


Figura 38. Pantallas para ver los datos registrados

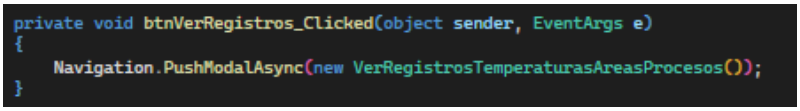


Figura 39: Ejemplo función de llamado a la pantalla para ver los registros

Las pantallas para ver los registros al iniciar el código se crea el objeto de manipulación de datos y un ObservableCollection de la clase de los respectivos datos que deseas tomar usando como ejemplo la de Áreas de frezado (Véase figura 40).

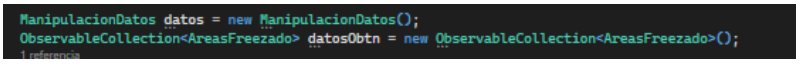


Figura 40. Ejemplo del inicio del código en ver registro

Al cargar la lista, en el evento Loaded o al cambiar la fecha con el evento DateSelected ejecuta la función “Agregar ()” (Véase figura 41).

```
private void dpFecha_DateSelected(object sender, DateChangedEventArgs e)
{
    Agregar();
}

0 referencias
private void Lista_Loaded(object sender, EventArgs e)
{
    Agregar();
}
```

Figura 41. Función DateSelected y Loaded en todas las pantallas para ver los registros

La función agregar es la misma en todas las pantallas para ver los registros con el cambio de la clase donde se guardarán los datos y la función que se llama en la manipulación de datos, usando un DataTable se piden los datos de la respectiva función de los datos de la plantilla, pasando por un foreach sobre las filas y siendo agregados al observable collection, cuando termina el foreach el item source de la lista se le asigna el observable collection (Véase figura 42).

```
private async void Agregar()
{
    DataTable dt = await datos.ObtenerAreasFreezado(dpFecha.Date.ToString("yyyy-MM-dd"));
    foreach (DataRow item in dt.Rows)
    {
        AreasFreezado dato = new AreasFreezado();
        dato.Fecha = Convert.ToDateTime(item[0]).ToString("dd/MM/yyyy");
        dato.Hora = item[1].ToString();
        dato.Freezadora = Convert.ToInt32(item[2]);
        dato.Temperatura = (float)Convert.ToDecimal(item[3]);
        dato.ConcentradoSal = (float)Convert.ToDecimal(item[4]);
        datosObtn.Add(dato);
    }
    Datos.ItemsSource = datosObtn;
}
```

Figura 42. Función Agregar de Áreas de Freezeado

Implementar una base de datos en una Raspberry Pi

Primero se requiere actualizar la Raspberry Pi para eso primero se ejecuta el comando en la terminal *sudo apt update* y después *sudo apt upgrade* (Véase figura) y se presiona Y seguido de Enter cuando sea pedido (Véase figura 43 y 44).

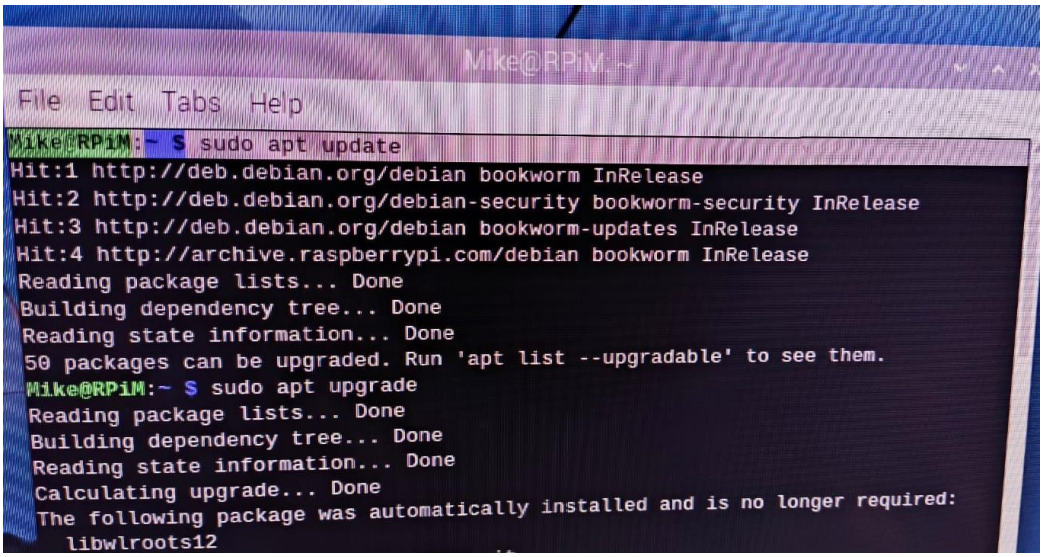


Figura 43: Actualización Raspberry pi 3

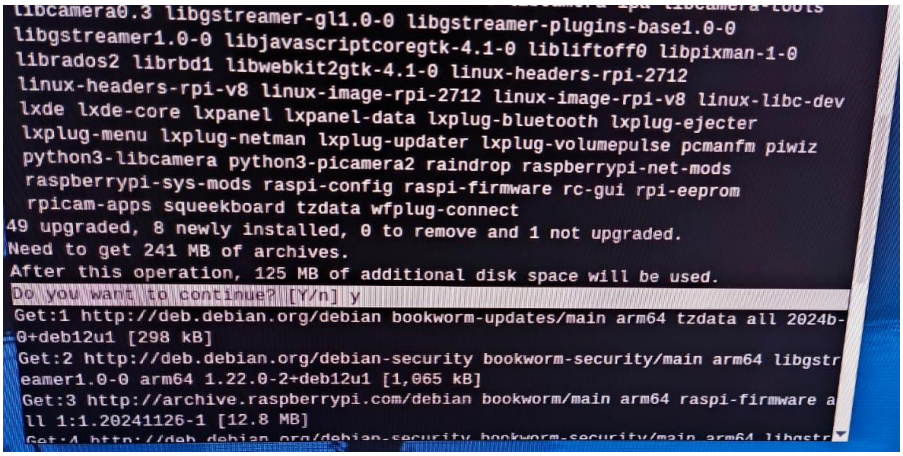


Figura 44: Línea donde pide presionar letra y

Ahora se utiliza el comando sudo *apt install mariadb-server* (véase figura 45) y se escribe ‘y’ en caso de ser pedido.

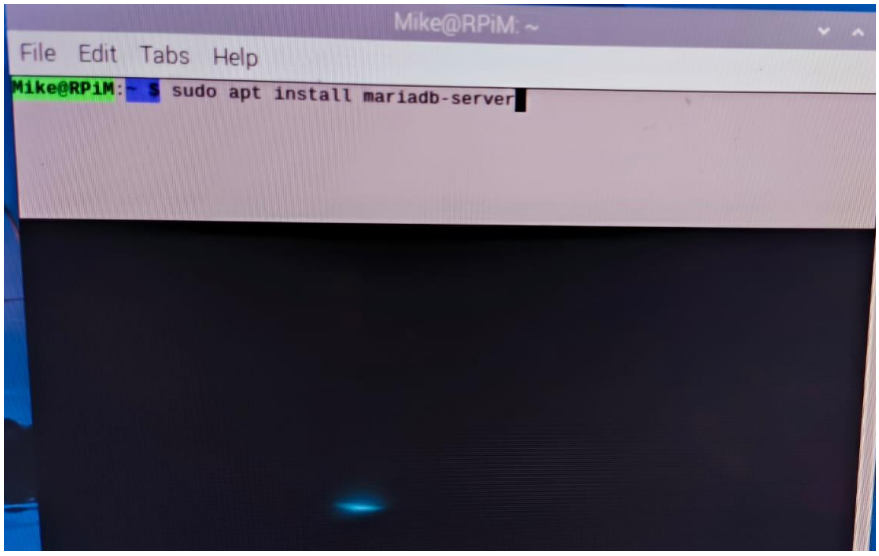


Figura 45: Instalación MariaDB

Se utilizará el comando *sudo mysql_secure_installation* para ingresar el usuario root, cuando te salga el mensaje como en la figura 46 presiona Enter, y después presionar ‘y’ en el resto de los mensajes siguientes, al finalizar le pedirá que escriba la contraseña y que la confirmen (Véase figura 47).

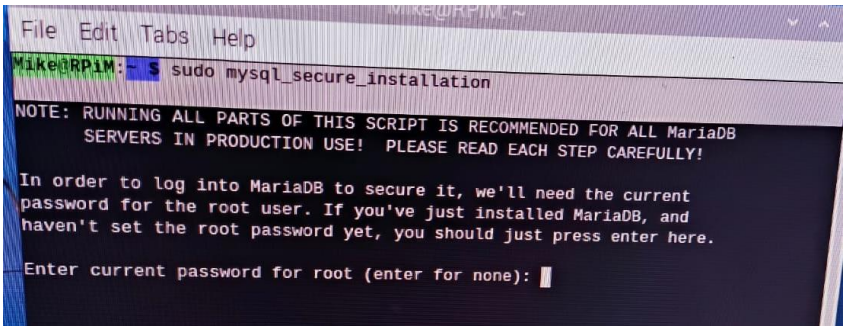


Figura 46: Mensaje inicial

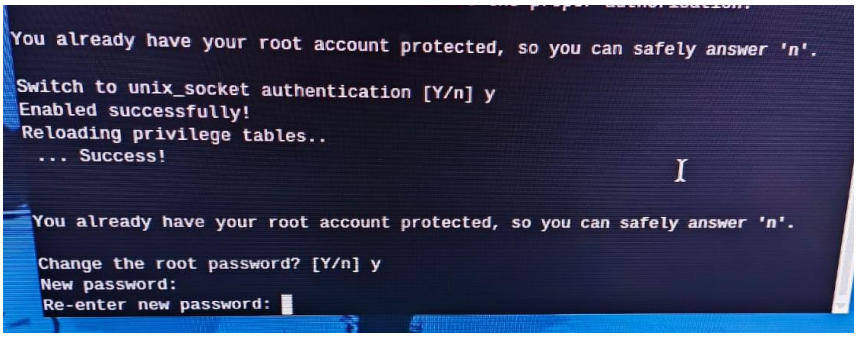


Figura 47: Finalización del usuario root

Ahora creado el usuario root podemos entrar a la base de datos con el usuario utilizando el comando *sudo mysql -uroot -p* para crear un usuario que nos permita comunicarnos remotamente con la base de datos, ya que el usuario root no está permitido para conexiones remotas, al entrar al usuario root, utilizaremos el comando *Create database* escribimos el nombre de la base de datos y finalizamos con ‘;’ después usamos *use* y ponemos el nombre de la base de datos (Véase figura 48).

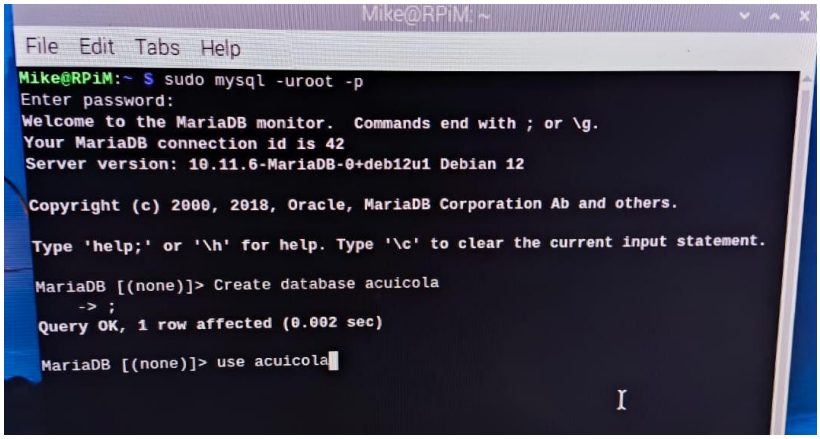


Figura 48: Creación y utilización de base de datos

Ahora utilizaremos los siguientes comandos para crear nuestro usuario que se pueda comunicar remotamente:

CREATE USER '<username>'@'%' IDENTIFIED BY '<password>';

GRANT ALL PRIVILEGES ON *.* TO '<username>'@'%';

FLUSH PRIVILEGES;

(Véase figura 49), el '%' hace referencia que puede comunicarse desde cualquier IP y el '*.*' se refiere que el usuario tendrá privilegios en todas las tablas de todas las bases de datos.

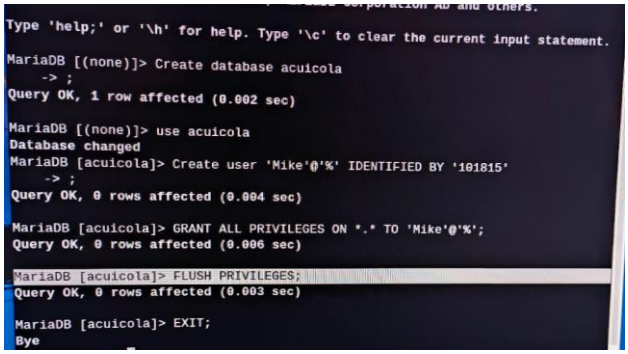


Figura 49: Creación del usuario

Ahora para que MariaDB admita conexiones remotas, necesitamos hacer unos cambios en uno de sus archivos, utilizaremos el comando para editar el archivo con permisos de administrador *sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf* y buscaremos la línea que dice bind-address y debería encontrarse en 127.0.0.1 (Véase figura 50) se cambia a 0.0.0.0 que hace referencia que puede admitir conexiones desde cualquier IP o de forma remota (Véase figura 51).

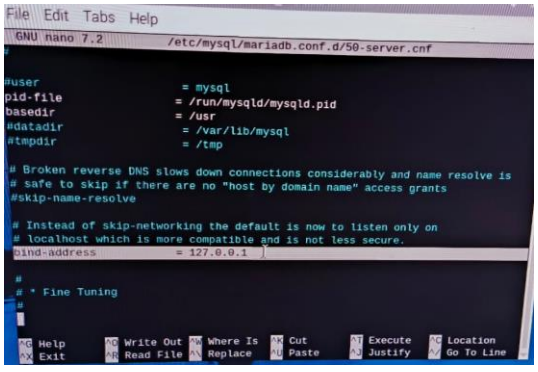


Figura 50: bind-address original

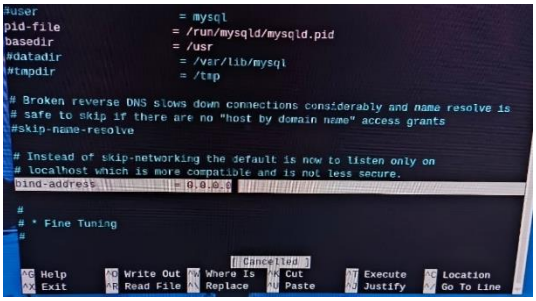


Figura 51: bind-address modificada

Se reinicia el servicio de MariaDB con el comando *sudo systemctl restart mariadb.*
En este momento ya se puede utilizar MariaDB en la Raspberry, pero, para un uso más cómodo se instalará el uso de un gestor de base de datos con interfaz web, se instalará phpmyadmin, utilizando el comando sudo apt install phpmyadmin -y seguido aparecerá un mensaje se elige apache2, aparece otro mensaje se presiona Yes (Véase figura 52 y 53), después pedirá contraseña y confirmar contraseña.

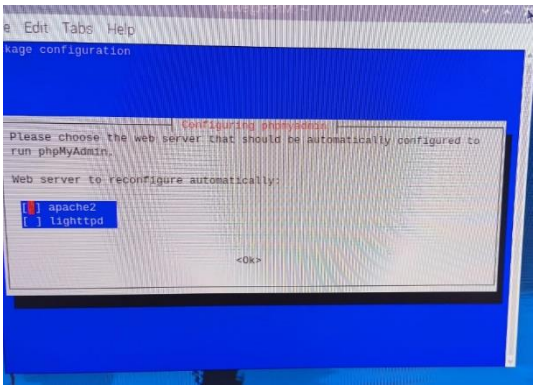


Figura 52: Selección de apache2

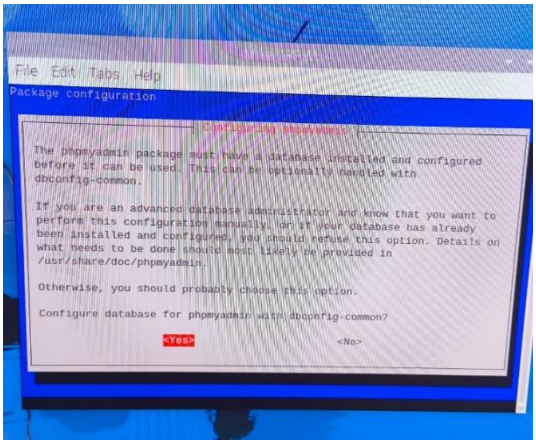


Figura 53: Mensaje numero 2

Se usa el comando *sudo ln -s /usr/share/phpmyadmin /var/www/html/phpmyadmin* y se reinicia Apache2 con el comando *sudo systemctl restart apache2*.

Para ingresar al gestor escribes en el navegador *<IPdeRaspberry>/phpmyadmin* aparecerá un inicio de sesión donde se ingresa con el usuario de MariaDB creado (Véase figura 54).

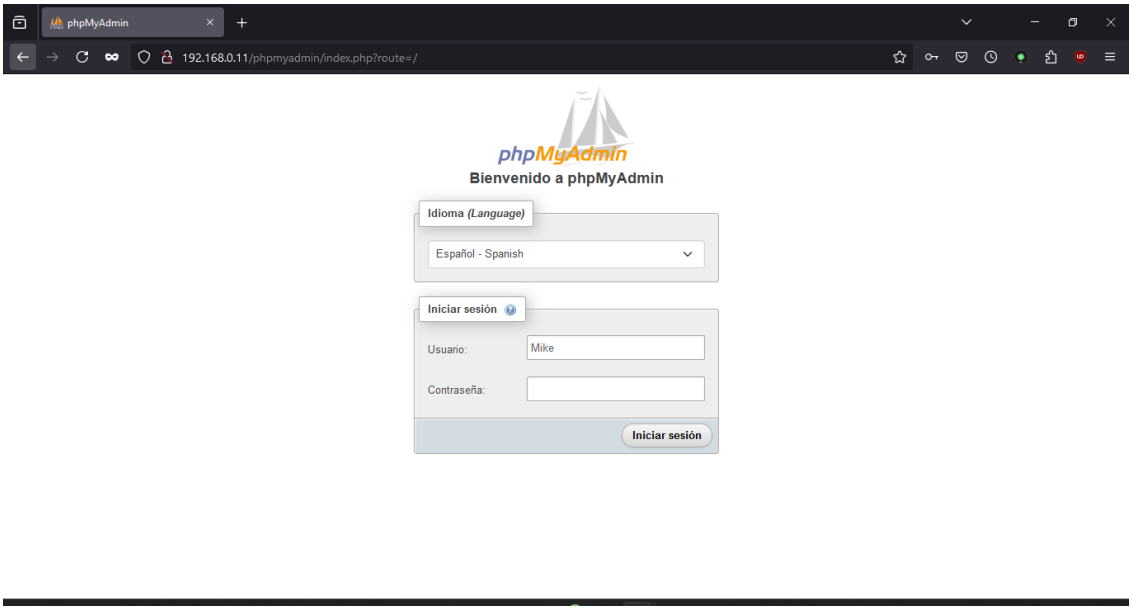


Figura 54: Inicio de sesión phpmyadmin

Dentro del gestor están las bases de datos (Véase figura 55), también dando clic en la base de datos aparecen las tablas y abajo puedes crear una nueva poniendo nombre y la cantidad de columnas y al darle al botón crear te llevara al gestor de tablas (Véase figura 56 y 57), dentro del gestor al terminar de poner las tablas puedes ver su código en SQL presionando el botón de abajo (Véase figura 58), además, también puedes ejecutar directamente el código SQL (Véase figura 59).

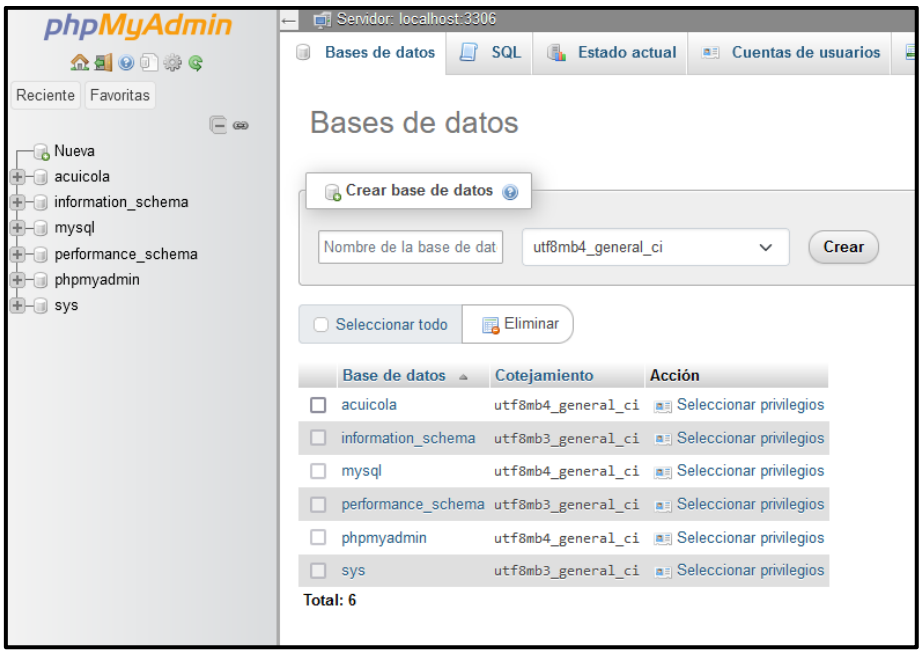


Figura 55: Bases de datos en phpmyadmin

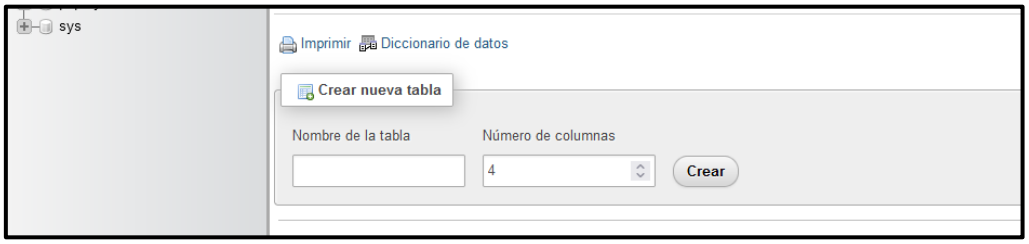


Figura 56: Tablas en phpmyadmin

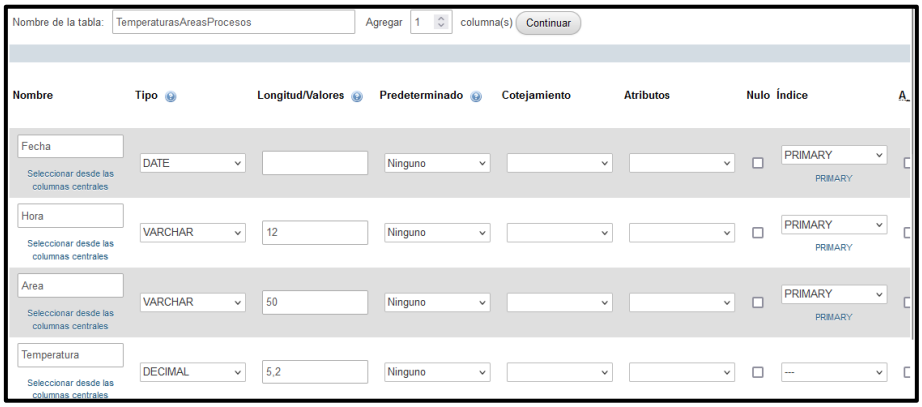


Figura 57: Gestor de tablas phpmyadmin

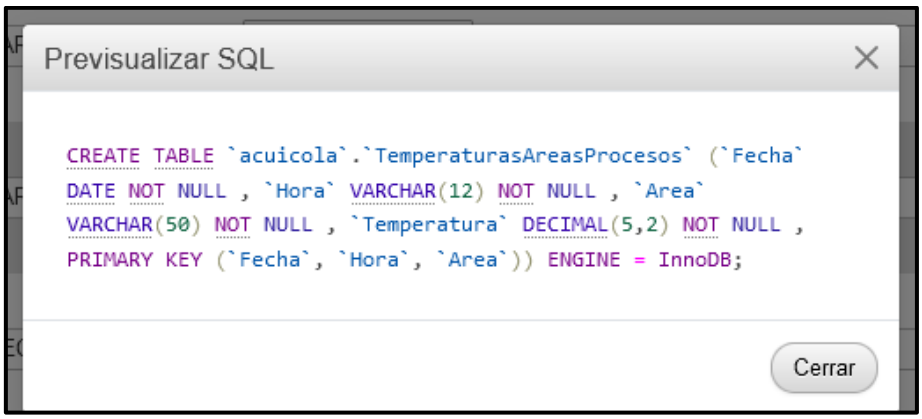


Figura 58: Previsualización código SQL

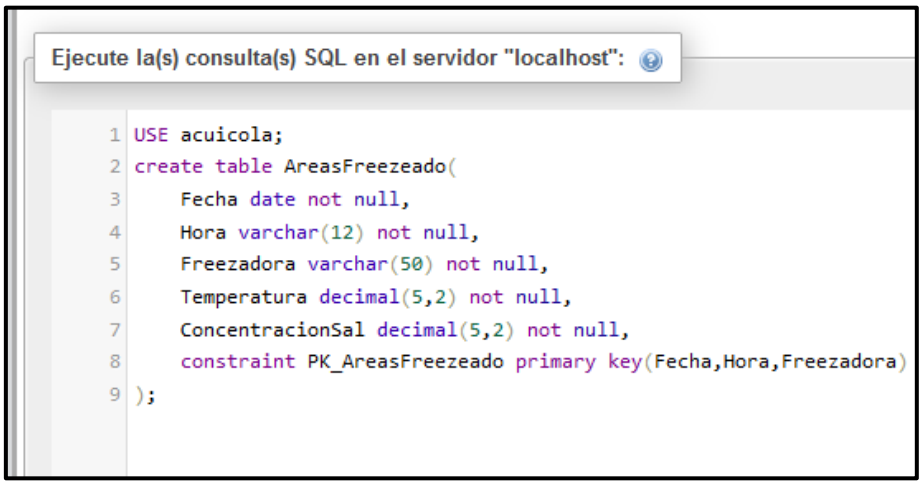


Figura 59: Uso de SQL

Desarrollar una aplicación de escritorio

Para el diseño de la aplicación de escritorio se optó por algo simple, usando solamente los componentes que otorga Windows Forms, en una pantalla se diseñó con: un DataGridView para mostrar los datos que se encuentra en la parte de abajo y dentro de un panel para dar textura, 5 radio Button dentro de un GroupBox para que no existan más de una opción marcada, cada radio Button será una opción para marcar los datos que se desean recibir, habrá un Label a un lado del GroupBox para mostrar el texto de la opción señalada, también tendrá un DateTimePicker con la propiedad Format en Custom y la propiedad CustomFormat en “yyyy-MM-dd” para que se utilice una fecha corta y sencilla de interpretar y también habrán dos botones uno para que cuando sea presionado exporte.

En la parte inicial del código se inicializará un objeto de manipulación de datos y un objeto de DataTable, también dentro del constructor se tendrá inicializado el área de freezeado y se mandaran traer datos (Véase figura 60).

```
ManipulacionDatos objeto = new ManipulacionDatos();
DataTable datos = new DataTable();
1 referencia
public Inicio()
{
    InitializeComponent();
    RDB_AreasFreezzado.Checked = true;

    if (RDB_AP.Checked)
    {
        LBL_Titulo.Text = RDB_AP.Text;
    }
    if (RDB_AreasFreezzado.Checked)
    {
        LBL_Titulo.Text = RDB_AreasFreezzado.Text;
    }
    if (RDB_DMetales.Checked)
    {
        LBL_Titulo.Text = RDB_DMetales.Text;
    }
    if (RDB_TAyP.Checked)
    {
        LBL_Titulo.Text = RDB_TAyP.Text;
    }
    if (RDB_TP.Checked)
    {
        LBL_Titulo.Text = RDB_TP.Text;
    }
    traerDatos();
}
```

Figura 60. Parte inicial del código de la aplicación de escritorio

Cuando se cambie la fecha del DateTimePicker se ejecutará el evento ValueChanged del mismo y se ejecutara la función “TraerDatos()” (Véase figura 61).

```
1 referencia
private void dateTimePicker1_ValueChanged(object sender, EventArgs e)
{
    traerDatos();
}
3 referencias
```

Figura 61. Evento ValueChanged del DateTimePicker

Al momento de seleccionar un área se ejecutará el evento CheckedChanged donde se pondrá el nombre del área en un texto y se mandará llamar la función “TraerDatos()” (Véase figura 62).

```
private void CheckedChanged(object sender, EventArgs e)
{
    if (RDB_AP.Checked)
    {
        LBL_Titulo.Text = RDB_AP.Text;
    }
    if (RDB_AreasFreezzado.Checked)
    {
        LBL_Titulo.Text = RDB_AreasFreezzado.Text;
    }
    if (RDB_DMetales.Checked)
    {
        LBL_Titulo.Text = RDB_DMetales.Text;
    }
    if (RDB_TAyP.Checked)
    {
        LBL_Titulo.Text = RDB_TAyP.Text;
    }
    if (RDB_TP.Checked)
    {
        LBL_Titulo.Text = RDB_TP.Text;
    }
    traerDatos();
}
```

Figura 62. Evento CheckedChanged

La función “TraerDatos()” primero verifica cual área o radio Button esta seleccionado, y dependiendo del área seleccionada es la función que mandara en ManipularDatos que ejecutara, metiendo la tabla que devuelve la función dentro del DataTable y al finalizar el código el DataSource del DataGridView toma la tabla recibida (Véase figura 63).

```
private void traerDatos()
{
    if (RDB_AP.Checked)
    {
        datos = objeto.ObtenerTemperaturaAreasProceso(dateTimePicker1.Text);
    }
    if (RDB_AreasFreezzado.Checked)
    {
        datos = objeto.ObtenerAreasFreezado(dateTimePicker1.Text);
    }
    if (RDB_DMetales.Checked)
    {
        datos = objeto.ObtenerDeteccionMetales(dateTimePicker1.Text);
    }
    if (RDB_TAyP.Checked)
    {
        datos = objeto.ObtenerTemperaturaAguaYProducto(dateTimePicker1.Text);
    }
    if (RDB_TP.Checked)
    {
        datos = objeto.ObtenerTemperaturaProductos(dateTimePicker1.Text);
    }
    DTGV_Datos.DataSource = datos;
}
```

Figura 63. Función TraerDatos

El botón llamado PDF en su evento Click invoca un SaveFieldDialog para asignar la ruta donde se guardará el archivo, el PDF en este caso, y al final toma el título del área de los datos tomados, y los datos tomados los guarda en una tabla y los envía como parámetros a la función para generar el PDF (Véase figura 64).

```
private void button1_Click(object sender, EventArgs e)
{
    if (datos.Rows.Count > 0)
    {
        using (SaveFileDialog saveFileDialog = new SaveFileDialog())
        {
            saveFileDialog.Filter = "PDF files (*.pdf)|*.pdf|All files (*.*)|*.*";
            saveFileDialog.FilterIndex = 1;
            saveFileDialog.RestoreDirectory = true;
            saveFileDialog.FileName = dateTimePicker1.Text + "--" + LBL_Titulo.Text;
            if (saveFileDialog.ShowDialog() == DialogResult.OK)
            {
                string headerText = LBL_Titulo.Text;
                DataTable dataTable = (DataTable)DTGV_Datos.DataSource;
                string filePath = saveFileDialog.FileName;
                GeneratePdf(headerText, dataTable, filePath);
            }
        }
    }
}
```

Figura 64. Evento Click del botón PDF

La función para generar el PDF comienza inicializando la licencia a utilizar, crea el documento siguiendo la sintaxis de la documentación, inicializa el tamaño y los márgenes de la hoja del documento, asigna primero el encabezado tomando el parámetro enviado, para poner los datos, utiliza la tabla que se le envió en los parámetros y mediante un forEach toma las columnas y les asigna su nombre correspondido, después en otro forEach toma los datos de las filas de la tabla y son

plasmados en sus respectivas columnas, al final se le manda un mensaje al usuario que se generó correctamente el PDF (Véase figura 65).

```
1 referencia
private void GeneratePdf(string headerText, DataTable dataTable, String filePath)
{
    QuestPDF.Settings.License = LicenseType.Community;

    Document.Create(container =>
    {
        container.Page(page =>
        {
            page.Size(PageSizes.A4);
            page.Margin(1, Unit.Centimetre);

            page.Header()
                .Text(headerText)
                .FontSize(30)  // Tamaño grande para el encabezado
                .FontFamily("Arial")  // Fuente Arial
                .Bold()
                .FontColor(Colors.Red.Medium)
                .AlignCenter();  // Centrar el encabezado

            page.Content().Column(column =>
            {
                column.Spacing(8);
                column.Item().PaddingVertical(20);
                // Agregar encabezados de las columnas
                column.Item().AlignCenter().Row(row =>
                {
                    // Centrar la fila
                    foreach (DataColumn col in dataTable.Columns)
                    {
                        row.RelativeItem().Text(col.ColumnName)
                            .FontFamily("Arial")  // Fuente Arial
                            .Bold()
                            .FontSize(15)
                            .AlignCenter();  // Centrar el texto
                    }
                });

                // Agregar datos de las filas
                foreach (DataRow dataRow in dataTable.Rows)
                {
                    column.Item().AlignCenter().Row(row =>
                    {
                        // Centrar la fila
                        foreach (var cell in dataRow.ItemArray)
                        {
                            row.RelativeItem().Text(cell.ToString())
                                .FontFamily("Arial")  // Fuente Arial
                                .FontSize(10)
                                .AlignCenter();  // Centrar el texto
                        }
                    });
                }
            });
        });
    }).GeneratePdf(filePath);

    MessageBox.Show("PDF generado correctamente.");
}
```

Figura 65. Función GeneratePDF()

El botón llamado Excel en su evento Click invoca un SaveFieldDialog para asignar la ruta donde se guardará el archivo, xlsx en este caso, y al final toma el título del área de los datos tomados, y los datos tomados los guarda en una tabla y los envía como parámetros a la función para generar el Excel (Véase figura 66).

```
private void button2_Click_1(object sender, EventArgs e)
{
    if (datos.Rows.Count > 0)
    {
        using (SaveFileDialog saveFileDialog = new SaveFileDialog())
        {
            saveFileDialog.Filter = "Excel files (*.xlsx)|*.xlsx|All files (*.*)|*.*";
            saveFileDialog.FilterIndex = 1;
            saveFileDialog.RestoreDirectory = true;
            saveFileDialog.FileName = dateTimePicker1.Text + "--" + LBL_Titulo.Text;

            if (saveFileDialog.ShowDialog() == DialogResult.OK)
            {
                string headerText = LBL_Titulo.Text;
                DataTable dataTable = (DataTable)DTGV_Datos.DataSource;
                string filePath = saveFileDialog.FileName;
                GenerateExcel(headerText, dataTable, filePath);
            }
        }
    }
}
```

Figura 66. Evento Click del botón Excel

La función para generar el Excel comienza poniendo el título que toma de los parámetros enviados, luego mediante un For agrega el nombre de las columnas de la tabla (Véase figura 67).

```
private void GenerateExcel(string headerText, DataTable dataTable, string filePath)
{
    // Crear un nuevo libro de Excel
    using (var workbook = new XLWorkbook())
    {
        // Agregar una hoja al libro
        var worksheet = workbook.Worksheets.Add("Reporte");

        // Agregar el encabezado
        worksheet.Cell(1, 1).Value = headerText;
        worksheet.Cell(1, 1).Style.Font.Bold = true;
        worksheet.Cell(1, 1).Style.Font.FontSize = 30;
        worksheet.Cell(1, 1).Style.Font.FontName = "Arial";
        worksheet.Cell(1, 1).Style.Alignment.Horizontal = XLAlignmentHorizontalValues.Center;
        worksheet.Range(1, 1, 1, dataTable.Columns.Count).Merge(); // Unir celdas para el encabezado

        // Agregar encabezados de columna
        for (int i = 0; i < dataTable.Columns.Count; i++)
        {
            worksheet.Cell(2, i + 1).Value = dataTable.Columns[i].ColumnName;
            worksheet.Cell(2, i + 1).Style.Font.Bold = true;
            worksheet.Cell(2, i + 1).Style.Font.FontSize = 15;
            worksheet.Cell(2, i + 1).Style.Font.FontName = "Arial";
            worksheet.Cell(2, i + 1).Style.Alignment.Horizontal = XLAlignmentHorizontalValues.Center;
            worksheet.Cell(2, i + 1).Style.Fill.BackgroundColor = XLColor.LightGray;
        }
```

Figura 67. Parte 1 Función GenerateExcel

Mediante un For doble comienza a agregar la información de las filas dentro de las respectivas columnas, y al finalizar se guarda en la ubicación seleccionada anteriormente y se envía un mensaje (Véase figura 68).

```
        // Agregar datos de las filas
        for (int i = 0; i < dataTable.Rows.Count; i++)
        {
            DataRow fila = dataTable.Rows[i];
            for (int j = 0; j < dataTable.Columns.Count; j++)
            {
                worksheet.Cell(i + 3, j + 1).Value = fila[j].ToString();
                worksheet.Cell(i + 3, j + 1).Style.Font.FontSize = 10;
                worksheet.Cell(i + 3, j + 1).Style.Font.FontName = "Arial";
                worksheet.Cell(i + 3, j + 1).Style.Alignment.Horizontal = XLAlignmentHorizontalValues.Center;
            }
        }

        // Ajustar el ancho de las columnas al contenido
        worksheet.Columns().AdjustToContents();

        // Guardar el archivo de Excel en la ruta seleccionada
        workbook.SaveAs(filePath);
    }

    MessageBox.Show("Archivo Excel generado correctamente.");
}
```

Figura 68. Parte 2 Función GenerateExcel

Análisis del impacto de la mejora

Para analizar el impacto, primero se analizó la situación o las actividades sin las aplicaciones, donde se hizo una toma de tiempo aproximado de cuanto le costaba desarrollar cada actividad, desde la toma de datos hasta la búsqueda de los documentos, también se tomó en cuenta los daños de los documentos, los atrasos y complicaciones en la búsqueda de los formatos correctos o de los necesarios para cada situación, y los errores normalmente cometidos por humanos.

Al implementarlo se tomaron de nuevo tiempo y se evaluaron los mismos aspectos que los anteriores, se tomó el tiempo de registro, guardado, y búsqueda de los datos, se examinó la probabilidad y la eficiencia, la probabilidad de errores como la confusión de datos, o humanos, y al ser todo digital el daño a la información es prácticamente inexistente.

Análisis de resultados y Discusión

Toma y análisis de requisitos

Al concluir la toma y análisis de los requisitos se diseñó las tablas requeridas en el sistema (véase figura 69), así como también partiendo de eso se pudieron interpretar la cantidad de pantallas y que elementos serian requeridos para la aplicación móvil, así mismo, partiendo de eso se dio un enfoque para la decisión de los elementos necesarios para generar los reportes.

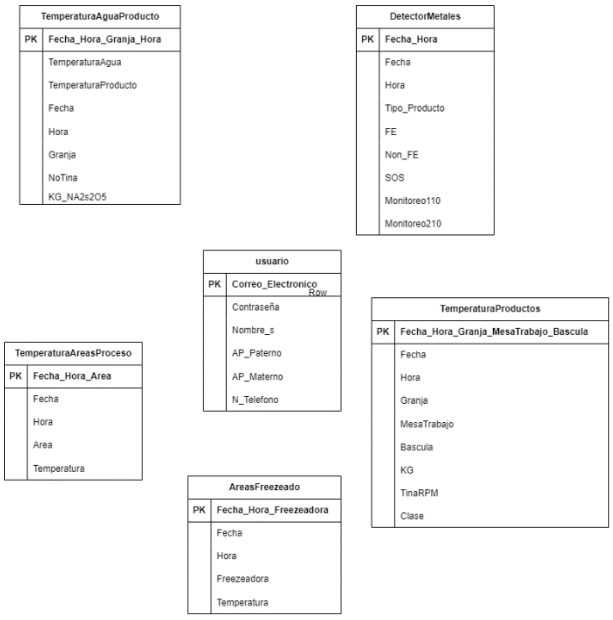


Figura 69: Diseño de tablas

Desarrollo de una aplicación móvil

Al concluir el desarrollo de la aplicación movil, se obtuvieron los siguientes resultados.
El Login o inicio de sesión de la aplicación con colores calmados y un logo referente a la misma (Véase figura 70).

Figura 70: Login

El registro de usuario se manejaron los datos de Nombre(s), apellido materno, apellido paterno, Numero de celular, correo electrónico y contraseña (Véase figura 71).
En caso de no ser valida la contraseña se enviará un mensaje como debería estar estructurada la contraseña (Véase figura 72).
Al momento de agregar el usuario en caso de que exista saltara un mensaje de error (Véase figura 73).

Figura 71: Pantalla Registrar usuario

Figura 72: Mensaje Contraseña invalida

Figura 73 Mensaje de usuario existente

El menú solo maneja los botones y colores muy neutros (Véase figura 74).

Figura 74: Menú

En las pantallas de registro de datos se utilizan colores poco hostigosos, no demasiado oscuros, pero tampoco demasiado brillantes para no molestar tanto la vista, se utilizan

esos colores de manera secundaria dejando un color blanco en casi toda la pantalla (Véase figura 75), en las pantallas para ver los datos, el color principal si se utiliza de forma más dominante de fondo de la pantalla (Véase figura 76).

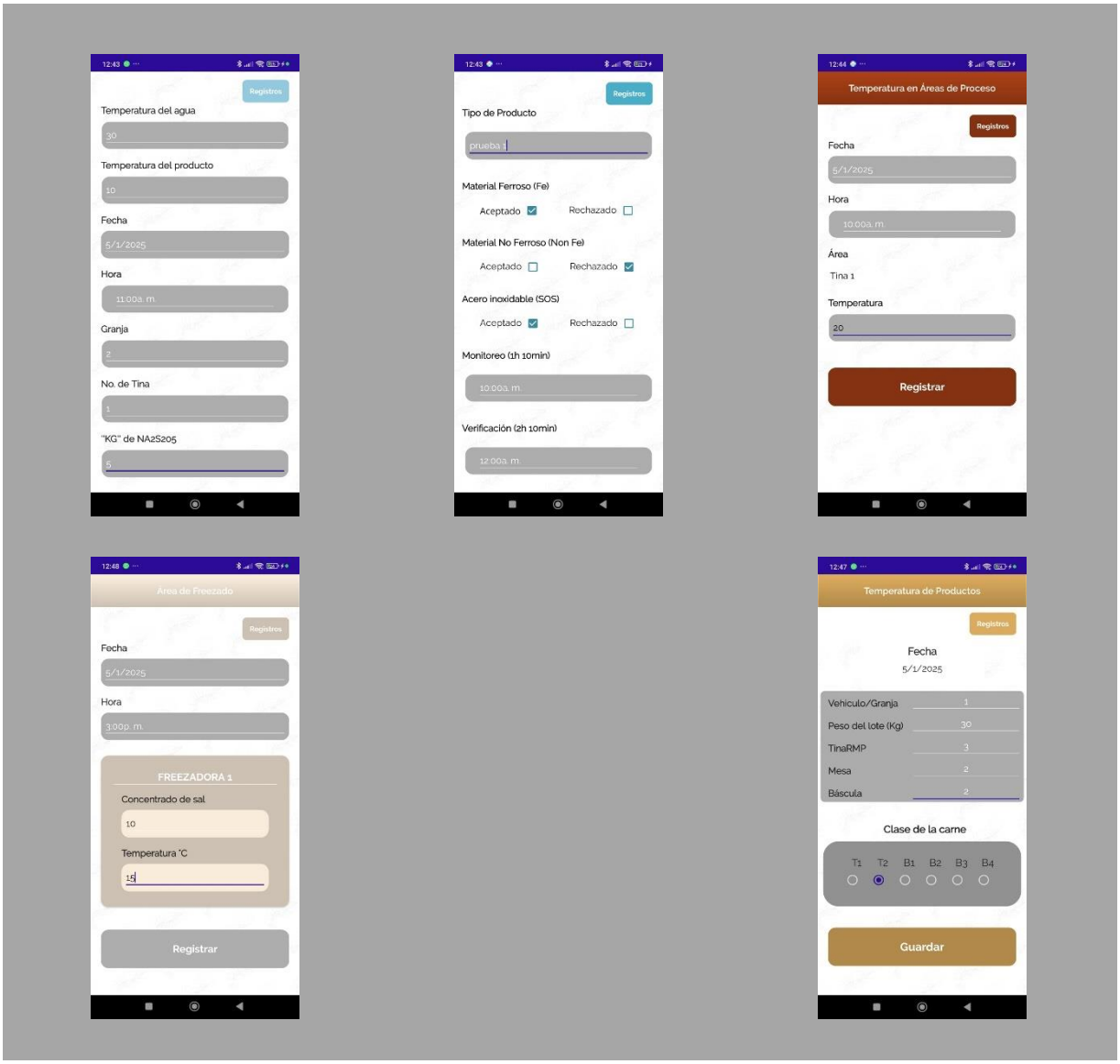


Figura 75: Pantallas Registrar datos

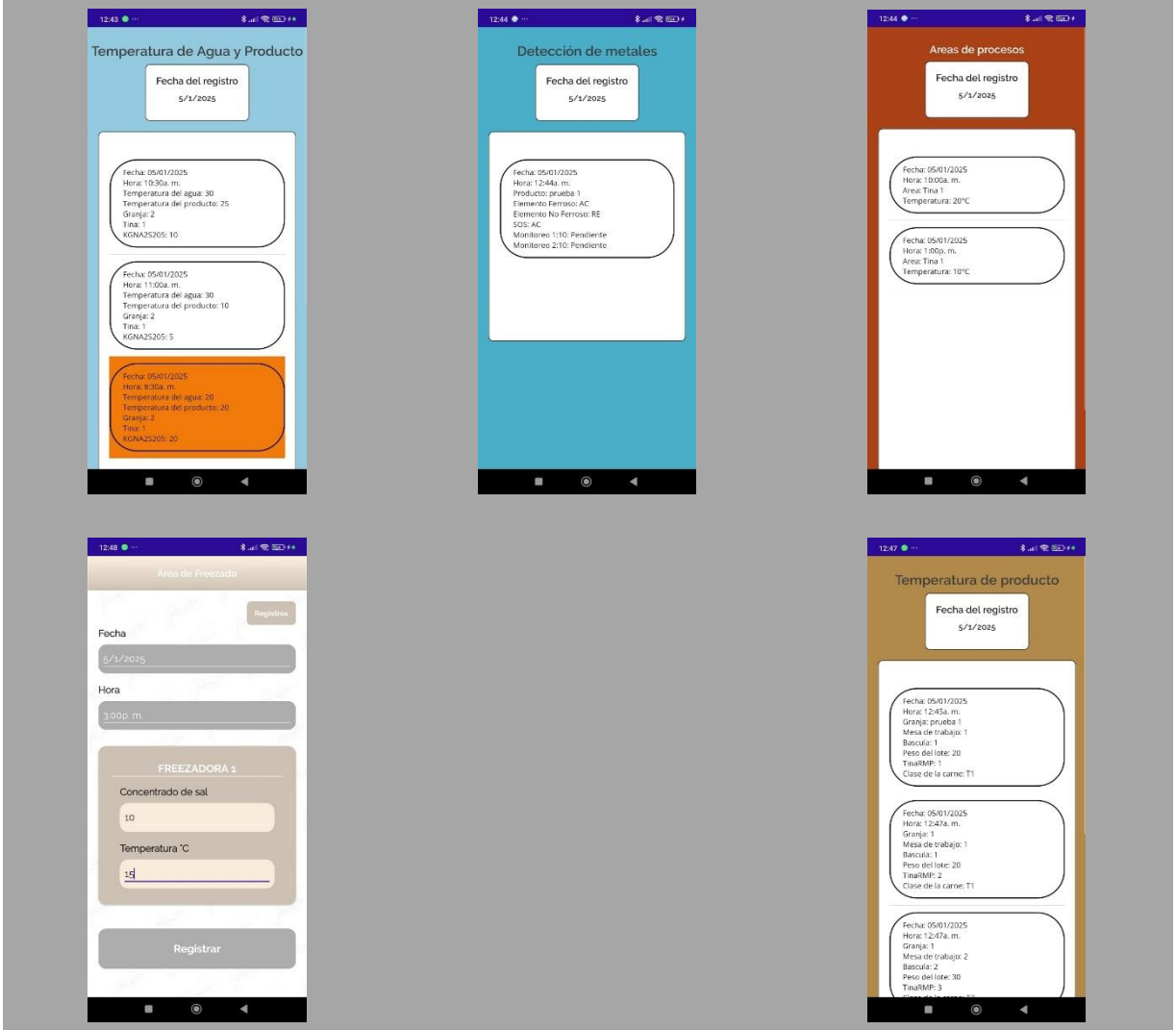


Figura 76: Pantallas muestra de datos

Implementar una base de datos en una Raspberry Pi

Después de implementar la base de datos en la Raspberry pi en el gestor de phpMyAdmin aparecerá en el apartado de la base de datos las tablas creadas con éxito (Véase figura 77) al presionar el '+' en el menú de la izquierda esta la información de la tabla, o bien se puede seleccionar alguna tabla directamente en este apartado.

Figura 77: Tablas de la base de datos

Desarrollar una aplicación de escritorio

La aplicación de escritorio se basaba en recibir los datos de las áreas requeridas y poder imprimir reportes tanto en PDF como Excel en el momento requerido (Véase figura 78-82).

Figura 78: Áreas Frezado Aplicación escritorio

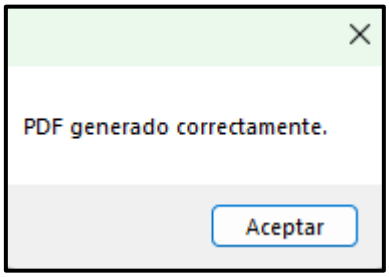


Figura 79: Mensaje éxito PDF creado

Figura 80: PDF creado

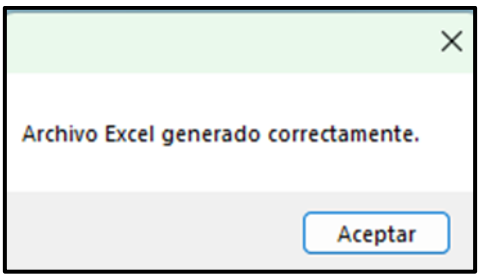
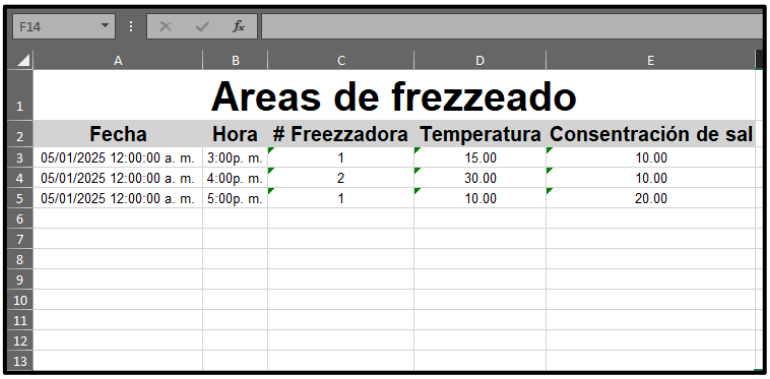


Figura 81: Mensaje Excel creado



	A	B	C	D	E
1	Areas de frezzeado				
2	Fecha	Hora	# Freezzadora	Temperatura	Consentración de sal
3	05/01/2025 12:00:00 a. m.	3:00p. m.	1	15.00	10.00
4	05/01/2025 12:00:00 a. m.	4:00p. m.	2	30.00	10.00
5	05/01/2025 12:00:00 a. m.	5:00p. m.	1	10.00	20.00
6					
7					
8					
9					
10					
11					
12					
13					

Figura 82: Excel creado

Análisis del impacto de la mejora

El tiempo medio para la toma de datos registrado a mano fue de 30 segundos, mientras que con la aplicación fue de 20 a 25 segundos, el tiempo de guardado de datos varía entre 30 segundos y 1 minutos dependiendo el área, mediante la aplicación el guardado es prácticamente instantáneo tomando máximo 5 segundos dependiendo la estabilidad de la red, la búsqueda de algún formato o datos guardados es más extenso llegando a ser desde 1 minuto hasta posibles 10 minutos, además de poder ser confundidos y regresar hacer otra búsqueda por el formato correcto o los datos correctos, mediante la aplicación el porcentaje de error es disminuido notablemente y en caso de ser cometido el error, el tiempo que se emplea en la búsqueda y re búsqueda son altamente disminuidos en un margen de 20 segundos a máximo 1 minutos, en caso de equivocarse de documento la re búsqueda es prácticamente el mismo tiempo.

La generación de reportes en formatos PDF y Excel es otra función de gran ayuda, la aplicación de escritorio permite generar estos reportes de forma automática, asegurando precisión y reduciendo el margen de error humano para su obtención física con mayor seguridad. También la digitalización del proceso permite un mayor seguimiento de la información, garantizando su seguridad y disponibilidad en cualquier momento. Esto no solo mejora la toma de decisiones basada en datos precisos, sino que también fortalece la confiabilidad y la auditoría de los procesos.

Conclusiones y recomendaciones

Una vez elaborado el proyecto se pudo llegar a la conclusión que la implementación de tecnologías computacionales para gestión del proceso de producción de una empresa acuícola haría un trabajo más efectivo y cómodo, tanto el proceso de registro de datos, almacenamiento de los formatos, y la búsqueda o consulta de los mismos se vieron afectados en gran medida, no solo en su tiempo de ejecución si no también en la disminución de errores humanos como también en una mayor protección o seguridad en los mismos. El tiempo del registro de datos se vio reducido un tiempo casi inexistente, reduciendo de 30 segundos el registro a mano, a 25 segundos con la aplicación, aun así, la comodidad del registro evitando la carga de documentos a cada área y disminuyendo el riesgo de ser dañados los documentos en su estancia en cada área da una mayor seguridad y disminuye en cierta medida la carga de trabajo del/la responsable de la actividad.

El almacenamiento de los datos registrados también se vio afectado de forma más notoria al registro, ya que antes después de ser plasmados en cada formato debían ser guardados en respectivas carpetas, cosa que con la implementación de las tecnologías computacionales se eliminó, y es un proceso prácticamente ahorrado al ser casi instantáneo el almacenamiento dentro de la base de datos en la Raspberry pi, no solo el tiempo fue una característica mejorada, sino también la seguridad o la protección de los datos, ya que al ser almacenados directamente en la base de datos al momento de ser tomados no se corre el riesgo de extravió o daños en los documentos.

La búsqueda o consulta de datos es considerablemente la que mejor se vio afectada, ya que se vio reducido en gran medida el tiempo de búsqueda a un tiempo casi inmediato, además, al momento de sacarlos evitando por completo ser dañados, además que si es necesario tenerlos en físico se podía imprimir el reporte en PDF o utilizar el archivo en Excel para otro procedimiento.

La implementación de las tecnologías computacionales ha llegado a generar mayor seguridad, eficiencia, y seguimiento de los datos comparado a con la utilización de formatos de papel, tomando en cuenta el estado y la eficacia del desarrollo, se

recomienda la utilización de Objetos IoT para un mejoramiento de procesos, medidores de temperatura en las diferentes áreas para ser monitoreados constantemente y notificar algún cambio repentino, así como en el área de detección de metales u otras áreas que requieren un monitoreo más regulado o donde se puede tener un monitoreo más constante. El mejoramiento de las capacidades utilizando las mismas tecnologías computacionales son muchas.

REFERENCIAS

Bidart, M. (2023, octubre 23). *Productos digitales, interfaces—UX/UI*. Repositorio institucional de la unlp. https://sedici.unlp.edu.ar/handle/10915/161396

britch, D., Bughiu, M., & Verhagen, J. (2024, agosto 30). Controles. *..NET MAUI. Microsoft Learn.* https://learn.microsoft.com/es-es/dotnet/maui/user-interface/controls/?view=net-maui-9.0

britch, D., Rastegarinia, M. H., Pryor, J., Gechev, I., & jconrey. (2024, junio 21). ¿Qué es .NET MAUI? *..NET MAUI. Microsoft Learn.* https://learn.microsoft.com/es-es/dotnet/maui/what-is-maui?view=net-maui-8.0

Caballero, E. R. (2023, abril 25). El superpoder invisible del diseño accesible. *480*. https://cuatroochenta.com/el-superpoder-invisible-del-diseno-accesible/

CoveríX. (2024, septiembre 11). 5 Beneficios de las estrategias de tecnología en una empresa. *CoveríX*. https://blog.coverix.mx/beneficios-estrategias-de-tecnologia-empresarial?

Godoy, S. (2024, agosto 8). Ventajas de la informática. *SAINT LEO UNIVERSITY*. https://worldcampus.saintleo.edu/blog/ventajas-informatica-empresas?

Gonzalez, S. (2023, diciembre 1). Creadores de aplicaciones de Android: Gestión de bases de datos. *AppMaster*. https://appmaster.io/es/blog/gestion-de-bases-de-datos-de-creadores-de-aplicaciones-de-android

InnoQubit. (s/f). Ventajas de la digitalizacion de una empresa. *InnoQubit*. Recuperado el 2 de enero de 2025, de https://www.innoqubit.com/blog/ventajas-de-la-digitalizacion-de-una-empresa/?

Khan, F. (2025, enero 24). Una guía completa para el análisis de datos. *astera*. https://www.astera.com/es/type/blog/data-analytics/

Lucena, P. (2023, mayo 6). *¿Qué es el framework?* Universidad CESUMA. https://www.cesuma.mx/blog/que-es-el-framework.html

MariaDB.org. (2023, junio 19). MariaDB. *MariaDB en resumen*. https://mariadb.org/es/

Mucci, T. (s/f). Toma de decisiones basada en datos. *IBM*. Recuperado el 3 de febrero de 2025, de https://www.ibm.com/mx-es/think/topics/data-driven-decision-making

Navarro, S. (2024, junio 27). ¿Qué son los motores de bases de datos? *¿Qué son los motores de bases de datos?* https://keepcoding.io/blog/que-son-los-motores-de-bases-de-datos/

oracle. (s/f-a). ¿Qué es la gestión de datos? *Oracle mexico*. Recuperado el 1 de enero de 2025, de https://www.oracle.com/mx/database/what-is-data-management/

oracle. (s/f-b). ¿Qué es MySQL? *Oracle mexico*. Recuperado el 29 de diciembre de 2024, de https://www.oracle.com/mx/mysql/what-is-mysql/

Paguayo. (2022, abril 26). ¿Qué es Rasberry Pi? *Raspberry Pi*. https://raspberrypi.cl/que-es-raspberry/

Platzi. (s/f). Tipos de datos MySQL/MariaDB. *Curso de Bases de Datos con MySQL y MariaDB*. Recuperado el 30 de diciembre de 2024, de https://platzi.com/tutoriales/4203-mysql-mariadb/20970-tipos-de-datos-en-mysqlmariadb/

Psicologia del color. (2020, enero 28). Los colores en las Aplicaciones Móviles. *Los colores en las Aplicaciones Móviles*. https://www.psicologiadelcolor.es/articulos/colores-aplicaciones-moviles/

Rodriguez, R. A., Vera, P. M., Giulianelli, D. A., & Cammarano, P. (2017, enero 10). *Implementación con Raspberry PI de un servidor portátil de contenidos*. Repositorio institucional de la unlp. https://sedici.unlp.edu.ar/handle/10915/63854

Serna, S. (2016). *Diseño de interfaces en aplicaciones moviles* (1a ed.). RA-MA.

Servicios Empresariales. (2024, agosto 2). 9 beneficios de la digitalización en las empresas. *Camara Madrid Servicios Empresariales*. https://serviciosempresariales.camaramadrid.es/9-beneficios-digitalizacion-empresas/?

Soporte técnico de Microsoft. (s/f). Que es una base de datos. *¿Qué es una base de datos?* Recuperado el 27 de diciembre de 2024, de https://support.microsoft.com/es-es/topic/conceptos-b%C3%A1sicos-sobre-bases-de-datos-a849ac16-07c7-4a31-9948-3c8c94a7c204#__toc257378454

Vásquez-Quispesivana, W., Inga, M., & Betalleluz-Pallardel, I. (2022). *Inteligencia artificial en acuicultura: Fundamentos, aplicaciones y perspectivas futuras*. http://dx.doi.org/10.17268/sci.agropecu.2022.008