# ESCUELA POLITÉCNICA NACIONAL

## FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

## Sistema de vigilancia para conductores privados sobre uso del vehículo.

### Implantación de un sistema de almacenamiento de archivos de secuencias de imágenes con estampa de tiempo y de localización (utilizando el formato de un módulo GPS) para un vehículo.

**TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TELECOMUNICACIONES**

**JONATAN ROMARIO PILICITA QUISHPE**

**jonatan.pilicita@epn.edu.ec**

**DIRECTOR: Ing. Ramiro Eduardo Morejón Tobar M.Sc.**

**ramiro.morejon@epn.edu.ec**

**DMQ, abril 2024**

## CERTIFICACIONES

Yo, JONATAN ROMARIO PILICITA QUISHPE declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

_____

**JONATAN ROMARIO PILICITA QUISHPE**

Certifico que el presente trabajo de integración curricular fue desarrollado por JONATAN ROMARIO PILICITA QUISHPE, bajo mi supervisión.

_____

**M.Sc. RAMIRO EDUARDO MOREJÓN TOBAR**

**DIRECTOR**

## DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

JONATAN ROMARIO PILICITA QUISHPE

M.Sc. RAMIRO EDUARDO MOREJÓN TOBAR

## DEDICATORIA

Dedico este trabajo de titulación a Dios por permitirme seguir adelante pese a todas las situaciones conllevadas en el trascurso de mi vida. Como dijo Viktor Frankl, en El hombre en búsqueda del sentido, “Los hombres ilustres siempre recomienzan, y eso los convierte en admirables e imitables” [1]. A mi madre Rebeca que siempre me sostuvo cuando más lo necesite, a mi padre Carlos por ser una persona ejemplar y bondadosa, a mis hermanos Yadira y Carlos por ser esa inspiración de siempre ser mejor.

**Jonatan Pilicita**

## AGRADECIMIENTO

Agradezco a mis padres, Carlos y Rebeca, que siempre estuvieron brindándome el apoyo incondicional para terminar esta hermosa etapa de mi vida; a mis hermanos, Yadira y Junior, por ser esas personas que guiaron mi camino siendo buenas personas y excelentes profesionales; a Esteban, por ser un buen amigo que me alentó a seguir adelante pese a la distancia que se encuentra; a Roddy, por ser un amigo incondicional desde los inicios universitarios; a Johnny, por compartir sus conocimientos y ser una excelente persona; a Jhon, por ser un buen amigo y compañero. Te agradezco mucho por ayudarme en esa etapa que fue mi rotura de tendón de Aquiles, gracias por "sostenerme"; a mis Ortopedistas Sarita y Eduardo por buscar mi bienestar; gracias, mi hermosa Poli, por compartir con personas extraordinarias y formarme como un buen ser humano y profesional.

Mis agradecimientos para el Dr. Hernán Barba por permitirme ser parte de su equipo de trabajo en la mejor Facultad de la EPN, así como al Ingeniero Marco Serrano por su calidez como persona y profesional. Agradezco al M.Sc. Ricardo Mena por sus pautas al momento de la formulación del proyecto, además, agradecerles por todas las explicaciones y enseñanzas que pude adquirir cuando fui parte del equipo del Departamento de Circuitos y Dispositivos Electrónicos.

Finalmente, agradecer a todos los profesores que tuve en mi etapa universitaria, quienes han aportado de su conocimiento y enseñanzas para poder desenvolverme como un buen profesional.

***Jonatan Pilicita***

# ÍNDICE DE CONTENIDO

## ÍNDICE DE FIGURAS

## ÍNDICE DE TABLAS

# ÍNDICE DE CÓDIGOS

## RESUMEN

En el presente documentó se describirá el sistema de vigilancia para conductores privados sobre uso del vehículo, este sistema permitirá al usuario el monitoreo de rutas realizadas en diferentes zonas geográficas, mediante el sistema de almacenamiento de archivos de secuencias de imágenes con estampa de tiempo y de localización del vehículo, se podrá acceder a la información deseada mediante una base de datos no relacional, la cual permitirá el manejo de grandes cantidades de volúmenes de datos utilizando un sistema de almacenamiento masivo.

En el primer término se presentarán los objetivos generales, específicos, alcance y el marco teórico para el desarrollo del presente trabajo de integración curricular (TIC). Del análisis de los requerimientos se determinará las características de la plataforma de sistema embebido. Se considerará una plataforma que soporte un sistema operativo que cumplan con los requerimientos que permita administrar los archivos con la base de datos no relacional. Se describirá las herramientas utilizadas para el desarrollo del sistema de gestión de archivos. Para el desarrollo de las aplicaciones de TIC se utilizará la ingeniería de requerimientos.

Una vez establecido los requerimientos se diseñará el sistema de gestión de archivos de secuencia de imágenes. Se implementará las fases de la ingeniería de requerimientos para dar seguimiento la construcción de la aplicación.

Las pruebas se realizarán utilizando archivos de imágenes con estampa de tiempo y ubicación los cuales serán generados mediante un script de Python. Los archivos así generados serán almacenados utilizando el formato “Bson” (Binary JSON) [2], en la base de datos dispuesta para el efecto. El formato Bson contiene la ruta de acceso al archivo en el sistema de almacenamiento y los atributos que se utilizaran en el dialogo de la aplicación.

La interfaz Gráfica de administración está constituida por dos módulos y está desarrollado utilizando “Qt Designer” para la construcción de la plantilla, y para sus funcionalidades se utilizará Python.

Seguidamente se realizarán las pruebas operativas del sistema de archivos, Los resultados de las pruebas realizadas se presentarán y se analizará dichos resultados, los que permitirá establecer las conclusiones y recomendación.



**PALABRAS CLAVE:** Sistema Embebido, NoSQL, Python, Interfaz Gráfica.

## ABSTRACT

This document will describe the surveillance system for private drivers regarding vehicle usage. This system will allow the user to monitor routes taken in different geographical areas through the storage system of image sequence files with timestamp and vehicle location. The desired information can be accessed through a non-relational database, which will handle large volumes of data using a massive storage system.

Firstly, the general objectives, specific objectives, scope, and theoretical framework for the development of this curriculum integration work (TIC) will be presented. The characteristics of the embedded system platform will be determined from the requirements analysis. A platform supporting an operating system that meets the requirements to manage files with the non-relational database will be considered. The tools used for the development of the file management system will be described. Requirements engineering will be used for the development of TIC applications.

Once the requirements are established, the sequence image file management system will be designed. The phases of requirements engineering will be implemented to track the construction of the application.

Tests will be carried out using image files with timestamp and location, which will be generated using a Python script. The files thus generated will be stored using the "Bson" (Binary JSON) format [2], in the database arranged for this purpose. The Bson format contains the file access path in the storage system and the attributes that will be used in the application dialog.

The administration graphical interface consists of two modules and is developed using "Qt Designer" for template construction, and Python will be used for its functionalities.

Subsequently, operational tests of the file system will be carried out. The results of the tests conducted will be presented and analyzed, allowing for conclusions and recommendations to be established.



**KEYWORDS:** Embedded System, NoSQL, Python, Graphical Interfac

# 1 INTRODUCCIÓN

Los Sistemas Embebidos en aplicaciones de Internet de la Cosas (IoT) han alcanzado una amplia difusión al punto que el número de dispositivos es comparable con el número de personas en el planeta [3]. Estos sistemas de bajo costo son cada vez más potentes, por esta razón las grandes empresas están usando arreglo de estos dispositivos en su infraestructura con el fin de reducir los costos de implementación y mejorar el desempeño. Estos miniordenadores, de alto rendimiento, son utilizada para automatizar las actividades industriales, y de la vida cotidiana. Realizan tareas tales como: servidores ligeros para alojamiento en sitios web, gestión de archivos de forma local o en la nube, captura de datos en tiempo de ejecución, y manejo de actuadores [3].

Las plataformas de Sistemas Embebidos en el entorno del presente trabajo de titulación se utilizarán como un servidor local cuya funcionalidad será la administración de archivos de imágenes en formato comprimido utilizando un índice de cabecera, en una base de datos no relacional (NoSQL).

La base de datos escogida es de código abierto, por tanto, no tiene costo y se descarga directamente de las fuentes del desarrollador. Para trabajar en la plataforma de sistema embebido se utilizó el código fuente en una compilación nativa realizada por Andy Felong [4]. La forma de estructuración de la base de datos para sus campos de búsquedas se obtendrá de los datos simulados mediante un script de Python correspondiente a un módulo GPS, y se embeberán en el índice de cabecera del archivo comprimido que contiene los atributos de: fecha, hora, ubicación, ruta, y prioridad. Esto campo permitirán al usuario realizar filtros de búsqueda. El campo llamado ruta contendrá la ubicación real de los archivos comprimidos almacenados en la unidad externa de gran capacidad y el campo de prioridad, el usuario podrá marca en un check box asignado la importancia del archivo para posteriormente descargarlo.

Para la Administración de archivos, se prevé la construcción de dos módulos de interfaz gráfica, el primer módulo permitirá la autenticación de los usuarios, los cuales están clasificados en 4 tipos, el segundo módulo permitirá la gestionarlo archivos mediante filtros. Seguidamente se definen los tipos de usuarios:

**Usuario privilegiado (root):** Dispone de todos los permisos del sistema de la base de datos.

**Usuario tipo read:** Tiene permisos de lectura de archivos.

**Usuario tipo redWrite:** Tiene permisos de lectura y escritura de archivos.

**Usuario tipo dbOwner:** Dispone de todos los permisos de administración en un base de datos especifica.

## 1.1 OBJETIVO GENERAL

Desarrollar un prototipo de gestión de archivos de imágenes provenientes de un sistema de vigilancia para conductores que dan servicio a adultos mayores, almacenando los archivos de imágenes en un dispositivo de gran capacidad.

## 1.2 OBJETIVOS ESPECÍFICOS

1. Establecer los requerimientos de la plataforma de sistemas embebidos a utilizarse y del sistema operativo que cumplan las funcionalidades para el uso de base de datos no relacional.
2. Estudiar distintos formatos de archivo de imágenes, así como, las etiquetas de ubicación, fecha y hora para su manipulación mediante scripts desarrollados en Python.
3. Implementar un sistema de almacenamiento de archivos de secuencias de imágenes con estampa de tiempo y de localización del vehículo con capacidades de gestión de archivos mediante una base de datos no relación.
4. Establecer mecanismos de autenticación de usuario para la gestión de archivos mediante filtros, que permitan realizar búsquedas y obtener respaldos.

## 1.3 ALCANCE

En el presente Trabajo de Titulación se utilizará una plataforma de sistemas embebidos con una gran capacidad de almacenamiento que permita almacenar archivo de imágenes para el monitoreo de rutas realizadas por un vehículo, que está destinado al servicio particular, el mismo que es operado por un conductor diferente del propietario del automotor.

El sistema estará implementando, utilizando una plataforma mínima con un sistema operativo con capacidades de manejos de archivos y la ejecución de aplicaciones desarrolladas en Python. Dado que componente contempla de manera exclusiva la creación y validación de usuarios, el almacenamiento y búsquedas, no contempla la captura de secuencias de imágenes o coordenadas de posicionamiento global (GPS), por lo que, las pruebas se realizaran utilizando un conjunto de archivos que simulen los que se obtendrían del componente de captura de imágenes dentro del proyecto.

### A. Diseño del prototipo

Para el diseño se requiere que tanto la plataforma de sistema embebido y su sistema operativo cumpla los requerimientos para trabajar con la base de datos no relacional.

En el análisis se considerarán principalmente las plataformas utilizadas en el desarrollo de la formación de la carrera.

Se estudiará los distintos formatos de archivos de manera que en su cabecera se pueda embeber parámetros tales como: fecha, hora, ubicación, ruta, y prioridad. Para el diseño de los archivos se investigará las distintas funciones en Python. Se establecerá la estructura de la base de datos no relacional de forma que se pueda gestionar el sistema de archivo, así como el entorno de desarrollador para trabajar con las diferentes funciones en Python. Se definirán los distintos programas para la construcción de los módulos de interfaz gráfica de manera que el usuario pueda interactuar con la base de datos, así como los mecanismos de Autenticación para su uso.

### B. Implementación del prototipo

Para la implementación del prototipo, una vez analizado las diferentes plataformas de sistemas embebidos, se determina la mejor opción que cumpla las características de hardware y software para realizar la aplicación. A demás, se selecciona el formato de archivo para almacenar secuencia de imágenes. Posteriormente, Se elige la base de datos no relacional adecuando para trabajar con la unidad externa de gran capacidad.

Se construye los módulos de interfaz gráfica que permitan al usuario realizar consulta de archivos con los parámetros: fecha, hora, ubicación, ruta, y prioridad.

### C. Análisis de resultados

En el análisis de resultados se evaluará y documentará las distintas funcionalidades para el acceso de los usuarios creados, así como la gestión de la base de datos mediante filtros, se finalizará con conclusiones y recomendaciones.

Funcionalidades principales de la aplicación para gestión de archivos de tipo comprimido:

1. Para el ingreso de los usuarios se evaluará la autenticación con el mecanismo SCRAM 256 como mecanismo de seguridad en la base de datos de MongoDB.
2. Para la administración mediante la interfaz gráfica de dependerá del rol de usuario al momento de realizar la autenticación.
3. Se mostrará al usuario privilegiado (root), las siguientes características.

Se presenta la interacción del Usuario con la base de datos de administración.

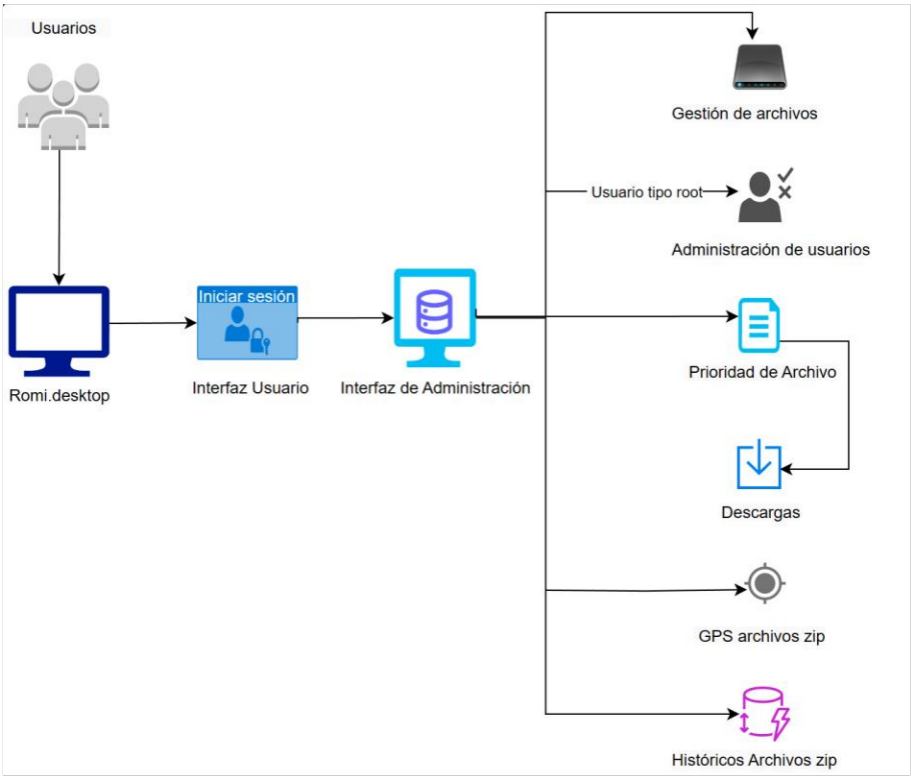



**Figura 1.3.1** Funcionalidades de la aplicación

Se puede observar en la Figura 1.3.1, en la aplicación constará de un ejecutable de tipo Desktop llamado Romi para la ejecución.

- **Fase de "Login" de usuario (Módulo 1)**

  Es esta fase el usuario podrá ingresar sus credenciales para verificar en la Autenticación de la base de datos de NoSQL mediante el mecanismo SCRAM 256, si existe un usuario podrá ingresar a la interfaz de administración, para el usuario privilegiado de tipo root se mostrará todas las opciones como se muestran en la figura 1.3.1, y si el usuario no privilegiado, la opción de Administración de usuario no aparecerá.

- **Fase de Interfaz de Administración (Módulo 2)**

  En esta fase de interfaz de administración, el usuario acceder a las colecciones(subelementos) creadas en la base de datos no relacional, utilizando filtros se podrá acceder a los archivos que se encuentran en la base de datos y gestionarlos, por hora, fecha, ubicación, y prioridad, existirá un parámetro en de ruta donde se podrá acceder, a la ruta real que se encuentra el archivo comprimido en la unidad externa.

- **Gestión de Archivos, unidad externa.**

  Todos los usuarios podrán acceder a la gestión de archivos, se podrá acceder a los archivos mediante filtros, ubicación, fecha, hora, y prioridad.

  Para el usuario es de tipo privilegiado root tendrá un campo en cada colección (archivo) en el cual podrá gestionar mediante 2 botones con las opciones de eliminar de la base de datos eliminar de la base de datos y eliminar el archivo de la unidad externa.

  No se realizará la fragmentación de la unidad externa de gran capacidad.

- **Administración de Usuarios**

  Esta opción solo podrá utilizar el usuario tipo root, en este campo se podrá crear usuarios y asignar roles, eliminarlos, vista de usuarios del sistema.

- **Prioridad de Archivos**

  Todos los usuarios podrán asignar prioridades a los archivos para posteriormente descárgalos.

- **GPS de Archivos de Imágenes**

  Los usuarios podrán visualizar la ubicación donde se recepto el archivo, mostrando en un mini mediante coordenadas decimales.

- **Histórico de Usuarios**

  El usuario de tipo root puede acceder al histórico para verificar los registros de ingreso al sistema.

## 1.4 MARCO TEÓRICO

### 1.4.1 DESCRIPCIÓN DE LOS SISTEMAS EMBEBIDOS

#### 1.4.1.1 Reseña Histórica

Los sistemas embebidos cada vez se van a adaptando a las nuevas tecnologías por su versatilidad de aplicaciones. En 1960, fue implementado por Charles Stark Draper el primer sistema embebido para el desarrollo del Sistema de Guía Apolo reduciendo el peso y tamaño de la computadora Guía, ayudando a los astronautas a obtener datos de vuelo en tiempo real [5].

En 1965, Autonetics, desarrollo EL D-17B, fue el primer sistema embebido producido en grandes cantidades, se utilizó por las fuerzas armadas de lo Estados Unidos en el sistema de guía de moles Minuteman para portal misiles balísticos intercontinentales [5].

Se lanzó en 1968 el primer sistema embebido para vehículos, posteriormente en 1971 se elaboró por la compañía Texas Instruments el primer microcontrolador este microcontrolador estuvo en el mercado por 4 años, el cual constaba de un procesador de 4 bits, y fue utilizado especialmente en calculadoras y dispositivos electrónicos pequeños, memoria solo de lectura (ROM) y memoria volátil (RAM), su precio tenía alrededor de 2$ al por mayor, Además, en 1987, se integra el primer sistema operativo , VxWorks en tiempo real, lanzado por Wind River, seguido por Windows Enbeddeb CE de Microsoft en 1996 [6].

Para finalizar la década de los 1990, fue lanzado el primer sistema Linux embebido.

A principio de la década de 2000, las soluciones integradas habían progresado de manera excepcional, dejando a las computadoras integradas obsoletas utilizadas en la vida cotidiana.

Los sistemas embebidos han recorrido un amplio camino siendo cada vez más robustos, disponiendo de funcionalidades muy avanzadas mediante su hardware y software. Estos miniordenadores toman un papel fundamental en la revolución del Internet de las Cosas (IoT) [7].

#### 1.4.1.2 Que es un Sistema Embebido

Sistema Embebido es un sistema integrado conformado por una combinación de hardware y software, están diseñados para realizar actividades específicas, y funcionan como parte de un sistema más grande, controlando las funciones dentro de un dispositivo multifuncional, se los denominan minicomputadoras que poseen un bajo costo y alto rendimiento con consumo de energía mínimos diseñados sobre sistema mecánico o eléctrico, pueden ser programados o tener una funcionalidad fija, estos

sistemas están constituidos por una interfaz para los sistemas de actividades fijas o pueden poseer interfaz de usuario (GUI) para actividades complejas [8].

### 1.4.1.3 Aplicaciones de Sistemas Embebidos

Los sistemas embebidos tienen aplicaciones en distintos sectores, Industriales, Telecomunicaciones, Salud, Automotriz, Comerciales, Militares y Aeroespaciales [6]

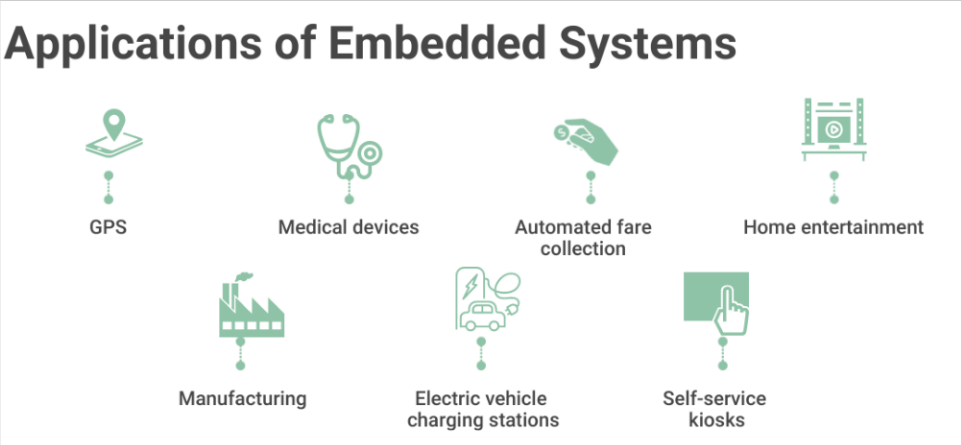



**Figura 1.4.1** Aplicaciones de Sistemas [9]

### 1.4.1.4 GPS

Los sistemas GPS en aplicaciones de sistemas embebidos se utilizan constantemente en vehículos y dispositivos móviles, su sincronización de datos de localización se realiza mediante satélites. Los receptores se encargan de obtener grandes cantidades de flujo de datos del GPS permitiendo el uso de posicionamiento global del vehículo o dispositivo integrado [9].

### 1.4.1.5 Dispositivos Médicos

Los dispositivos médicos electrónicos se han integrado al IoT con ayuda de los sistemas embebidos permitiendo crear aplicación de detección de enfermedades, escaneo de imágenes. La resonancia magnética (RM), escáneres de rayos X se han utilizado en tratamientos médicos por muchos años, pero con la adaptación de los sistemas embebidos los dispositivos médicos cada vez son más portables, facilitando el monitoreo de signos vitales, patrones de sueño mediante relojes inteligentes [10].

### 1.4.1.6 Colección de Tarifas Automatizada

Las soluciones automáticas de recolección de tarifas consisten en máquinas de tickets, tarjetas inteligentes, tarjetas de cintas magnéticas que permiten al usuario utilizar transporte público, realizar compras directamente en los supermercados, evitando la aglomeración de fila al momento realizar su comprar, solo se necesita poseer una tarjeta y escanear el código del producto y la compra se carga directamente a tu cuenta [9].

#### 1.4.1.7 Entretenimiento en casa

La tecnología del hogar inteligente implica introducir a varios dispositivos en la vida cotidiana, facilitando las actividades como, calefacción, seguridad, y entretenimiento. Los televisores inteligentes disponen de tarjetas de red para permitir la conexión a internet, así como un sistema operativo para disponer de aplicaciones de steaming por lo que desempeñan una función importante en la actualidad [11].

#### 1.4.1.8 Fabricación

Las industrias han adaptado sistemas integrados para la producción. Mediante robots se realizan tareas que requieren alta precisión o tareas de trabajos peligros, están equipados con actuadores, sensores, y software permitiendo aumentar la precisión, velocidad, y la producción operativa de la industria [12].

#### 1.4.1.9 Estaciones de Carga de Vehículos Eléctricos

Los sistemas embebidos están experimentado un cambio gigantesco en la industria automotriz. Los vehículos eléctricos cada vez son adquiridos por la población del mundo por su consumo de energía limpia, las estaciones de carga disponen de energía para recargas las baterías de autos eléctricos, y su infraestructura es muy parecida a las gasolineras convencionales y el tiempo de carga puede demorar de 25 a 90 minutos para su caga total. La Administración de batería están controlados por sistemas integrados que se encargan de supervisar la batería de litios de los vehículos eléctricos, verificando se efectúe correctamente la carga, descarga, y optimizando el consumo de batería [13].

### 1.4.2 ANALISIS DE PLATAFORMAS DE SISTEMAS EMBEBIDOS.

En esta sección se analizarán dos plataformas de sistemas embebidos consideradas en el transcurso de la formación profesional, para elegir una que se adapte a las dependencias para trabajar conjunto con una la base de datos no relacional. Estas plataformas se utilizaron en el laboratorio de Sistemas Embebidos.

#### 1.4.2.1 Arduino Uno R3

Es una plataforma de sistema embebido de código abierto que posee un software y hardware construida para facilita la interacción con dispositivos electrónico, permitiendo a los usuarios controlar de manera sencilla: sensores, módulos, actuadores, cámaras, utilizando su entorno de desarrollador integrado (IDE). Su placa posee un microcontrolador basado en el ATmega328P. Este microcontrolador se encarga de efectuar los requerimientos del usuario ejecutando las distintas rutinas mediante la programación en el IDE [14].


**Figura 1.4.2** Arduino Uno R3 versión DIP [15]

### 1.4.2.2 Características de Arduino Uno R3

Arduino Uno dispone de tableros de desarrollo, con entradas y salidas analógicas o digitales, permitiendo al usuario conectarse y comunicarse directamente a otros dispositivos, cuenta con 14 pines de input/output digital (6 son para salidas PWM) y 6 entradas analógicas, un oscilador de cristal de 16 MHz el cual se encarga de establecer la frecuencia del tiempo para garantizar la precisión del microcontrolador ejecutando las instrucciones del usuario, un conector de potencia con conversión CA a CC, una conexión USB que permite cargar el código fuente al microcontrolador para ejecutar las distintas rutinas programadas, también puede brindar energía a la placa sin necesidad de una fuente de alimentación externa [14].

La placa de Arduino uno dispone de una entrada de alimentación máxima de 20 voltios, y su corriente máxima de 40 mA por pin, sus entrada y salida, se pueden utilizar en el modo Alto/Bajo (High/ Low). Los pines analógicos de la placa pueden ser utilizados para leer los niveles de voltaje de sensores, actuadores, y otros dispositivos analógicos [14].

**1. Microcontrolador Atmega 328P**

El microcontrolador Atmega38P, es un chip principal de la placa de Arduino Uno R3 con capacidad de realizar operaciones en unidades de 8 bit a la vez, dispone de una tecnología AVR RICS de alto rendimiento y una memoria flash ISP de 32KB con capacidades de lectura y escritura, EEPROM de 1 KB, SRAM DE 2 KB. Las características físicas de Arduino se presentan en el Anexo I.

### 1.4.2.3 Raspberry PI

Es una plataforma de sistema embebido de código abierto y pose un hardware y software, se ha vuelto muy popular entre los aficionados de la tecnología de miniordenadores. Raspberry Pi es una computadora de bajo costo, tiene un tamaño semejante a una tarjeta de crédito, no tiene una interfaz de usuario incorporada para su funcionamiento, se debe conectar a un monitor o televisor mediante HDMI, Para interactuar con la placa y realizar acciones es necesario conectarse a la Raspberry, puede ser realizar mediante consola con tan solo activar los protocolos de comunicación desde la termina. Dispone de gran variedad de aplicaciones como: navegadores web, entornos de desarrollador como Visual Studio Code, QT Designer para el desarrollo de interfaz gráficas, estas aplicaciones no están cargas por defecto en la Raspberry Pi, Por lo tanto, es necesario instalar paquetes extras utilizando el comando APT [18].

Para la ejecutar la placa raspberry pi, se debe instalar un sistema operativo, como Raspbian (Debian Linux), Windows, Android, IoT Core, Ubuntu entre otros.

Raspberry Pi fue desarrollada por la Fundación Raspberry pi en Reino Unido, es una serie de potentes y pequeños computadores con un alto rendimiento, su lanzamiento fue en el ano de 2012, con el pasar del tiempo gracias a su optimo desempeño han lanzado varias versiones. Todas las versiones constan de un sistema Broadcom en un chip (Soc) y su CPU integrada compatible con ARM y una unidad de procesamiento de gráficos (GPU) en chip [19]

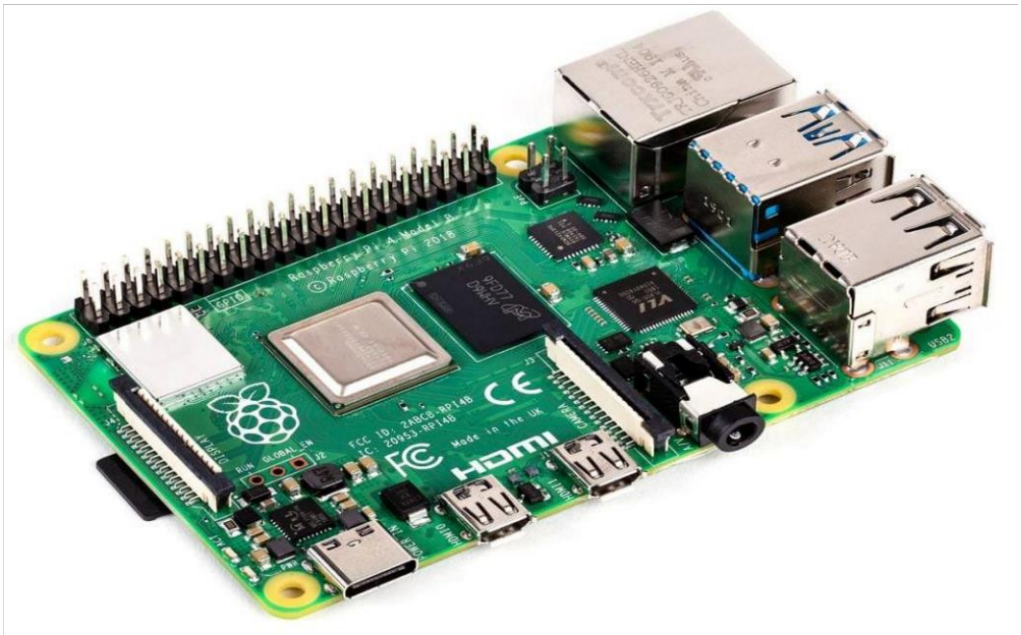


**Figura 1.4.3** Raspberry Pi 4 Modelo B [20]

### 1.4.2.4 Características de Raspberry Pi 4

Raspberry Pi 4 Modelo B dispone de un procesador de cuatro núcleos de 64 bit de alto rendimiento, compatible con pantallas duales con alta resolución (4K). Se puede conectar mediante el puerto HDMI mini su decodificación puede ser hasta de 4Kp60, y su memoria RAM hasta de 8 GB, LAN inalámbrica de doble banda de 2,4/5 GHz, un módulo bluetooth de generación 5.0, Interfaz Gigabit Ethernet, puertos USB version3.0, y capacidad PoE de manera independiente. Raspberry Pi 4 modelo B tiene un alto rendimiento por que utiliza un procesador Broadcom BCM2711, 4 núcleos Cortex-A74(ARM v8) 64-bit (SoC) con velocidad 1.5 GHz, semejante a los sistemas de PC x86 básico [20]. Para más detalles técnicos de las características físicas de Raspberry Pi se presentan en el Anexo II.

## 1.4.3 SISTEMAS OPERATIVOS EN RASPBERRY PI

Raspberry PI tiene algunas limitaciones al momento de elegir su sistema operativo (OS), debido a su arquitectura ARM. Para la elección del OS debe estar diseñado para esta arquitectura, es importante asegurarse de que el sistema sea compatible con el hardware de la placa, así como su capacidad de procesamiento.

Se presenta 3 sistemas operativos comunes para trabajar con Raspberry pi.

### 1.4.3.1 Sistema operativo Raspberry PI

Raspberry Pi OS está basado en Debian y es la distribución oficial de Raspberry Pi, dispone de una variedad de paquetes lo que permite utilizar herramientas de desarrollo, aplicaciones, software multimedia, servidores web, entre otros. Para la instalación de nuevos paquetes se debe gestionar el sistema de gestión de paquetes APT que permite la instalación, eliminación, y actualización de paquetes de software en sistemas basados en Debian, Ubuntu y Raspberry Pi Os [24] .

El sistema operativo Os dispone de un escritorio PÍXEL (Basado en LXDE, por lo tanto, puede utilizar como una computadora de escritorio proporcionando a los usuarios de Raspberry pi sea amigable y flexible. Pixel ofrece distintas aplicaciones precargadas como navegador Web, juegos, editores de texto (programación), también tiene opciones de personalización de barra de tareas, menú desplegable, fondos de pantalla adaptando a las necesidades del usuario [25]. Para descargar el sistema operativo puede seguir los pasos del Anexo III

### 1.4.3.2 Sistema operativo Ubuntu

Ubuntu es la distribución de Linux más utilizada en el mundo, tanto que, la fundación Raspberry PI ha optado en adaptar este sistema operativo para mejorar el ambiente de

trabajo con relación al sistema operativo Raspberry Pi OS. Dispone de algunas versiones que son menos estables, por lo cual, se debe considera el modelo de Raspberry PI para que funcione de una manera eficiente. Existe algunos cambios mínimos, como disponer del navegador Firefox instalado por defecto, se puede instalar parques extras mediante el comando APT [27].

Raspberry PI 4 modelo B, ahora tiene una versión de Ubuntu que contiene escritorio para mejor la interacción entre la placa y el usuario, dispone de variedad de paquetes para la instalación de aplicaciones, alguno de ellos viene precargados lo que hace que este sistema operativo sea una buena opción para trabajar con Raspberry Pi. Para descargar el sistema operativo puede seguir los pasos de Anexo IV

#### 1.4.3.3 Sistema operativo Windows

Windows 10 IoT Core (Internet de las Cosas), no es reconocido por la comunidad de Raspberry PI, su versión de Windows es optimizada para diseñar proyectos IoT, su desempeño es excelente en dispositivos ARM, por lo que se ajusta perfectamente para trabajar con la placa de Raspberry Pi en proyectos muy interesantes [27].Tiene una limitación para ejecutar programas, pero funciona muy bien con paquetes Linux y scripts de Python. Sin embargo, no es posible ejecutar los paquetes de software de escritorio de Windows. La gestión de Windows 10 IoT está disponible en la web, y es necesario disponer de una PC de Windows 10 para realizar su instalación [28].

En la actualización del Sistema Operativo Windows para Raspberry Pi 4, se puede ejecutar con la versión completa de Windows 10 en dispositivos ARM. Los requerimientos mínimos para la instalación se encuentran en el Anexo V

### 1.4.4 FORMATO DE ARCHIVOS COMPRIMIDOS Y IMÁGENES

Existe una variedad de formato de archivos comprimidos y de imágenes con las cuales se puede manejar su cabecera o metadatos. Los formatos de archivos pueden utilizarse dependiendo del sistema operativo y el tipo de arquitectura del dispositivo, para ello se analiza las necesidades de uso de archivos y compatibilidad con el entorno de desarrollador. Para acceder a su información existen algunas herramientas de trabajo, puede ser mediante la interfaz gráfica de usuario o con distintos lenguajes de programación, Python, Java, C, entre otros.

#### 1.4.4.1 Archivo tipo ZIP

Los archivos de tipo Zip realizan la compresión para reducir el peso del contenido real, comprimiendo uno a más archivos en un solo elemento. Por lo tanto, el archivo será más fácil de almacenar o exportar, se puede descomprimir el archivo ZIP (extraer su

contenido), para obtener los archivos sin pérdida de información, esta es una forma de optimizar él envió de grandes cantidades de datos. Los archivos ZIP son parecidos a una carpeta en su ordenador, la cual se puede agrupar en una sola carpeta distintos formatos de archivo, lo que hace el archivo ZIP es comprimir la carpeta contenedora o también un solo archivo. El archivo zip se puede identificar por su extensión .zip al final del nombre del archivo o por su cambio de icono, realiza una eliminación de datos redundantes a lo que se le conoce como (compresión sin pérdida de información).

- **Meta data de archivo ZIP**

En el archivo ZIP se representa como un parámetro de la meta data con el nombre de archivo local, describe la información de un archivo especifico como: nombre, método de compresión, comentario entre otros, utiliza 4 bytes para la firma específica para describir la diversa estructura del archivo. Se detalla los parámetros de cabecera de archivo en la siguiente figura [30].

| | |
|---|---|
| firma de cabecera de archivo | 4 bytes |
| versión necesaria | 2 bytes |
| indicador de bits universal | 2 bytes |
| método de compresión | 2 bytes (8=DEFLATE; 0=UNCOMPRESSED) |
| hora de última modificación del archivo | 2 bytes |
| fecha de última modificación del archivo | 2 bytes |
| crc-32 | 4 bytes |
| tamaño comprimido | 4 bytes |
| tamaño descomprimido | 4 bytes |
| longitud de nombre de archivo | 2 bytes |
| longitud de campo adicional | 2 bytes |
| nombre de archivo | variable |
| campo adicional | variable |

**Figura 1.4.4** Metadatos de archivo ZIP [30]

- **Manejo de Meta data con Python**

  Python es una herramienta para trabajar con metadatos tipo ZIP. El manejo de archivos se puede realizar importando algunos módulos en el entorno de desarrollador.

  Las clase y métodos de Python se muestran en la tabla 1.1

**Tabla 1.1** Clase en Python para generar archivos [31]

| **Clase ZipFile** | Es una biblioteca de Python que permite trabajar con archivo ZIP. |
|---|---|
| **Métodos** | **Descripción** |
| *zipfile.ZipFile(),* | Permite abrir el archivo, puede ser una ruta a un archivo de tipo cadena, o un objeto semejante a un archivo. |
| *ZipFile.write()* | Permite agregar o un archivo de texto a un ZIP. |
| *ZipInfo* | Permite crear un objeto para agregar metadatos. |
| *ZipFile.namelist()* | Permite obtener mediante una lista los nombres de los archivos que se encuentran en el ZIP. |
| *ZipFile.getinfo(nombre_archivo)* | Permite ver la meta data del archivo ZIP, como tamaño, fecha de acceso, etc. |
| *ZipFile.writestr()* | Permite añadir contenido a los metadatos |
| *ZipFile.write()* | Permite agregar archivos al archivo ZIP agregando junto a la metada opcional. |
| *ZipFile.comment* | Permite obtener el comentario que contiene el archivo ZIP. |
| *os.path.basename (),* | Permite obtener la ruta de archivo, y devuelve en nombre de archivo sin la ruta. |

### 1.4.4.2 Archivo Tipo TAR

Los archivos tipo TAR realizan la compresión de uno o varios archivos, son basados en Unix y trabaja por la línea de comandos, pero tiene la diversidad de ser compatibles con la mayoría sistemas operativos, ahora funciona muy bien en Windows, y es uno de los formatos de archivos más utilizados. La cabera de archivos TAR contiene metadatos como se muestra en la figura 1.4.5

| Desplazamiento de campo | Tamaño de campo (bytes) | Campo |
|---|---|---|
| 0 | 100 | Nombre de archivo |
| 100 | 8 | Modo de archivo |
| 108 | 8 | ID de usuario numérico del propietario |
| 116 | 8 | ID de usuario numérico del grupo |
| 124 | 12 | Tamaño del archivo en bytes (base octal) |
| 136 | 12 | Hora de la última modificación en formato de hora Unix numérico (octal) |
| 148 | 8 | Suma de comprobación para el registro de encabezado |
| 156 | 1 | Indicador de enlace (tipo de archivo) |
| 157 | 100 | Nombre del archivo vinculado |

**Figura 1.4.5** Metadatos de archivo TAR [32]

- **Manejo de Meta data con Python**

  Python es una herramienta versátil al momento de trabajar con metadatos de tipo TAR. El manejo de archivos se puede realizar algunos métodos para extraer información importe.

  Los métodos de Python se muestran en la tabla 1.2

  **Tabla 1.2** Métodos en Python para extraer metada de archivos TAR [33]

| **Clase TarFile** | Es una biblioteca de Python que permite trabajar con archivo TAR. |
|---|---|
| **Métodos** | **Descripción** |
| *tarfile.open ()* | Permite abrir el archivo, dependiendo de los argumentos del método. |
| *TarFile.close()* | Permite cerrar el archivo TAR. |
| *TarFile.extractall()* | Permite extraer los elementos del archivo en la ruta deseada. |
| *TarFile.extract()* | Permite extraer un archivo específico de archivo TAR en una ruta especifica. |
| *TarFile.getmember()* | Devuelve una lista de objetos de tipo "TarInfo" de un archivo específico TAR. |
| ***Clase TarInfo*** | Es una clase que presenta la información de metados de cada archivo TAR. |
| **Métodos** | **Descripción** |
| *TarInfo.fromtarfile()* | Permite crear un objeto "TarInfo" ya existente y una cabecera opcional. |
| *TarInfo.tobuf()* | Convierte la información de metadatos en un buffer de datos. De cada archivo TAR. |
| *TarInfo.pax_headers()* | Devuelve un diccionario con la información de los encabezados PAX asociados al archivo TAR. |
| *TarInfo.toformat()* | Realiza la conversión de metadatos a un formato deseado por el usuario. |
| *TarInfo.unlink(target)* | Permite eliminar un archivo en específico del archivo TAR. |

### 1.4.4.3 Archivo Tipo JPG

Los archivos JPG son utilizados para imágenes con el formato JPEG. En particular, contienen un mecanismo de compresión con pérdida para minimizar el peso del archivo, esto hace que la imagen pierda calidad, pero es imperceptible para el ojo humano. Se puede transmitir grandes volúmenes de datos con facilidad por vía Web y hace que muchos dispositivos electrónicos utilicen este formato debido a su aceptable cálida y peso en dispositivos con almacenamiento limitado, Por lo tanto, es uno de formato más utilizado para imágenes, fotos, gráficos. JPG es un formato compatible distintos sistemas operativos lo que permite visualizar los archivos sin necesidad de agregar extensiones para su ejecución [34].

Los archivos con formato JPG disponen de una metadata que proporcionar la información de la imagen, para acceder es necesario buscar la sección "Exif" que posee

parámetros como fecha modificación, hora, ubicación geográfica, tamaño de la imagen, entre otros. Los datos Exif puede variar dependiendo del dispositivo que captura la imagen.

| | |
|---|---|
| Dimensiones | 4000 x 2667 |
| Ancho | 4000 píxeles |
| Alto | 2667 píxeles |
| Resolución horizontal | 300 dpi |
| Resolución vertical | 300 dpi |
| Fabricante de la cámara | Canon Inc. |
| Modelo de la cámara | Canon EO S7000 |
| Tiempo de exposición | 2 segundos |

**Figura 1.4.6** Ejemplo de metadatos de archivo JPG [35]

- **Manejo de Meta data con Python**

  Python ayuda a extraer los metadatos de tipo JPG. Dispone de diferentes bibliotecas para extraer de información importe del archivo de imagen.

  Los de Python para acceder a los metadatos de archivos de tipo JPG, se muestran en la **Tabla 1.3.**

**Tabla 1.3** Métodos en Python para obtener metados de tipo JPG [36]

| **Biblioteca Pillow** | Es una biblioteca que está disponible en Python. Se tiene que instalar para acceder a sus métodos y trabajar con la extracción de metadatos de archivos con formato JPG. |
|---|---|
| **Métodos** | **Descripción** |
| *Image.open()* | Permite abrir una imagen desde una ruta o un archivo. El objeto Image representa a la Imagen procesada. |
| *Image.size()* | Permite obtener las dimensiones de la imagen (alto, ancho) en formato de tupla. |
| *Image.format* | Devuelve el formato del archivo de la imagen. |
| *Image.mode* | Obtiene el modo de color de la imagen en formato RGB o escala de gris. |
| *img_getexif()* | Permite obtener toda la información de la metada EXIF de la imagen: img, es la imagen que selección para la extracción de metada. |
| *ExifTags.TAGS.get(tag, tag)* | Permite el mapeo del identificador tag que corresponde en el diccionario de etiquetas EXIF. |
| *ExifTags.GPSInfo* | Permite obtener la información de GPS en la meta data del archivo. |

### 1.4.5 BASES DE DATOS NO RELACIONAL

#### 1.4.5.1 MongoDB

MongoDB es una base de datos no relacional (NoSQL) de código abierto. La implementación para crear base de datos es adaptable a las necesidades de usuario al momento de estructuración de documentos, facilitando la búsqueda de datos multivariable al momento de su almacenamiento como es el caso común de creación de filas y columnas en el estándar del base de datos (MySQL). La Bases de datos NoSQL tienen la facilidad de guardar documentos en colecciones como unidad básica. Su notación para su estructura de documentos puede ser la clave de cadena y se asigna a un valor que puede ser de distintos tipos (numéricos, cadenas, valores booleanos, matrices, valores vacíos u otro objeto). Se puede estructurar utilizando el lenguaje de programación Python en forma de diccionarios. El formato más utilizado para el intercambio de datos en la Web es de tipo BSON (Binary JSON), el cual codifica el formato JavaScript Object Notation (JSON), minimizando su velocidad de búsqueda, y espacio de una manera eficiente [37].

#### 1.4.5.2 Instalación de MongoDB en Raspberry PI

Los Raspberry PI es una plataforma con características espectacular con variedad de aplicación. Se puede instalar cómo servidores locales o en la Web, ahora los desarrolladores de MongoDB han optado por incluir un archivo pre compilador hasta las versiones 4.4 para que trabaje con el procesador de la Raspberry PI [38].

En la actualidad existen nuevas versiones de Raspberry PI, Por lo tanto, la extensión de la arquitectura cambia lo que hace que los repositorios antiguos de MongoDB estén obsoletos para trabajar con esta plataforma.

MongoDB se puede instalar en distintos sistemas operativo, pero es esencial conocer la arquitectura del dispositivo a instalar. MongoDB tiene un sitio web oficial que proporciona instrucciones dependiendo en el sistema operativo de instalación, así como claves públicas para acceder a sus repositorios que contiene los archivos de instalación. Sin embargo, existe limitación en Raspberry PI 4 que posee la arquitectura ARMv8.0. Dado que en el sitio oficial de MongoDB presenta la información de compatibilidad y requisito mínimos de instalación de la versión 5.0 los cuales son:

**ARM 64**

MongoBD requiere ARMv8.2-A o posterior microarquitectura arm64, A partir de la MongoDB 5.0, mongodb, mongos, y mongo Shell ya no admite plataformas que no cumplan con este requisito mínimo de microarquitectura arm64 [39]. Pese a estas

limitaciones se puede compilar Community Edition desde el código fuente en su versión 5.0.5 utilizando una compilación nativa [4].

### 1.4.5.3 Uso de MongoDB con Python

Python permite trabajar conjuntamente con bases de datos mediante el entorno de desarrollado integrado (IDE). Vale recalcar que se puede utilizar cualquier IDE para la construcción de la estructura de conexión con la NoSQL.

- **PyMongo**

MongoDB contiene un controlador oficial de Python llamado PyMongo. Para realizar la conexión a MongoDB es necesario instalar Pymongo utilizando pip, el cual es el sistema de gestión de paquete de Python. Posteriormente, es necesario importar en tu script el módulo de Pymongo.

Para instalar del módulo de pymongo se lo puedo conseguir ingresado en la shell de Raspberry Pi el siguiente comando; *sudo pip install pymongo.* A demás, es necesario importar en Módulo al script como; *import pymongo.* Para posteriormente crear una conexión a la base de datos no relacional.

### 1.4.5.4 CouchDB.

Apache CouchDB es una base de datos de datos no relacional (NoSQL) de código abierto, su forma de almacenamiento de datos en formatos de documento JSON, esto implica que utiliza modelos de datos que no tienen esquema facilitando el almacenamiento de información. CouchDB se puede ejecutar en Raspberry PI a grandes servidores. Existe versiones como PouchDB que puede utilizar navegadores web para el manteniendo de los datos del usuario de manera sincronizada y Couchbase Lite proporciona el almacenamiento de datos en la nube diseñada para dispositivos móviles e IoT [40].

CouchDB permite realizar réplicas de bases de datos en diferentes nodos, brindando estabilidad y alta disponibilidad, manteniendo copias idénticas de bases de datos en diferentes nodos y una alta tolerancia al fallo. La réplica en CouchDB es bidireccional, lo que significa que, al momento de cambiar la configuración de un nodo, todos los cambios replican a los nodos existentes [40].

#### 1.4.5.5 Instalación de CouchDB en Debian 11

Para realizar la instalación de Apache CouchDB es necesario habilitar Snaps en Raspberry PI e instalar CouchDB.

Los Snaps se pueden instalar desde la tienda de Snap STORE, es una tienda de aplicaciones muy conocida por millones de personas.

Para instalar de mejor manera se debe utilizar la última versión del sistema operativo Raspbian [41]. Los comandos para actualizar la raspberry pi:

- *Sudo apt update*
- *Sudo apt install snap*
- *Sudo reboot*

Una vez reiniciado la raspberry pi se procede a instalar CouchDB.

- *sudo snap install couchdb.*

Para instalar CouchDB es muy importante la versión que se va a instala, debía a que existen limitaciones en el procesador de la placa. Existen versiones compatibles. Algunas versiones pueden necesitar bibliotecas que no son compatible con Raspberry Pi o simplemente no se puede trabajar con la arquitectura. CouchDB utiliza entornos de red distribuida. Por lo tanto, los datos no residen en un solo servidor o nodo [41].

### 1.4.6 CONSTRUCCION DE INTERFAZ GRAFICA.

Existe diferentes aplicaciones para construir interfaces gráficas. Esto proporciona al usuario una forma interactiva con el dispositivo de una manera más amigable, la creación de botón, zonas de trabajo, barra, facilitan el control del software.

La construcción de GUI es una buena opción para cumplir los requerimientos funcionales del usuario. Existe diferentes herramientas para la construcción de interfaces graficas mediante aplicaciones, tales como:

#### 1.4.6.1 QT Designer

Qt Designer es una herramienta para el diseño y construcción de interfaces graficas de usuario (GUI). Puede construir widgets personalizados de manera que se pueda reutilizar el código en diferentes proyectos. De esta manera el usuario puede construir interfaces más simples para el uso. Tiene la facilidad de trabajar con widgets con tan solo arrastrarlos a la zona de trabajo de QT y posteriormente programarlos mediante el lenguaje de programación Python. A demás, se puede cambiar las propiedades establecida por Qt Designer de forma dinámica [42].

### 1.4.6.2 Qt Designer con Python

QT Designer se caracteriza por disponer gran variedad de herramienta para el diseño de GUI. Python se puede integrar perfectamente para el manejo de widgets. Primeramente, en necesario diseña la plantilla según las necesidades del usuario, Qt Designer es amigable en el proceso de construcción, tal que, el usuario puede arrastrar los elementos que conformaran la plantilla y con ayuda de Pyqt5 o PySide se implementa su funcionamiento. [43]

#### 1. PYQT5

PyQt5 es una biblioteca de Python que permite enlazar a la aplicación de QT Designer. A demás, puede crear nuevos Widgets en la plantilla creada en QT Designer. Con tan solo crear un nuevo script, posteriormente se carga el archivo Ui proporcionado por el programa de QT Designer. Listo, se puede interactuar con los elementos creados en QT desde Python. A demás, se puede crear nuevos elementos en la plantilla en momento de ejecución del programa. Para utilizar Pyqt5 se debe importar en el script las distintas librerías de uso [44].

#### 2. PYSIDE

PySide es el enlace oficial entre Qt mediante Python. A demás, Qt Designer tiene una opción que utiliza para guardar el código en Python. Por lo tanto, se puede visualizar el código en una vista previa de Qt Designer o directamente guardar en archivo con una extensión de tipo .py, A demás, puede interactuar con los widgets una vez generado el archivo código, se puede agregar el funcionamiento con los elementos de la biblioteca de PySide [45].

### 1.4.7 Entorno de desarrollador integrado

Para utilizar las librerías de Python en necesario trabajar con entorno de desarrollador integrado (IDE). Existe variedad de IDE que se adapta fácilmente por ser multiplataforma.

### 1.4.7.1 Visual Studio Code

Visual Studio Code es un editor de código fuente gratuito, ligero. Es compatible con variedad de sistema operativos, tiene dispone de soporte para Node.js, JavaScript, Type Script. A demás, dispone de una gran variedad de extensiones para trabajar con lenguajes de programación como: Python, C++, C, entre otros. Es multiplataforma por lo que los usuarios optan por trabajar con Visual Studio Code en el desarrollo de sus proyectos [46].

Visual Studio Code se puede instalar en Raspberry Pi, utilizando el gestor de paquetes ***apt*** con el siguiente comando; *sudo apt install code*, se instalará la versión disponible en el repositorio de Raspbian.

### 1.4.8 INGENIERIA DE REQUERIMIENTOS.

Es una rama de la Ingenia de Software que permite estableces objetivos, funcionalidades y restricciones en la construcción del prototipo, analizando los requerimientos, necesidades presentadas por el cliente. A demás, garantiza la creación efectiva de un software con las distintas funcionalidades para resolver las necesidades del usuario de forma satisfactoria [47]

#### 1.4.8.1 Requerimientos de Obtención

Los requerimientos de obtención son las necesidades y dominios de la parte interesadas en la construcción del producto de software. La información recolectada esta proporcionada por manuales comerciales, estándares de utilización, entre otras. Para la recopilación de la información de requisitos que debe cumplir, es necesario entender el problema de la construcción del software que prevé resolver [47]

Existen algunas técnicas de recolección de información para obtener los parámetros de construcción, entre ellas [48]:

1. **Encuentras:** Se realiza cuestionarios para difundir a las personas interesadas en construcción del proyecto, accediendo a la información sobre las necesidades y expectativas.
2. **Entrevistas:** Se accede a conversaciones individuales con las partes interesadas.
3. **Grupos Focales:** Se reúnen un pequeño grupo de personas para resolver las dudas de las expectativas y necesidades para el sistema de software.
4. **Observación:** Se observa las partes interesadas y el medio de trabajo para recolectar información de las necesidades y expectativa.
5. **Creación de Prototipo:** Esta técnica se encarga de crear un prototipo de trabajo del sistema software, y se puede utilizarse para la recolección de comentario de las partes interesadas.

#### 1.4.8.2 Especificación de Requerimiento

La especificación de requisitos es el proceso de documentar los requisitos analizados en los requisitos de obtención con el fin de producir modelos formales de software más eficientes. El objetivo para crear un documento debe ser comprensible tanto para el equipo de desarrolladoras como para las partes de interés.

Existen varios tipos de requerimientos que se especifica comúnmente, tales como [48]:

1. **Requisitos Funcionales:** Describe las actividades que debe realizar el sistema software, enfocado en funcionalidades como la validación de entrada, almacenamiento de la información y la interfaz de usuario.
2. **Requisitos No Funcionales:** Describe la eficiencia de las actividades del sistema de software, su rendimiento, fiabilidad, facultades de uso, seguridad de uso.
3. **Restricciones:** Define las limitaciones o restricciones en el momento de elaborar el sistema de software.
4. **Criterio de Aceptación:** Define las condiciones que debe cumplir para dar por concluido el desarrollo del sistema de software y listo para su difusión.

# 2 METODOLOGÍA

Para cumplir con los requerimientos de la aplicación del prototipo de software en la plataforma de sistema embebido, se enfocará en la administración de archivos comprimidos. En el presente capitulo se dispondrá de varios apartados, en el aparatado 2.1.1, se implementará el proceso Análisis de Plataforma de Sistema Embebido. En la sección 2.1.2. Se implementa el Análisis de Selección de Sistema Operativo, en el aparado 2.1.3. Se realizará el Análisis de Formatos de Archivo para la construcción de los filtros, en el apartado 2.2. Se describe en base a la Ingeniería de Requerimientos de Software.

## 2.1 Análisis Requerimientos

### 2.1.1 Análisis De Plataforma De Sistema Embebido

En base las características presentadas en el apartado 1.4.2. Se procede a realizar una tabla comparativa con las características de software y hardware de las plataformas de sistema embebido como se muestra en la tabla 2.1.

En parte a la asignación mediante un visto $(\surd)$ representa que la plataforma de sistema embebido contiene esa característica, y con $(\times)$ no dispone de la característica.

**Tabla 2.1** Características de Hardware y Software de Arduino Uno y Raspberry Pi 4

| Descripción | Característica | Arduino Uno | Raspberry Pi 4 |
|---|---|---|---|
| Hardware de Código Abierto | Hardware | √ | × |
| Redimiendo de Procesador | Procesador | × | √ |
| Memoria de Procesamiento | RAM | × | √ |
| Pines en la Placa | GPIO | × | √ |
| Sistema Operativo | OS | × | √ |
| Arquitectura de CPU | ARM | × | √ |
| Valor de Adquisición | Costo | √ | × |
| Interfaz Serial | UART | √ | √ |
| Conexión a Internet Independiente | WLAN | × | √ |
| Conexión por cable a Internet | Ethernet | × | √ |
| Puertos USB 3.0 | USB | × | √ |

En base la tabla 2.1. Se elige la Plataforma de Sistema Embebido Raspberry Pi. Tanto que, dispone de características como el procesador basado en ARM con el cual se puede ejecutar Aplicaciones, Servidores Web, Instalar Sistemas Operativos, Trabajar con bases de dato no relacionales como es el caso de este componente de TIC.

### 2.1.2 Análisis De Selección De Sistema Operativo

En base las características presentadas en el apartado 1.4.3. Se procede a realizar una tabla comparativa entre los distintos sistemas operativos del sistema embebido. A demás, se presenta su forma de instalación, compatibilidad con la plataforma de sistema embebido como se muestra en la tabla 2.2.

**Tabla 2.2** Características de Sistemas Operativos Para Raspberry Pi.

| Característica | Raspberry Pi OS | Ubuntu (OS) | Windows IoT (OS) | Windows 10 (OS) |
|---|---|---|---|---|
| Arquitectura del Sistema Operativo | ARM | ARM | ARM | ARM, x86, x64 |
| Interfaz de Usuario | Escritorio(LXDE) | Depende de la versión de Ubuntu | No por defecto | No por defecto |
| Facilidad de instalación | Fácil de instalar, recomendando para Raspberry Pi | Moderadamente Fácil, proceso común de instalación en Ubuntu | Fácil de configuración | Fácil de configuración |
| Desarrollo de Software | Amplio soporte para lenguajes de programación | Amplio soporte para lenguajes de programación | Enfocado en desarrollo de IoT con herramientas específicas | Enfocado en desarrollo de IoT con herramientas específicas |
| Compatibilidad de Hardware | Recomendado para Raspberry PI | Amplia Compatibilidad | Compatible Específico para IoT | Compatible Específico para IoT |
| Comunidad y Soporte | Actualizaciones Regulares | Actualizaciones Regulares | Actualización específica para IoT | Actualización específica para IoT |
| Estabilidad de Software | Muy estable para trabajar con Raspberry Pi. | Depende de la versión de Ubuntu instalada | Estable para manejos de proyectos IoT | Estable para manejos de proyectos IoT |

Una vez analizadas las características de la tabla 2.2. Se elige el Sistema Operativo Raspberry PI OS, por su versatilidad y estabilidad. A demás, dispone de un interfaz de usuario de tipo LXDE que permite al usuario que sea más amigable la interacción al momento de ejecutar aplicaciones, instalar paquetes, entornos de desarrollador, entre otros.

### 2.1.3 Análisis De Formatos De Archivo

En base las características presentadas en el apartado 1.4.4. Se procede a realizar una tabla comparativa con las características de los formatos de archivos como se muestra en la tabla 2.3.

**Tabla 2.3** Características de formatos de archivos.

| Característica | ZIP | TAR | JPG |
|---|---|---|---|
| Tipo de Archivo | Comprimido | Archivo de cinta (Tape Archive) | Imagen |
| Compresión | Sí | No | No |
| Acceso a Meta data/Encabezado | Facilidad de acceder a la meta data y compresión de archivo. | Relativamente fácil para el acceso a la metada y compresión de archivos. | Fácil de acceder a la metada mediante Exif. |
| Función en Python para Acceder a Meta data | Biblioteca “zipfile” Zipfile.ZipFile() | Biblioteca *“tarfile” Tarfile.open()* | Biblioteca PI “Pillow” img=Image.open() img_getexif() |

Una vez analizados las características de la tabla 2.3. Se elige el formato de archivo de tipo ZIP, esto permite almacenar secuencia de imágenes y agregar a su cabecera la metada deseada.

### 2.1.4 Análisis de base de datos no relacional.

En base las características presentadas en el apartado 1.4.5. Se procede a realizar una tabla comparativa con las características de base de datos (NoSQL) como se muestra en la tabla 2.4.

**Tabla 2.4** Características de Bases de Datos NoSQL.

| Característica | MongoDB | CouchDB |
|---|---|---|
| Arquitectura | Orientada a documentos (NoSQL) | Orientada a documentos (NoSQL) |
| Instalación en Raspberry Pi | Depende de la versión de Raspberry PI y el OS | Depende de la versión de Raspberry PI y el OS |
| Conexión con Python | Conexión a MongoDB mediante pymongo | Conexión a CouchDB mediante couchdb |
| Consultas y Operaciones | Consultas avanzadas | Consultas basadas en MapReduce |
| Casos comunes de Uso | Se opta cuando la estructura de datos no está definida, Grandes volúmenes de datos | Cuando la recolección de datos tiende a fallos críticos. |
| Estructura de Datos | Formato BSON | Formato JSON |
| Métodos para obtención de datos | Tiene variedad de métodos definidos para las búsquedas de datos. | Funciones MapReduce |

Una vez analizados las características de la tabla 2.4. Se elige a base NoSQL MongoDB por su caso común de uso, en la forma de estructurar los datos mediante el formato BSON, métodos para realizar búsqueda que se adaptan de mejor manera al componente de TIC.

### 2.1.5 Formato Del Nombre Del Archivo Comprimido

En esta sección se describirá la estructura del formato del archivo comprimido. El formato ZIP es el seleccionado, al cual se agregará en la cabecera los parámetros de Fecha, Hora, Ubicación. Los cuales permite identificar el archivo con facilidad. Para ello, se crea las funciones de Python que permitan la simulación de recepción de imágenes mediante una cámara y datos de un módulo GPS.

Para las imágenes y datos GPS se considera: Imágenes receptadas en formato JPG, que han sido almacenadas en el sistema de archivos de la Raspberry Pi, se puede acceder mediante la ruta de ubicación. Los datos GPS, pueden ser entregado por el módulo GPS UBLOX NEO-6M [49], sin embargo, la información de hora y fecha son tomados del sistema Raspberry Pi. Para el parámetro ubicación en coordenadas decimales (latitud, longitud) se simula de una lista de ubicaciones en Python, en razón de que esta característica es parte de otro componente de este proyecto. Detalles en el Anexo VI

Un ejemplo de la sintaxis y del formato correspondiente se muestran seguidamente.

**Fecha**: parámetro fecha **Hora:** parámetro hora **Ubicación:** parámetro ubicación.zip

Fecha: 2023-11-30 Hora: 00:43:24 Ubicación: 0.0427, -78.1456.zip

### 2.1.6 Estructura De Filtros

Una vez seleccionada la base de datos de MongoDB se estructura los datos con el formato BSON que permite agregar colecciones y dentro de las colecciones documentos. Para la estructura de los filtros se crea mediante el formato BSON [3].

**Tabla 2.5** Estructura De Filtros

| **SUBELEMENTO (COLECCIONES) Formato BSON (documentos)** | |
|---|---|
| **Clave** | **Valor** |
| Fecha | Fecha de Raspberry Pi |
| Hora | Hora de Raspberry Pi |
| Ubicación | Ubicación Lista creada |
| Ruta | Ruta de la unidad externa |
| Prioridad | Prioridad (0 por defecto) |

Se agregan los parámetros de búsqueda en la base de datos de MongoDB. Esto permite conseguir una referencia de los parámetros del archivo zip. Vale recalcar que las colecciones o como lo llamaremos ahora subelementos puede contener varios documentos.

Ahora para cada archivo zip está representado como un documento en MongoDB.

### 2.1.7 BUSQUEDA DE ARCHIVOS (FILTROS)

La búsqueda de archivos se realiza mediante una tabla widget creada en Qt Designer. En la cual el usuario podrá elegir el parámetro de búsqueda, para posteriormente ingresar el valor a buscar.

Las búsquedas se realizar obteniendo las referencias de las colecciones que se encuentran creadas y se acceder con métodos de recolección que proporciona MongoDB [50].

## 2.2 Ingeniería de Requerimientos

Mediante las etapas de la Ingeniera de Requerimientos se analizan previamente los requerimientos para la construcción de la aplicación de sistema embebido. Las etapas consideradas son: Análisis de requerimientos, Diseño, Codificación, Pruebas, Despliegue, como se puede observar en la Figura 2.2.1.

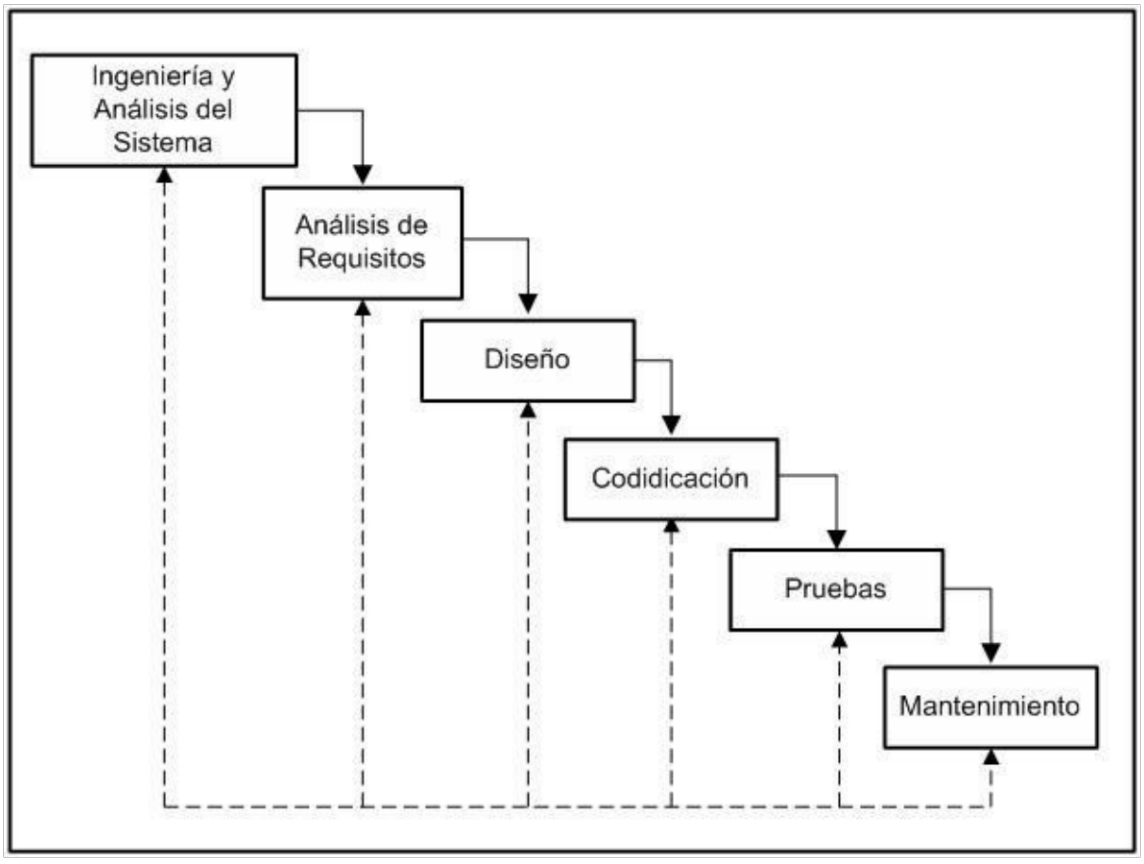



**Figura 2.2.1** Fase de desarrollo del Software [51]

- **Análisis de Requerimientos:** En esta fase se explorar los requerimientos funcionales y no funcionales para el desarrollo de la aplicación.

- **Diseño:** Se implementará los diagramas de clases y actividades, para la construcción de la aplicación. Para la implementación de la plantilla de los módulos 1 y 2 de Interfaz Gráfica de Usuario se utiliza Qt Designer.
- **Codificación:** Para la funcionalidad de los elementos de la interfaz, se utiliza la librería Pyqt5, así como para realizar filtros y funcionamiento de widgets de los módulos 1 y 2.
- **Pruebas:** En la evaluación de pruebas se considera usuarios no entrenados. Los dispondrán de acceso a los distintos módulos. Se realizarán varias pruebas de búsqueda de Archivos mediante filtros, Descarga de archivos, Asignación de Prioridad a las subelementos, Gestión de Usuarios, Gestión de Base de datos, Manejo de interfaces WLAN.
- **Despliegue:** La aplicación considera 2 módulos, El primer módulo está construido por una interfaz gráfica de usuario que permite la autenticación en la base de datos. En el segundo módulo permite la gestión de archivo mediante filtro. Se puede proyectar a la creación de un nuevo módulo 3 para la fragmentación de la unidad externa de gran capacidad.

Para obtener los requerimientos funcionales y no funcionales se realiza el análisis de los objetivos del proyecto. Del flujo de datos del componente de captura de imágenes y posición GPS. Se describe los requerimientos mediante (Snow Card) [52]. A demás, se sigue las etapas definidas en la Ingeniería de requerimientos para la construcción de la aplicación.

#### 2.2.1.1 REQUERIMIENTOS FUNCIONALES

Los requerimientos funcionales se establecen mediante la consulta a potenciales usuarios sobre los objetivos del proyecto y el aspecto que debería presentar la aplicación los cuales se puede visualizar en la tabla 2.6. Seguidamente describe los requerimientos funcionales mediante tarjetas (Snow Card) [52]. Se considera los módulos de interfaz gráfica para el análisis de requerimientos. En Módulo 01: Se establece como interfaz gráfica de usuario permitiendo la verificación de existencia de usuarios en el sistema de base de datos; en Módulo 02: Se establece como interfaz gráfica de usuario permitiendo la administración de archivos y usuarios en el sistema de base de datos. Los detalles del análisis y Snow Card se presentan en el Anexo VII

**Tabla 2.6** Requerimientos Funcionales

| REQUERIMIENTOS | | | | |
|---|---|---|---|---|
| **Id** | **Tipo** | **Modulo** | **Sección** | **Descripcion** |
| F01/M1 | Funcional | BDD | Instalación de Base de Datos | Instalar la base de datos en el sistema de Raspberry Pi. |
| F02/M1 | Funcional | Registro | Creación de usuarios | Creación de usuario privilegiado. |
| F03/M1 | Funcional | Login | Permite al usuario el ingreso a la base de datos | Ingreso de credenciales para inicio de sesión. |
| F04/M2 | Funcional | Administración de Base de datos | Barra Lateral Izquierda | Implementación de una barra lateral que permita alojar los nombres de bases de datos y subelementos. |
| F05/M2 | Funcional | Administración de Base de datos | Contenedor de Botones | Se crea 4 botones en el contenedor para realizar Descargar, Histórico, GPS, Crear base de datos |
| F06/M2 | Funcional | Administración de Base de datos | Zona de Filtros. | Se creará una zona de filtrado que permita contener una tabla para realizar filtros de búsqueda con parámetros: Fecha, Hora, Ubicación, y Prioridad. |
| F07/M2 | Funcional | Administración de Base de datos | Barra Superior Central | Se creará una barra superior central que permita contener 5 botones para el manejo de la red inalámbrica, gestión de usuario, eliminar usuario, agregar usuarios. |
| F08/M2 | Funcional | Administración de Base de datos | Barra Superior Derecha | Se creará una barra superior derecha que permita contener un diplay y 3 botos para el manejo de ventana de la interfaz grafica |

### 2.2.1.2 REQUERIMIENTOS NO FUNCIONALES

Se describe los requerimientos no funcionales del sistema de software en la tabla 2.7. Requerimientos no Funcionales de los Snow Card se encuentran detallados en el Anexo VIII

**Tabla 2.7** Requerimientos No Funcionales

| REQUERIMIENTOS | | | | |
|---|---|---|---|---|
| **Id** | **Tipo** | **Modulo** | **Sección** | **Descripcion** |
| FN01/M1 | No Funcional | Login | Ingreso de usuario | Autenticación de usuario |
| FN02/M1 | No Funcional | Login | Control de ingreso | Interfaz de usuario |
| FN03/M2 | No Funcional | BDD | Ingreso de usuario | Conexión con Base de Datos |

## 2.2.2 DISEÑO

Para el diseño de la aplicación se realizan: el diagrama de clase, el diagrama de actividades, el diagrama de conexión, el diagrama de funcionamiento, los cuales se muestras seguidamente.

### 2.2.2.1 Diagramas de Clases

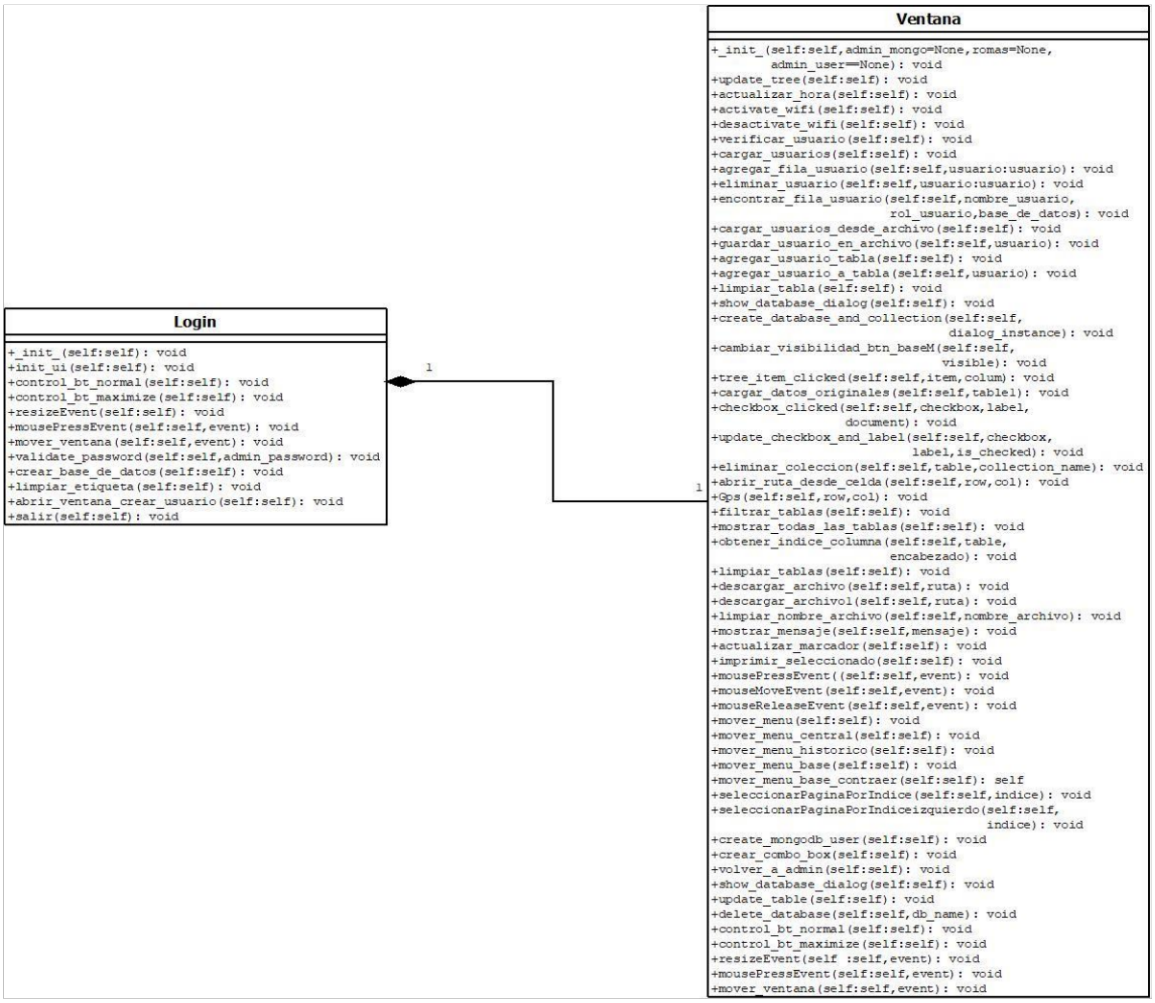



**Figura 2.2.2** Diagrama de Clases de la aplicación

### 2.2.2.2 Diagrama de Actividades

Se presenta el comportamiento de la interacción del usuario con el sistema como se muestra en la figura 2.2.3.

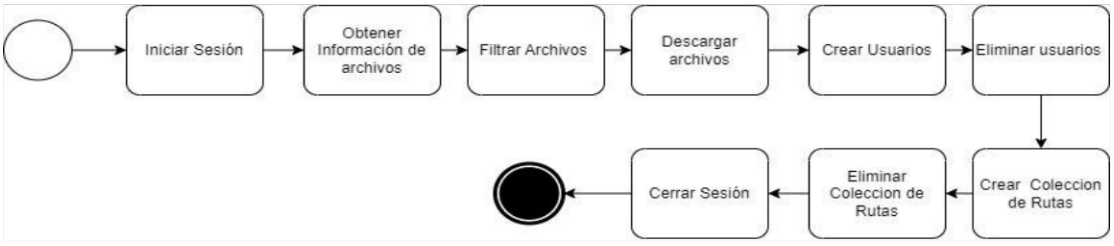



**Figura 2.2.3** Diagrama de Actividades de la aplicación.

### 2.2.2.3 Diagrama de Conexión

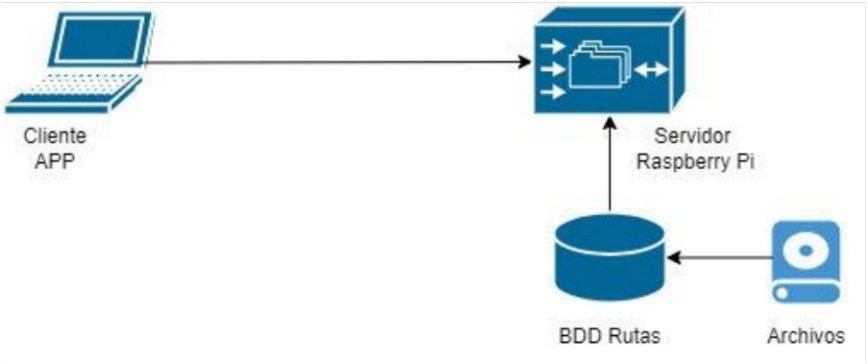



**Figura 2.2.4** Componentes de Conexión del Software

Como se puede visualizar en la figura 2.2.4 la conexión de la aplicación como cliente se encuentra la Raspberry Pi; el cliente se encarga de gestionar mediante una base de datos no relacional archivos comprimidos que contienen una secuencia de imágenes las cuales se encuentran alojadas en una unidad externa(USB). El diagrama de conexión de la construcción en físico se encuentra en el Anexo IX.

### 2.2.2.4 Diagrama de Funcionamiento

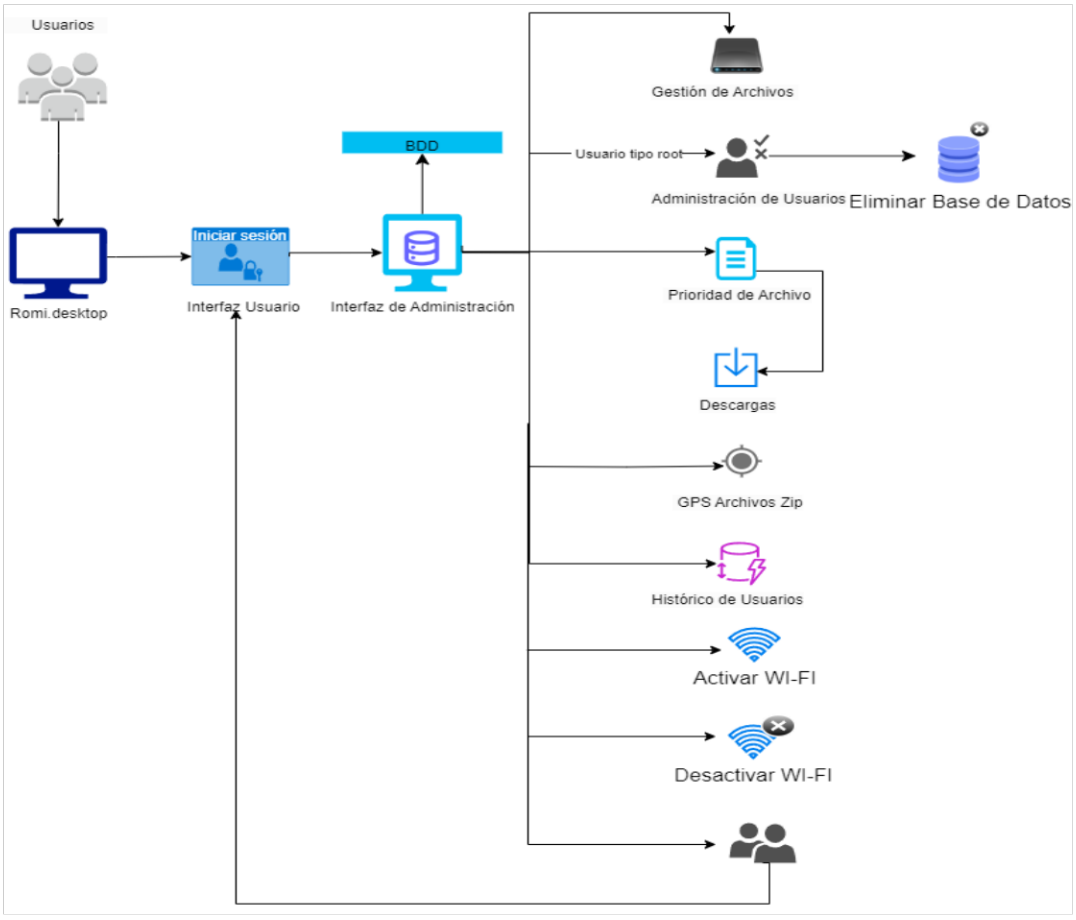



**Figura 2.2.5** Componentes de conexión del Software.

### 2.2.2.5 Diseño del Módulo 1

La aplicación estará constituida por un módulo 1 llamado “Login”, Por lo cual, se presentará la interfaz de ingresos a la Base de Datos, A demás permite la funcionalidad de conectarse con la base de datos no relacional que trabaja de forma local (Servidor Local). Para la autenticación de usuario se ingresa los campos de nombre del usuario, contraseña, nombre de base de datos en la cual se encuentra agrego el usuario. Para la verificación de usuario en el sistema se dispone de un llamado Iniciar Sesión. Enseguida, en una línea de texto que muestra la validación de usuario. Una vez verificado el ingreso, podrá acceder al Módulo 2 gesto de base de datos o cerrar sesión. Para la construcción del Módulo 1. Se utiliza la herramienta de diseño Qt Designer con la versión 5.15.2 disponible para Raspberry Pi 4. La cual Interactuar directamente con los Widgets de forma amigable. Solo es necesario arrastrar y soltar los Widgets en la zona de trabajo. Una vez realizado el diseño del Módulo 1, se guardará el archivo con el nombre Principal que contiene una extensión .ui.

Para la programación de los widgets y creación de nuevos elementos. Se utiliza Pyqt5, el cual permite enlazar la interfaz construida por el usuario en QT Designer utilizando el lenguaje de programación Python.Para la creación de los widgets en Qt Designer se crea un archivo .ui llamado Principal.ui, la platilla se puede obtener en el Anexo X. A continuación, se describe la construcción de Módulo 1.


**Figura 2.2.6** Interfaz ingreso de usuario (Módulo 1).

En la Figura 2.2.6 se puede observar en la interfaz del Módulo 1 correspondiente al inicio de sesión.

### 2.2.2.6 Diseño del Módulo 2

Si puede visualizar en la figura 2.2.7. El módulo de gestor de base de datos en el entorno de trabajo de QT Designer, permitiendo observar las zonas para las actividades presentadas en la sección 2.2.1 Análisis de Requerimientos. Existen más elementos creados en la interfaz mediante Pyqt5 para cumplir los requerimientos propuestos.


**Figura 2.2.7** Diseño de Interface Principal (Módulo 2)

## 2.2.3 IMPLEMENTACIÓN EN PYTHON

En este apartado se realiza la codificación del sistema de software, En el cual se utiliza el entorno de desarrollado Integrado VS Code [46]. Fue instalado sobre Raspberry Pi mediante el administrador de paquetes APT [18]. La versión disponible en el repositorio de Debian 11 es 1.85.2, con soporte Node.js: 18.15.0, OS: Linux arm64 6.1.21-v8+. Se realizan método en los distintos módulos. Para el módulo uno se tiene el archivo de Python con el nombre de principal.py. y un archivo tipo .ui llamado Principal.ui.

### 2.2.3.1 Implementación del Módulo 1

#### 1. Inicio de sesión

Para le módulo no crea métodos para inicializar la ventana, como se visualiza en la Código 2.1.

```python
import sys
from PyQt5.QtWidgets import QApplication, QMainWindow, QWidget, QVBoxLayout, QPushButton, QLineEdit, QLabel,QMessageBox
import pymongo
from pymongo.errors import ServerSelectionTimeoutError,OperationFailure,InvalidURI
from AdminMongo import AdminMongoDB
from romas_solo_interfaz import Ventana
import hashlib ## otras librerias
import os
from PyQt5.QtWidgets import QMainWindow, QApplication,QLineEdit
from PyQt5.QtGui import QGuiApplication,QIcon
from PyQt5.QtCore import QTimer
from PyQt5 import QtCore, QtWidgets
from PyQt5.uic import loadUi
import icons_rc
from PyQt5.QtCore import Qt

class Login(QMainWindow):
    def __init__(self):
        super(Login, self).__init__()
        loadUi(os.path.join(os.path.dirname(__file__), 'Principal.ui'), self) # Se carga el diseno de Qt designer
        # Método cuando para inciar
        self.init_ui()
        self.admin_mongo=None
        self.romas=None
```

**Código 2.1** Iniciación de ventana de Módulo en VS Code.

## 2. Conexión de Ranuras

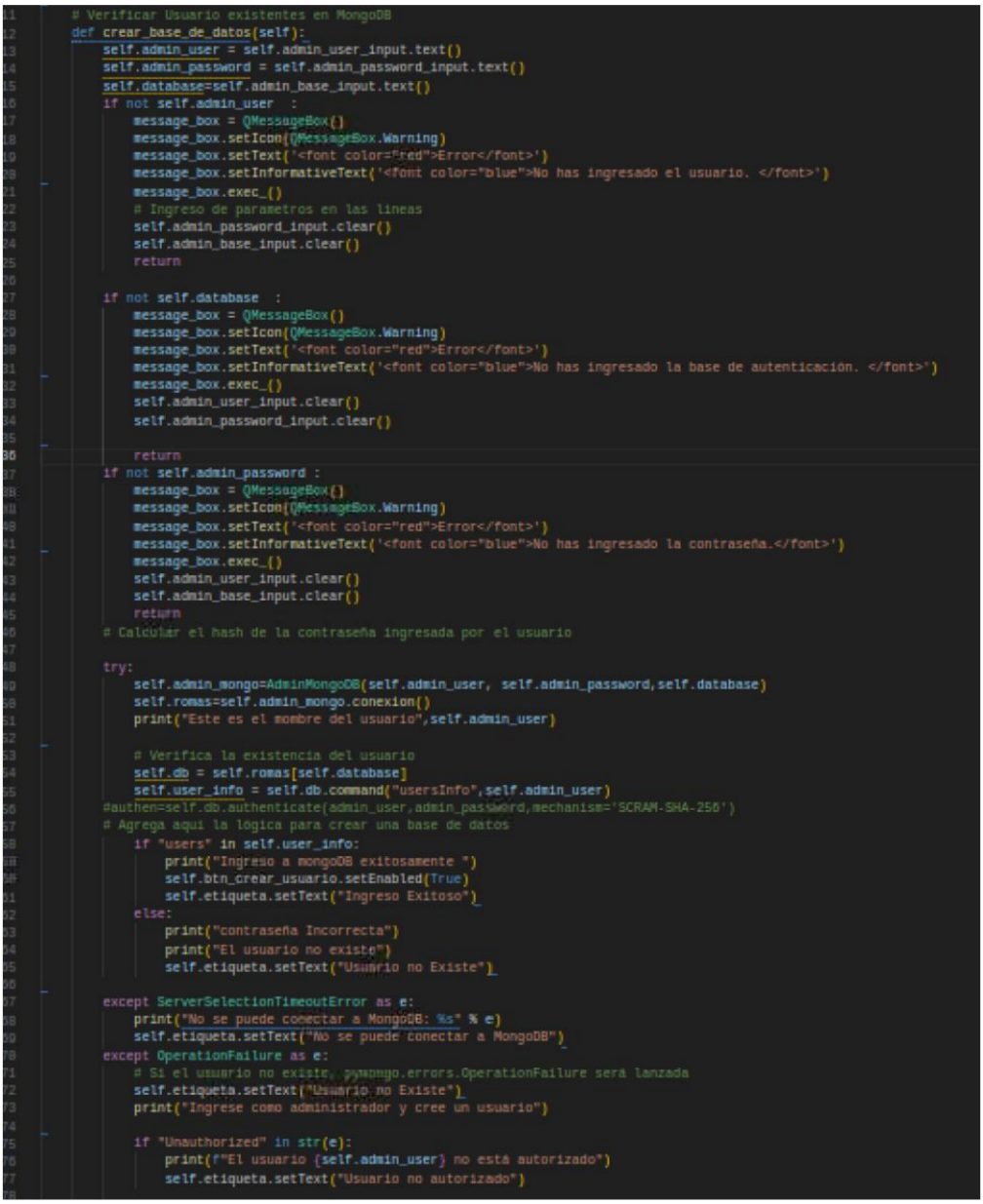
Se crea un método para inicializar las señales y ranuras de los elementos que contiene el módulo 1. Código 2.2 describe el proceso.

```python
def init_ui(self):
    self.bt_normal.hide()
    self.click_posicion = None
    self.click_posicion = None
    self.bt_minimize.clicked.connect(lambda :self.showMinimized())
    self.bt_normal.clicked.connect(self.control_bt_normal)
    self.bt_maximize.clicked.connect(self.control_bt_maximize)
    self.bt_close.clicked.connect(lambda: self.close())
    # Eliminar barra de titulo y opacidad
    self.setWindowFlag(QtCore.Qt.FramelessWindowHint)
    self.setWindowOpacity(1)
    self.setWindowFlags(QtCore.Qt.FramelessWindowHint)
    self.setAttribute(QtCore.Qt.WA_TranslucentBackground)
    # SizeGrip5
    self.gripSize = 10
    self.grip = QtWidgets.QSizeGrip(self)
    self.grip.resize(self.gripSize, self.gripSize)

    # mover ventana
    self.frame_superior.mouseMoveEvent = self.mover_ventana

    icon_user = QIcon(":/icons/icons/user.svg")
    icon_lock = QIcon(":/icons/icons/key.svg")
    icon_base = QIcon(":/icons/icons/database.svg")
    self.admin_user_input.addAction(icon_user, QLineEdit.LeadingPosition) #QLineEdit.TrailingPosition
    self.admin_password_input.addAction(icon_base, QLineEdit.LeadingPosition)
    self.admin_password_input.setEchoMode(QLineEdit.Password)
    self.admin_base_input.addAction(icon_lock, QLineEdit.LeadingPosition)
```

**Código 2.2** Método de conectar señales y ranuras del Módulo 1 en VS Code.

## 3. Verificación de Usuario en MongoDB.

Permite ingresar los valores en las líneas de texto y posteriormente verificar en la base de la base de datos su coincidencia.



```python
    # Verificar Usuario existentes en MongoDB
    def crear_base_de_datos(self):
        self.admin_user = self.admin_user_input.text()
        self.admin_password = self.admin_password_input.text()
        self.database=self.admin_base_input.text()
        if not self.admin_user  :
            message_box = QMessageBox()
            message_box.setIcon(QMessageBox.Warning)
            message_box.setText('<font color="red">Error</font>')
            message_box.setInformativeText('<font color="blue">No has ingresado el usuario. </font>')
            message_box.exec_()
            # Ingreso de parametros en las lineas
            self.admin_password_input.clear()
            self.admin_base_input.clear()
            return

        if not self.database  :
            message_box = QMessageBox()
            message_box.setIcon(QMessageBox.Warning)
            message_box.setText('<font color="red">Error</font>')
            message_box.setInformativeText('<font color="blue">No has ingresado la base de autenticación. </font>')
            message_box.exec_()
            self.admin_user_input.clear()
            self.admin_password_input.clear()

            return
        if not self.admin_password :
            message_box = QMessageBox()
            message_box.setIcon(QMessageBox.Warning)
            message_box.setText('<font color="red">Error</font>')
            message_box.setInformativeText('<font color="blue">No has ingresado la contraseña.</font>')
            message_box.exec_()
            self.admin_user_input.clear()
            self.admin_base_input.clear()
            return
        # Calcular el hash de la contraseña ingresada por el usuario

        try:
            self.admin_mongo=AdminMongoDB(self.admin_user, self.admin_password,self.database)
            self.romas=self.admin_mongo.conexion()
            print("Este es el mombre del usuario",self.admin_user)

            # Verifica la existencia del usuario
            self.db = self.romas[self.database]
            self.user_info = self.db.command("usersInfo",self.admin_user)
        #authen=self.db.authenticate(admin_user,admin_password,mechanism='SCRAM-SHA-256')
        # Agrega aqui la lógica para crear una base de datos
            if "users" in self.user_info:
                print("Ingreso a mongoDB exitosamente ")
                self.btn_crear_usuario.setEnabled(True)
                self.etiqueta.setText("Ingreso Exitoso")
            else:
                print("contraseña Incorrecta")
                print("El usuario no existe")
                self.etiqueta.setText("Usuario no Existe")

        except ServerSelectionTimeoutError as e:
            print("No se puede conectar a MongoDB: %s" % e)
            self.etiqueta.setText("No se puede conectar a MongoDB")
        except OperationFailure as e:
            # Si el usuario no existe, pymongo.errors.OperationFailure será lanzada
            self.etiqueta.setText("Usuario no Existe")
            print("Ingrese como administrador y cree un usuario")

            if "Unauthorized" in str(e):
                print(f"El usuario {self.admin_user} no está autorizado")
                self.etiqueta.setText("Usuario no autorizado")
```

**Código 2.3** Método de Verificación de Usuario en MongoDB en VS Code

## 4. Conexión entre el Módulo 1 y 2

Mediante este método se puede conectar entre los 2 módulos. Se crea un objeto de tipo ventana con sus parámetros. Se inicializa en la clase Login módulo 1. Mediante este objeto de tipo Venta se puede utilizar el método show para mostrar la ventana del módulo 2. A demás, se utiliza la variable self.dialog para controlar la vista de los usuarios. Se puede visualizar en la Código 2.4.

```python
    # Conexión con Módulo 2
    def abrir_ventana_crear_usuario(self):


        self.hide()
        self.dialog=Ventana(admin_mongo=self,romas=self.romas,admin_user=self.admin_user)


        if self.admin_user =="jonatan":# Usuario root
            self.dialog.show()  # Muestra la ventana de creación de usuarios
            self.admin_user_input.clear()
            self.admin_password_input.clear()
            self.admin_base_input.clear()
            self.etiqueta.clear()

        elif self.admin_user !="jonatan":
            self.dialog.cambiar_visibilidad_btn_baseM(False)
            self.dialog.show()
            self.btn_crear_usuario.setEnabled(False)
            self.admin_user_input.clear()
            self.admin_password_input.clear()
            self.admin_base_input.clear()
            self.etiqueta.clear()

            # Llama al método para crear el ComboBox en la segunda ventana
            self.dialog.crear_combo_box()
            if self.dialog.comboBox:
                self.verticalLayout_4.addWidget(self.dialog.comboBox)

            self.btn_crear_usuario.setEnabled(False)
            self.admin_user_input.clear()
            self.admin_password_input.clear()
            self.admin_base_input.clear()
            self.etiqueta.clear()
```

**Código 2.4** Método de Conexión entre el Módulo 1 y 2 en VS Code.

## 5. Inicialización de Ventana

Se importa las librerías con las cuales se va a utilizar, posteriormente se carga la plantilla creada para el módulo 2 en Qt Designer. Se inicializa las ranuras para los elementos del módulo. Se puede visualizar en el Código 2.5.

```python
from PyQt5.QtWidgets import QMainWindow, QApplication,QPushButton
import sys
import os
import csv # Para los historicos de Usuario
import pandas as pd
import zipfile
from tabulate import tabulate
import icons_rc
from PyQt5.QtWidgets import QCheckBox,QFileDialog,QAbstractItemView,QInputDialog,QLabel,QGridLayout,QToolBar,QComboBox,QTreeWidgetItem, QTreeWidget,QSizeGrip,QApplication,QTableWidgetItem ,QDialog
from PyQt5.QtCore import QDate,QCoreApplication,Qt,QPropertyAnimation,QEasingCurve,QUrl,QTimer, QDateTime,QTime
from PyQt5 import QtCore,QtWidgets # Para el control el movimiento de la ventana
from pymongo import MongoClient,ReturnDocument
from PyQt5.uic import loadUi
from pymongo.errors import PyMongoError,OperationFailure
from PyQt5.QtGui import QIcon,QPixmap,QColor,QDesktopServices,QFont, QColor,QPalette

import  subprocess# libreria de subproceso
from PyQt5.QtWebEngineWidgets import QWebEngineView # para el gps
import folium
import shutil # Para la descarga de archivos zip
from bson import ObjectId # para el id de la base de datos para eliminarla
import re # para limpiar el texto de la ruta para de descarga del archivo
from Dialogo import DatabaseInputDialog
class Ventana(QMainWindow):
    def __init__(self,admin_mongo=None,romas=None,admin_user=None):
        super().__init__()
        self.admin_mongo=admin_mongo
        self.romas=romas
        self.admin_user=admin_user
        self.all_usuarios = [] # para cargar los usuarios
        #self.ventana_principal=ventana_principal
        loadUi(os.path.join(os.path.dirname(__file__),'Módulo_Gestión.ui'),self)
        # Parámetros de Iniciacion para mover la ventana
        self.bt_normal.hide()
        self.click_posicion = None
        self.click_posicion = None
        self.bt_minimize.clicked.connect(lambda :self.showMinimized())
        self.bt_normal.clicked.connect(self.control_bt_normal)
        self.bt_maximize.clicked.connect(self.control_bt_maximize)
        self.bt_close.clicked.connect(lambda: self.close())
```

**Código 2.5** Librerías y Parámetros para Iniciar el Módulo 2 en VS Code.

## 6. Población de la barra lateral izquierda con los nombres de las bases de datos

Mediante esos métodos se puede cargar los datos a la barra lateral izquierda. Ya realizado la conexión a MongoDB. Se obtiene la lista de las bases de datos en el sistema. Posteriormente se recorrer mediante un ciclo de repetición obteniendo los nombres de las bases y sus colecciones. Tal que, la estructura se asigna a un widget de tipo árbol.

```
        # Crear un QTreeWidget para mostrar bases de datos y colecciones
    def update_tree(self):
        # Obtener la lista de bases de datos
        self.combo_box_databases.clear()
        self.db_list = self.romas.list_database_names()

        # Crear un QVBoxLayout para el QFrame
        self.combo_box_databases.addItems(self.db_list)
        # Limpiar el árbol antes de volver a cargar
        self.tree_widget.clear()
        # Agregar bases de datos y colecciones al árbol
        for self.db_name in self.db_list:
            db_item = QTreeWidgetItem(self.tree_widget, [self.db_name])
            db = self.romas[self.db_name]
            collection_names = db.list_collection_names()
            for collection_name in collection_names:
                collection_item = QTreeWidgetItem(db_item, [collection_name])

        self.tree_widget.itemClicked.connect(self.tree_item_clicked)
```

**Código 2.6** Método para poblar la barra lateral Izquierda en VS Code.

## 7. Filtros de Búsqueda.

Este método es el más importante de la aplicación. Permite que el usuario por medio de un combo box elija un parámetro de búsqueda como se visualiza en el Código 2.7. Lo que permite realizar busque de los archivos en la unidad externa de gran capacidad (USB). Los métodos completos se proporcionan en el Anexo XI.

```
def mostrar_todas_las_tablas(self):
    # Obtener el valor seleccionado del QComboBox
    seleccion = self.campo_busqueda.currentText()
    print("Opcion seleccionada:", seleccion)
    # Obtener el diseño interno del QFrame
    frame_layout = self.frame_tabla.layout()

    # Verificar si hay un diseño interno antes de intentar contar
    if frame_layout is not None:
        # Obtener el número de elementos en el diseño interno
        num_elementos = frame_layout.count()

        # Variable para rastrear si al menos una tabla tiene coincidencia
        alguna_tabla_con_coincidencia = False


        # Filtrar cada tabla por separado
        for i in range(num_elementos):
            widget = frame_layout.itemAt(i).widget()
            if isinstance(widget, QTableWidget):
                table = widget
                table.setSortingEnabled(False)

                # Obtener el índice de la columna correspondiente al tipo seleccionado
                col_seleccionada = self.obtener_indice_columna(table, seleccion)

                # Obtener el texto del QLineEdit
                texto = self.linea.text().lower()
                print("Este es el valor de texto de la QLabel:", texto)
                print("Valor de col_seleccionada:", col_seleccionada)
                print("Valor de self.la_columna_prioridad:", self.la_columna_prioridad)

                # Variable para rastrear si hay coincidencia en esta tabla
                tabla_con_coincidencia = False

                # Filtrar las filas
                for row in range(table.rowCount()):
                    fila_visible = False

                    # Verificar si el índice de columna es válido
                    if col_seleccionada != -1:
                        # Si la seleccion es "Prioridad" y la columna actual es "Prioridad"
                        if seleccion == "Prioridad" and col_seleccionada == self.la_columna_prioridad:
                            # Obtener el widget de la celda que contiene el QCheckBox y el QLabel
                            container_widget = table.cellWidget(row, col_seleccionada)
```

**Código 2.7** Método para filtro de búsqueda en VS Code.

### 2.2.4 PRUEBAS

Una vez analizados las distintas secciones, se procede a realizar pruebas de funcionamiento con los requerimientos de la sección 2.2. Para la evaluación de prueba se utiliza VNC Viewer para conectarse a la plataforma de Raspberry PI [22].

#### 2.2.4.1 Prueba de Instalación de MongoDB en Raspberry PI.

Para realizar la Instalación en Raspberry Pi, se sigue en tutoría proporcionado por Andy Felong [4]. Para la verificación de la instalación se procede a verificar el estado del servicio de MongoDB en la Raspberry PI como se muestra en la figura 2.2.8

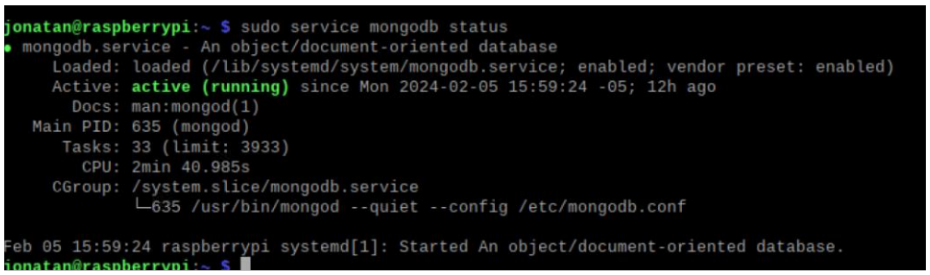



**Figura 2.2.8** Servicio de MongoDB en Raspberry PI.

- **Activación de Autenticación en MongoDB**

Como se puede visualizar en la figura 2.2.9. Está habilitada la seguridad, es necesario realizarlo para brindar mayor seguridad al usuario. Mediante la habilitación es necesario ingresar con un usuario y clave.

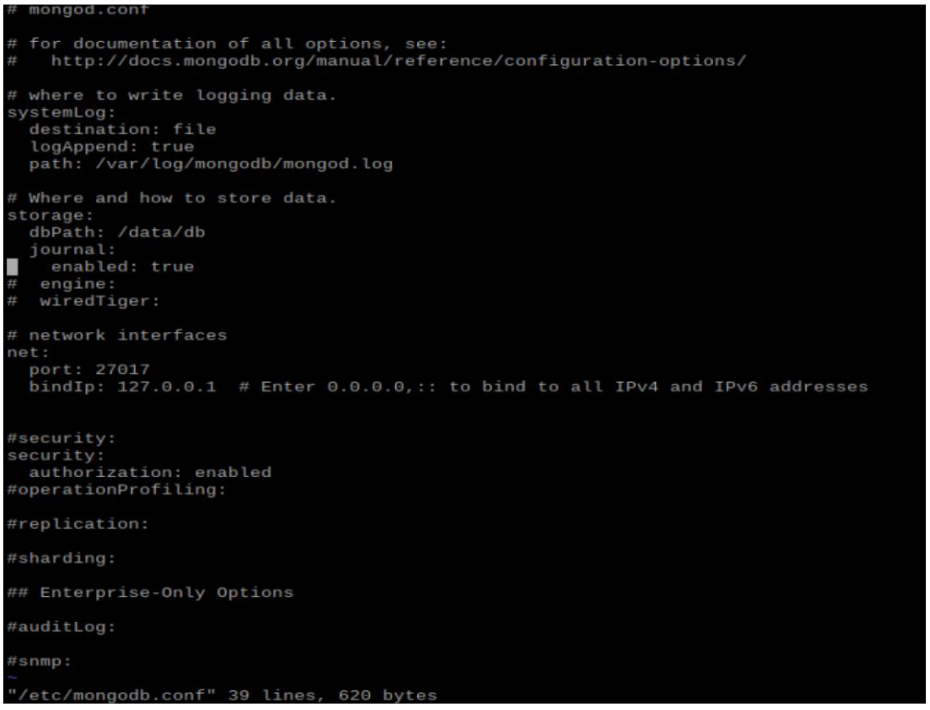



**Figura 2.2.9** Servicio de MongoDB en Raspberry PI.

- **Versión de MongoDB instalado**

Para acceder a la versión de MongoDB en necesario de colocar el comando mongo como se visualiza en la figura 2.2.10.


**Figura 2.2.10** Versión Instalada de MongoDB en Raspberry PI.

### 2.2.4.2 Inicio de Sesión

El inicio de sesión se ingresa a través de usuarios por defecto en los roles: read, readWrite, dbOwner. La creación de usuarios se detalla en el Anexo XII

Se puede visualizar en la figura 2.2.11. El ingreso de los parámetros del usuario. En este caso se ingresa correctamente. En parámetro implícito que se encuentre abajo del botón Inicio de Sesión, el cual muestra el Ingreso Exitoso. Para acceder al módulo dos en necesario presionar el botón con el icono de →]. Para las restricciones en los parámetros de ingreso en el Módulo 1, usuarios no encontrados. Se expone en Anexo XIII.

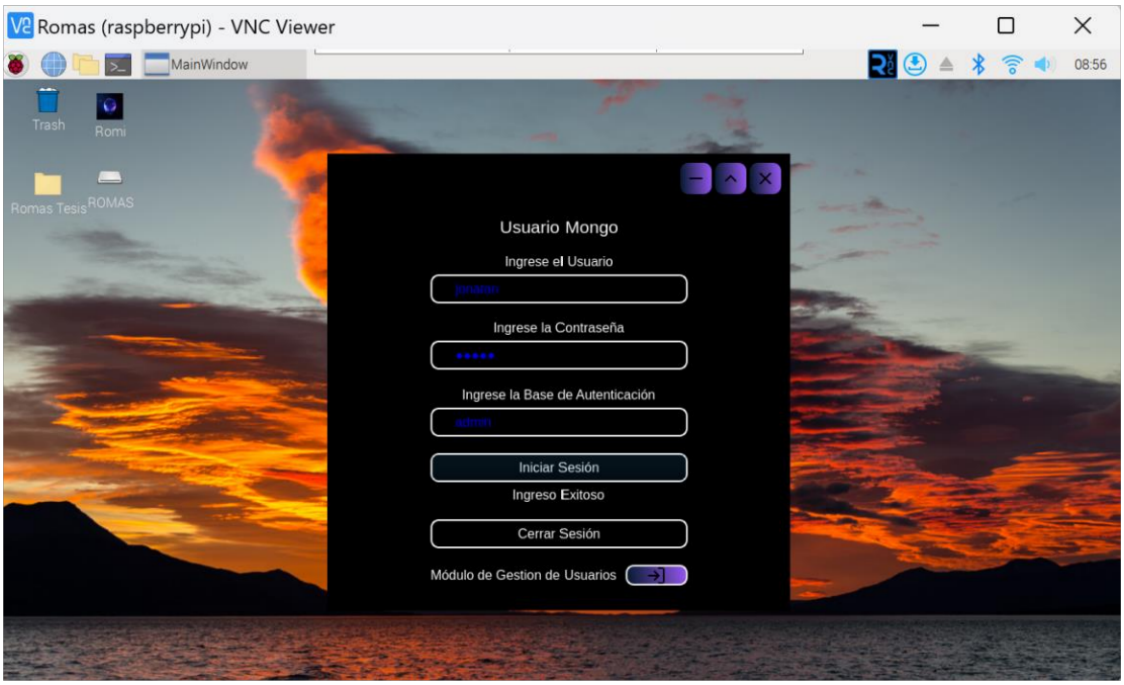



**Figura 2.2.11** Inicio de Sesión de Módulo 1.

### 2.2.4.3 Barra Izquierda Lateral

Una vez realizado el inicio se sesión. El usuario puede acceder al Módulo 2. La interfaz que observara el usuario privilegiado se muestra en la figura 2.2.12. Se puede visualizar en la barra izquierda lateral contiene las bases de datos del sistema y las colecciones (Subelemento). Se puede observar en la figura 2.2.7. El modo previsualización de QT Designer. La Barra lateral Izquierda del Módulo 2. El diseño de la barra lateral izquierda que contendrá los nombres de las bases de datos del sistema.


**Figura 2.2.12** Inicio de Sesión de Módulo 1.

### 2.2.4.4 Contenedor de Botones

Permite gestionar los datos registrados actualmente en la base de datos los cuales son: Base, Descargas, Histórico, GPS Imag. Los detalles del funcionamiento de cada botón se los puede visualizar en el Anexo XIV


**Figura 2.2.13** Contenedor de Botones Módulo 2.

### 2.2.4.5 Zona de Filtrado

Se procede a realizar los filtros por los parámetros Fecha, Hora, Ubicación, Prioridad con los cuales se puede ver los registros que cumplen con los parámetros de filtrado. El proceso para el filtrado se lo puede visualizar en el Anexo XV

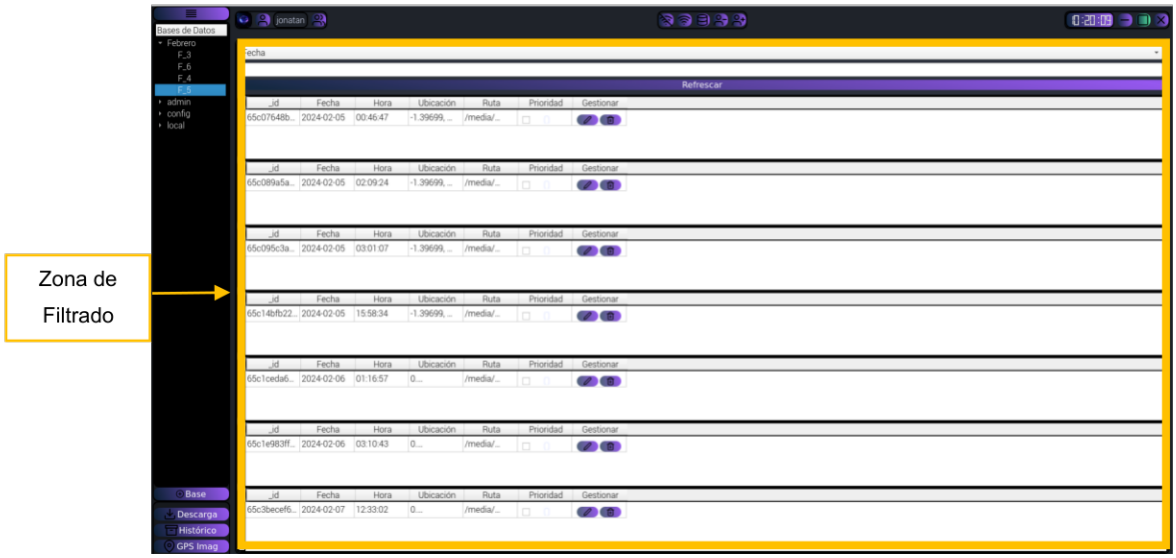



**Figura 2.2.14** Zona de Filtrado.

### 2.2.4.6 Barra Superior Izquierda

Como se puede observar en la figura 2.2.15 contiene un Icono de Usuario y el nombre del usuario que ingreso en el sistema. Para más detalle de funcionamiento en Anexo XVI

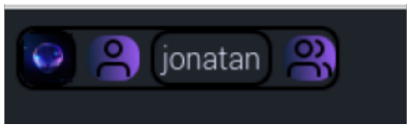



**Figura 2.2.15** Barra Superior Izquierda.

### 2.2.4.7 Barra Superior Central

Se puede visualizar en la figura 2.2.16 una barra superior central, la cual contiene 5 botones. El primer botón permite desactivar la Interfaz WLAN. El segundo botón permite activar la interfaz WLAN. El tercer botón permite eliminar bases de datos disponibles en el sistema. En el cuarto y quinto botón, Permite gestionar usuario en el sistema. Detalle de funcionamiento de la barra superior central se encuentra en el Anexo XVII

**Figura 2.2.16** Barra Superior Central.

### 2.2.4.8 Barra Superior Derecha

Gestiona la hora del sistema, la visualización de la ventana y el cierre del sistema. Los detalles del uso se muestran el en Anexo XVIII

**Figura 2.2.17** Barra Superior Derecha.

# 3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

## 3.1 RESULTADOS

En base a los cuadros realizados en el apartado 2.2.4. Funcionalidades del Sistema. Se cumple los requerimos planteados para el diseño del prototipo. Las pruebas de funcionamiento en la base de datos se pueden visualizar en el Anexo XIX.

### 3.1.1 INSTALACIÓN DE BASE DE DATOS

Se cumple todos los requerimientos de la base de datos para trabajar como un servidor local de archivos de imágenes con secuencia de imágenes. Se instaló correctamente la base de datos sobre la plataforma de Sistema Embebido Raspberry PI.

**Tabla 3.1** Verificación de instalación de la base de datos

| INSTALACIÓN DE MONGODB | | |
|---|---|---|
| Descripción | Realizado | No Realizado |
| Instalación | √ | |
| Habilitación de Seguridad(Autenticación) | √ | |
| Verificación de Versión | √ | |
| Cambio de Puerto por Defecto | | √ |

### 3.1.2 Módulo 1 “LOGIN”

Como se presenta en la tabla 3.1 del Módulo 1, el ingreso de nombre del usuario, contraseña, base de datos de autenticación, así como la verificación de botones para acceder a la autenticación con el mecanismo SCRAM 256 son realizados exitosamente, A demás, se verifica exitosamente el control de la venta mediante los botones de la barra superior derecha.

**Tabla 3.2** Módulo 1

| MODULO 1 "Login" | | | |
|---|---|---|---|
| **Descripción** | **Atributo** | **Funciona** | **No Funciona** |
| Nombre Usuario | Parámetro 1(Línea de texto): | √ | |
| Contraseña de Usuario | Parámetro2(Línea de texto): | √ | |
| BDD de Autenticación | Parámetro 3(Línea de texto): | √ | |
| Verifica en el Sistema | Parámetro 4(Botón) | √ | |
| Mensaje de Verificación en el Sistema | Parámetro 5(Label) | √ | |
| Cerrar Sesión | Parámetro 6 (Botón) | √ | |
| Control de Ventana | Barra Horizontal Derecha | √ | |

### 3.1.3 Módulo 2

Se presenta el análisis de resultados en la tabla 3.3. las funcionalidades se la interfaz de gestión de usuario se realiza satisfactoriamente, así como las zonas que prevén la interacción con el usuario para el manejo de archivos comprimidos.

**Tabla 3.3** Resultados del Módulo 2

| MÓDULO 2 | | | | |
|---|---|---|---|---|
| **Descripción** | **Tipo** | **Atributo** | **Funciona** | **No Funciona** |
| Contenedor de base de datos | Barra Izquierda Lateral | Barra Izquierda Lateral | √ | |
| | Contiene de botones | Descarga | √ | |
| Contiene botones para | | Histórico | √ | |
| | | GPS img | √ | |
| | | Base | √ | |
| | Zona de filtrado | Filtro por Fecha | √ | |
| Zona que permite al usuario | | Filtro por Hora | √ | |
| realizar filtros | | Filtro por Ubicación | √ | |
| | | Filtro por Prioridad | √ | |
| Barra Contenedora de la | Barra Superior Izquierda | Cambio de usuario | √ | |
| etiqueta y regreso de modulo | | Etiqueta | | √ |
| | Barra Superior Central | WI-FI | √ | |
| Barra contenedora de | | Desactivar | √ | |
| Botones para Gestión de | | WI-FI | √ | |
| WIFI, Gestión de Usuario | | Eliminar Base de Datos | √ | |
| y Bases de Datos | | Eliminar Usuario | √ | |
| | | Agregar Usuario | √ | |
| Contiene un Display con | Barra Superior Derecha | Display | √ | |
| la hora y 3 botones para | | Minimizar | √ | |
| controlar la ventana | | Maximizar | √ | |
| | | Cerrar | √ | |

- **Recuperación de Archivo mediante Filtro de la Unidad Externa (USB)**

Para visualizar el contenido del archivo que se encuentra en la unidad externa de gran capacidad (USB). El usuario debe realizar el filtro para acceder a la información de la cabecera. Una vez encontrado el archivo deseado mediante el filtro, el usuario debe dar clic sobre la celda Ruta. Esta columna abre la ruta en donde se encuentra el archivo y lo selecciona en un cuadro de dialogo como se muestra en la siguiente figura 3.1.1

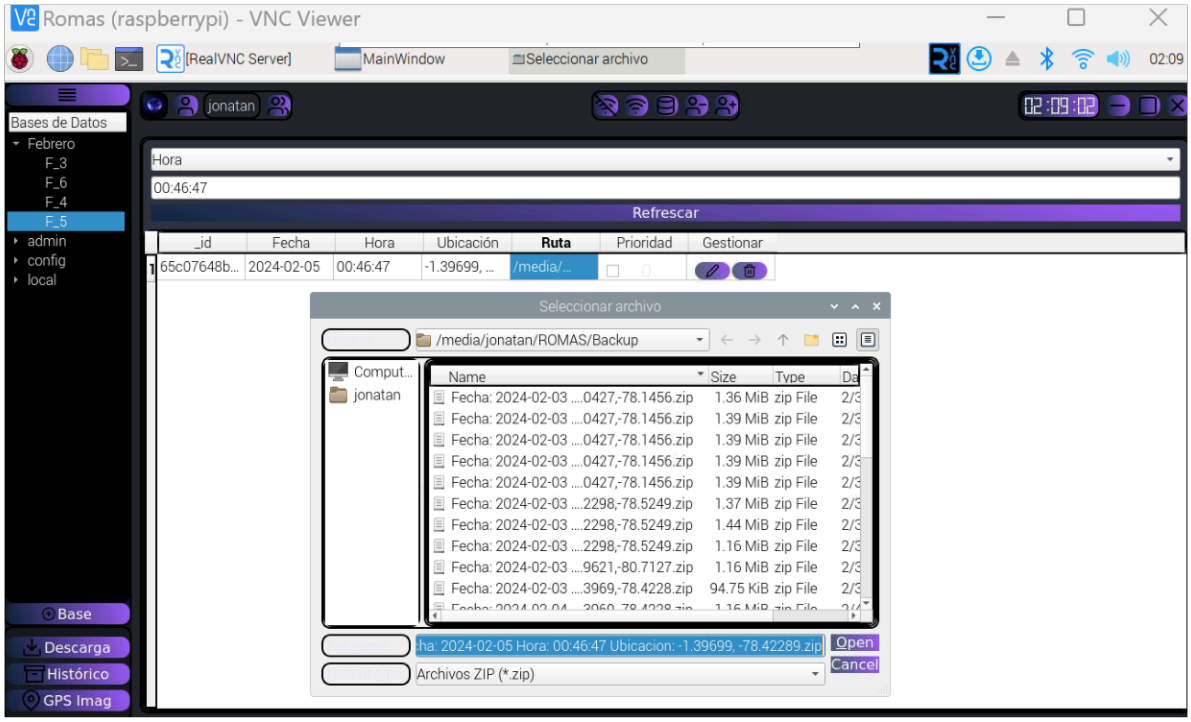



**Figura 3.1.1** Visualización de Nombre del Archivo Filtrado por Parámetro Hora.

- **Contenido del archivo Zip Encontrado.**

Una vez encontrado el archivo se procede a verificar su contenido como se muestra en la figura 3.1.2. El archivo ZIP contiene imágenes tipo JPG

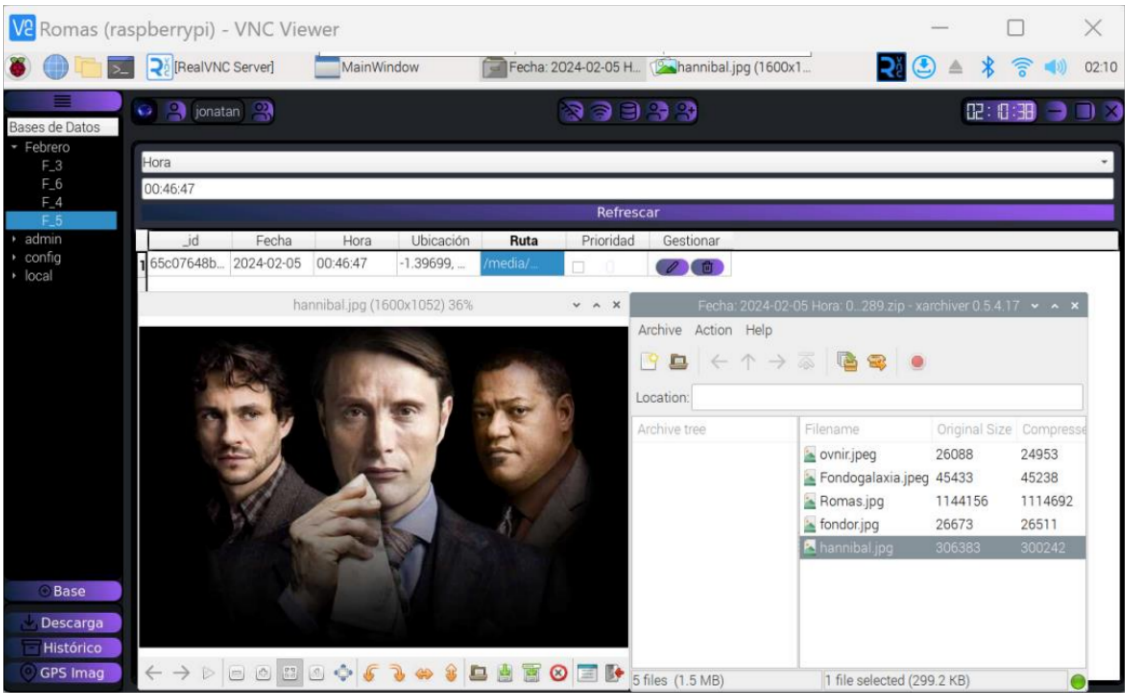



**Figura 3.1.2** Visualización del Contenido del Archivo ZIP.

Para verificar su coincidencia con el archivo recuperado mediante el filtro. Se busca el archivo en la unidad externa de gran capacidad (USB), y se verifica su contenido como se visualiza en la figura 3.1.3.

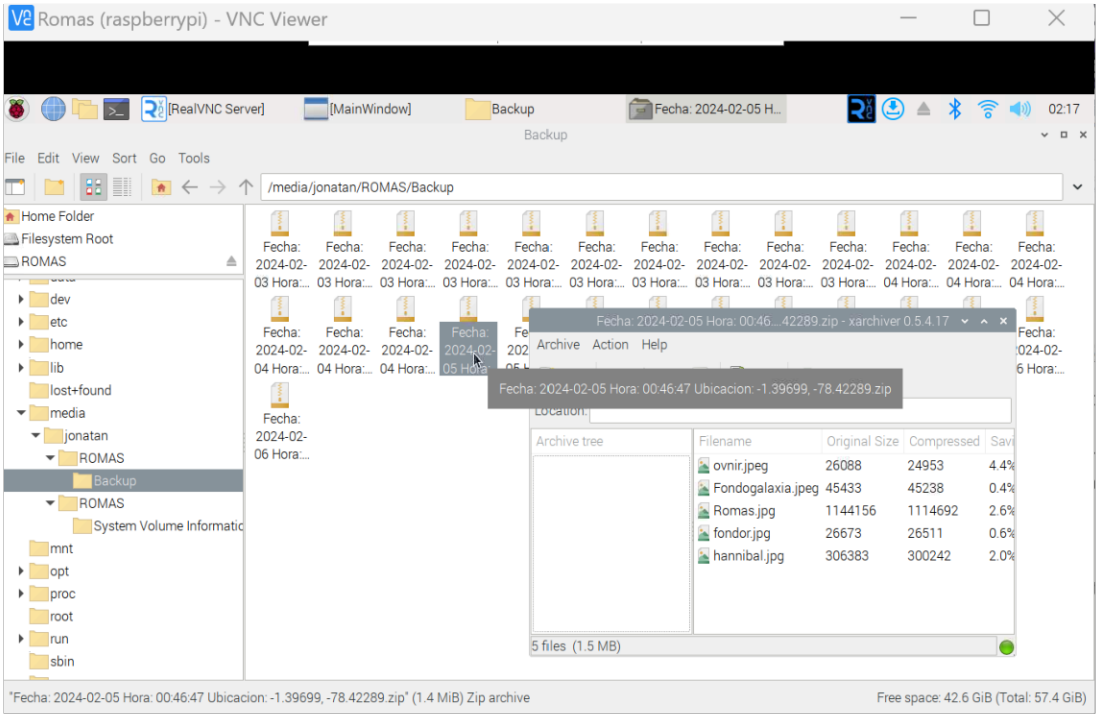



**Figura 3.1.3** Visualización del Contenido del Archivo ZIP en la USB.

## 3.2 CONCLUSIONES

- Raspberry Pi dispone de variedad de paquetes que se encuentra en el repositorio del Sistema Operativo, los cuales permiten descargar aplicaciones para la construcción de interfaces gráficas de una manera sencilla. Al momento de crear la plantilla se puede acceder a los widgets contenedores de QT Designer en el script de Python utilizando la biblioteca de Pyqt5. De esta manera se puedo agregar las tablas para crear los filtros en zona central de aplicación.
- Raspberry Pi es una plataforma de sistema embebido que permite trabajar como un servidor local utilizando un base de datos NoSQL. De esta manera se puede gestionar archivos comprimidos desde una unidad externa sin sobrecarga a la plataforma. La forma que se guarda las referencias de la ruta de la unidad externa en la cual se encuentra el archivo, en la base de datos no relacional, permite localizar con facilidad el archivo en la unidad externa.
- MongoDB es una base de datos NoSQL que utiliza la estructura de datos sin formato, los que permite al usuario recuperar información deseada de una forma rápida con ayuda de las herramientas de Python.
- MongoDB es muy versátil al momento de estructurar sus datos. Dispone de variedad de métodos para buscar parámetros específicos que se encuentre en la base de datos en este presente componente se utilizó para permitir generar referencias a las metadas al momento de realizar los filtros.
- Raspberry Pi tiene limitación al momento de trabajar con Bases de datos de tipos No relacional, es necesario verificar las dependencias de software y hardware desde el sitio oficial.
- Después de haber realizado el trabajo se pudo verificar que el uso de la ingeniería requerimientos resulta indispensable porque permite al desarrollador visualizar todos los componentes a implementarse y sus relaciones.
- Si bien es cierto simplifica enormemente el diseño de la presentación del aplicativo, genera adicionalmente segmentos de código que no necesariamente van hacer utilizados y que deben ser remplazados por la funcionalidad de desarrollador, esto eventualmente requiere analizar detenidamente lo antes señalado incrementando el volumen de trabajo.

## 3.3 RECOMENDACIONES

Al momento de desarrollar la aplicación se fueron presentando distintas anomalías por lo cual se considera las siguientes recomendaciones.

- Raspberry Pi tiene limitación al momento de trabajar con bases de datos de tipos no relacional, es necesario verificar las dependencias de software y hardware desde el sitio oficial para que cumplas los requerimientos y se realice con éxito la instalación.
- Se recomienda verificar el sistema de archivo de la unidad externa de gran capacidad, para no prever problemas al momento de enviar archivos comprimidos con el formato ext4 que utiliza Raspberry.
- La implementación del sistema de almacenamiento de archivos de secuencia de imágenes con estampa de tiempo y de localización se construye dos módulos para la gestión de archivos. Se recomienda la creación de un tercer módulo el cual permita manejar la fragmentación de un sistema de almacenamiento masivo, cuando el proyecto entre en su etapa de pruebas con grandes volúmenes de información.
- Durante el proceso realizado para definir la plataforma con la cual se desarrolló este proyecto se pudo verificar que nuevas plataformas con mejores prestaciones y sistemas operativos estable pudieron considerarse, por esta razón se recomienda que los cursos relacionados con esta temática aborden estas nuevas y versátiles plataformas de sistemas embebidos.

# 4 REFERENCIAS


[1] V. F. -G. Allport, EL HOMBRE EN BUSCA DE SENTIDO, Barcelona: EDITORIAL HERDER, 1991.

[2] MongoDB Companny, «SON and BSON,» 1 Agosto 2023. [En línea]. Available: https://www.mongodb.com/es/json-and-bson. [Último acceso: 28 Mayo 2023].

[3] J. Peyton, «Hands-on Introduction to Embedded Systems & IOT,» [En línea]. Available: https://tinyurl.com/2u6bt459. [Último acceso: 22 Mayo 2023].

[4] A. Felong, « MongoDB 5.0 under Raspberry Pi OS (64-bit),» 8 Agosto 2021. [En línea]. Available: https://andyfelong.com/2021/08/mongodb-4-4-under-raspberry-pi-os-64-bit-raspbian64/. [Último acceso: 28 Mayo 2023].

[5] W. Lawrence, «Embedded System,» 25 Noviembre 2023. [En línea]. Available: https://www.guru99.com/embedded-systems-tutorial.html. [Último acceso: 24 Junio 2023].

[6] TechTarget, «Embedded System,» 1 Agosto 2023. [En línea]. Available: https://www.techtarget.com/iotagenda/definition/embedded-system . [Último acceso: 20 Junio 2023].

[7] Oloyede O. A, Akinwole A. Yekini N.A, Akinade A. O, «OVERVIEW OF EMBEDDED SYSTEM & ITS APPLICATION».,» 1 Agosto 2023. [En línea]. Available: https://www.researchgate.net/publication/361562662_OVERVIEW_O_EMBEDDED_SYSTEM_ITS_APPLICATION . [Último acceso: 28 Septiembre 2023].

[8] R. Electronics, «Explicación de los sistemas embebidos,» [En línea]. Available: https://reboundeu.com/es/insights/blog/embedded-systems-explained-15/. [Último acceso: 20 Agosto 2023].

[9] H. Ashtari, «What Are Embedded Systems? Meaning, Components, and Applications,» 30 Octubre 2023. [En línea]. Available: https://www.spiceworks.com/tech/tech-general/articles/what-are-embedded-systems/. [Último acceso: 20 Agosto 2023].

[10] Orthigone, A. Raymond., «What You Need to Know About Embedded Systems in Medical Applications,» 2 Junio 2021. [En línea]. Available: https://orthogone.com/embedded-systems-in-medical-applications/ . [Último acceso: 20 Octubre 2023].

[11] Medium, « The Use of embedded systems in smart home technology,» 27 Junio 2023. [En línea]. Available: https://medium.com/@iiesbangalorebl/the-use-of-embedded-systems-in-smart-home-technology-58668621fce8.

[12] EKTOS, «TinyML Applications Transforming The Manufacturing Industry,» 19 Octubre 2023. [En línea]. Available: https://www.linkedin.com/pulse/4-tinyml-applications-transforming-manufacturing-industry.

[13] Medium, «The Vital Role of Embedded Systems in Electric Vehicles,» 9 Agosto 2023. [En línea]. Available: https://medium.com/@embeddedbox.official/the-vital-role-of-embedded-systems-in-electric-vehicles-9771cbd450bc.

[14] H. T. Developer, «Arduino UNO: The Key to Building Your Own Electronics,» 25 Junio 2023. [En línea]. Available: https://hackthedeveloper.com/what-is-an-arduino-uno/.

[15] SoftwareEngineenering, «Software engineering history,» [En línea]. Available: https://proyectoarduino.com/arduino-uno-r3/#google_vignette.

[16] Arduino, «What power supply can I use with my Arduino board?,» [En línea]. Available: https://support.arduino.cc/hc/en-us/articles/360018922259-What-power-supply-can-I-use-with-my-Arduino-board-.

[17] F. Projects, «AN INTRODUCTION TO ARDUINO UNO PINOUT,» 7 Noviembre 2018. [En línea]. Available: https://duino4projects.com/an-introduction-to-arduino-uno-pinout/.

[18] W.-M. Lee y C. Chng, «Introduction to IoT Using the Raspberry Pi,CODE Magazine,» 31 Agosto 2022. [En línea]. Available: https://www.codemag.com/Article/1607071/Introduction-to-IoT-Using-the-Raspberry-P. [Último acceso: 20 Diciembre 2023].

[19] GeeksforGeeks, «Raspberry-Pi a computer for Geeks,» [En línea]. Available: https://www.geeksforgeeks.org/raspberry-pi-a-computer-for-geeks/ . [Último acceso: 26 Diciembre 2023].

[20] M. electronics, «Raspberry Pi 4 Modelo B / 8GB RAM,» [En línea]. Available: https://mcielectronics.cl/shop/product/raspberry-pi-4-modelo-b-8gb-ram-raspberry-pi-28296/. [Último acceso: 26 Diciembre 2023].

[21] R. Pi, «Raspberry Pi 4 Model B,» Diciembre 2023. [En línea]. Available: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf?_gl=1*xarqxd*_ga*OTE2MzQ5NzMzLjE3MDU3ODA5NTA.*_ga_22FD70LWDS*MTcwNTg5MTA4OS4yLjEuMTcwNTg5MTQxOS4wLjAuMA... [Último acceso: 26 Diciembre 2023].

[22] RaspberryTips, «Getting Started With VNC on Raspberry Pi (Bookworm update),» [En línea]. Available: https://raspberrytips.com/use-vnc-raspberry-pi/.

[23] J. Butts, «How to SSH Into Your Raspberry Pi,How-ToGeek,» 10 Julio 2022. [En línea]. Available: https://www.howtogeek.com/768053/how-to-ssh-into-your-raspberry-pi/.

[24] RaspberryTips, «What is APT,» [En línea]. Available: https://raspberrytips.com/glossary/apt/.

[25] RaspberryTips, «How To Quickly Identify Which OS Is Running On Raspberry Pi,» [En línea]. Available: https://raspberrytips.com/which-raspberry-pi-os-is-running/.

[26] RaspberryTips, «Upgrade Raspberry Pi OS to the Latest Version (2024),» [En línea]. Available: https://raspberrytips.com/update-raspberry-pi-latest-version/.

[27] RaspberryTips, «15 Best Operating Systems for Raspberry Pi,» [En línea]. Available: https://raspberrytips.com/best-os-for-raspberry-pi.

[28] G. Phillips, «How to Install Windows 10 IoT Core on Raspberry Pi 3",,» 12 Julio 2018. [En línea]. Available: https://www.makeuseof.com/tag/raspberry-pi-windows-10-iot-core-projects. [Último acceso: 20 Diciembre 2023].

[29] C. Colimnist, «How to Install Windows 10 on Raspberry Pi 4 [Step-by-Step],EaseUS,» 2 Enero 2024. [En línea]. Available: https://www.makeuseof.com/tag/raspberry-pi-windows-10-iot-core-projects/.

[30] A. F. Platform, «Ejemplo de ByteArray: Lectura de un archivo .zip,» [En línea]. Available: https://help.adobe.com/es_ES/as3/dev/WS5b3ccc516d4fbf351e63e3d118666ade46-7d53.html.

[31] P. S. Foundation, «zipfile — Work with ZIP archives, Python 3.9.7 Documentation,» 2024. [En línea]. Available: https://docs.python.org/3/library/zipfile.html.

[32] Fileformat, «¿Qué es un archivo TAR?," Formatos de archivos de compresion,Docuementacion,» [En línea]. Available: https://docs.fileformat.com/es/compression/tar/.

[33] P. S. Foundation, «Tarfile — Leer y escribir archivos tar," Python 3.9.7,Documentation,» 2024. [En línea]. Available: https://docs.python.org/es/3/library/tarfile.html#tarfile.open.

[34] T. F. a. K. Dube, «What is a JPG (JPEG) File?, How to Open, Edit, and Convert JPG or JPEG Files," Lifewire,» 7 Julio 2021. [En línea]. Available: https://www.lifewire.com/jpg-jpeg-file-4139913.

[35] D. G. Ionos, «EXIF: visualizar y guardar los metadatos de las imágenes,» 23 Octubre 2020. [En línea]. Available: https://www.ionos.mx/digitalguide/paginas-web/diseno-web/que-son-los-datos-exif/.

[36] T. F. a. K. Dube, «Extract EXIF Data from an Image using Python,» 18 Octubre 2022. [En línea]. Available: https://geekyhumans.com/extract-exif-data-an-image-using-python/.

[37] IBM, «¿What is MongoDB?,» [En línea]. Available: https://www.ibm.com/topics/mongodb.

[38] M. Developer, «Install & Configure MongoDB on the Raspberry Pi,» [En línea]. Available: https://www.mongodb.com/developer/products/mongodb/mongodb-on-raspberry-pi/.

[39] MongoDB, «Nota de producto,Desarrollo de aplicaciones,» [En línea]. Available: https://www.mongodb.com/docs/v5.0/administration/production-notes/#std-label-prod-notes-supported-platforms-x86_64.

[40] IBM, «¿Qué es CouchDB?,» [En línea]. Available: https://www.ibm.com/mx-es/topics/couchdb.

[41] S. Canónico, «Instalar CouchDB en Raspberry Pi,» 19 Enero 2024. [En línea]. Available: https://snapcraft.io/install/couchdb/raspbian.

[42] L. Pozo, «Qt Designer and Python: Build Your GUI Applications Faster,Real Python,» [En línea]. Available: https://realpython.com/qt-designer-python/.

[43] P. Geeks, «Building GUI Applications with Qt Designer and Python,» [En línea]. Available: https://pythongeeks.org/python-qt-designer/.

[44] Unipython, «PYQT5 INTERFÁCES GRÁFICAS CON PYTHON,» 19 Enero 2024. [En línea]. Available: https://unipython.com/pyqt5-interfaces-graficas-con-python/.

[45] M. Fitzpatrick, «he complete PySide2 tutorial — Create GUI applications with Python,» 31 Enero 2024. [En línea]. Available: https://www.pythonguis.com/pyside2-tutorial/.

[46] M. Heller, «What is Visual Studio Code? Microsoft’s extensible code editor , InfoWorld,» 8 Julio 2022. [En línea]. Available: ttps://www.pythonguis.com/pyside2-tutorial.

[47] Visure, «Ingeniería de Requerimientos: Paso a Paso, Visure Solutions,» [En línea]. Available: https://visuresolutions.com/es/blog/proceso-de-ingenieria-de-requisitos/.

[48] GeeksforGeeks, «Requirements Engineering Process in Software Engineering,» [En línea]. Available: https://www.geeksforgeeks.org/software-engineering-requirements-engineering-process/.

[49] J. E. Ortiz., M. A. Pillalaza, «Desarrollo de Módulos con Raspberry Pi utilizando Comunicaciones Inalámbricas, Para el Laboratorio de Microprocesadores de la Esfot,» Tesis de Grado, Escuela. de Formación de Tecnólogos, Escuela Politécnica Nacional, Quito, 2018.

[50] MongoDB, «Collection Methods,» [En línea]. Available: https://www.mongodb.com/docs/v5.0/reference/method/js-collection/#collection-methods.

[51] J. Ruiz Osorio, «Ciclo de Vida del Software. Blog de Juliana Ruiz Osorio,» [En línea]. Available: https://ruizji.blogspot.com/2012/02/ciclo-de-vida-del-software.html. [Último acceso: 20 Enero 2024].

[52] J. Robertson., S. Robertson, «Volere Requierement Specification Template, » Available: https://www.cs.uic.edu/~i440/VolereMaterials/templateArchive16/c%0Volere%20template16.pdf