ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DESARROLLO DE UN PROTOTIPO PARA EL MONITOREO Y CONTROL DE ACCESO DE LAS UNIDADES DE TRANSPORTE ESTUDIANTIL DE LA EPN MEDIANTE UN APLICATIVO WEB Y UN APLICATIVO MÓVIL.

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICAY REDES DE INFORMACIÓN.

ALCOCER BAHAMONDE MARCOS ANDRES

CHIGUANO FERNÁNDEZ MYRIAM TATIANA

DIRECTOR: ING. XAVIER ALEXANDER CALDERÓN HINOJOSA M.Sc.

Quito, diciembre 2020

AVAL

Certifico que el presente trabajo fue desarrollado por Alcocer Bahamonde Marcos Andrés y Chiguano Fernández Myriam Tatiana, bajo mi supervisión.

Ing. Xavier Calderón M.Sc.
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Nosotros, Alcocer Bahamonde Marcos Andrés y Chiguano Fernández Myriam Tatiana, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración dejamos constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

Alcocer Bahamonde Marcos Andrés	Chiguano Fernández Myriam Tatiana

DEDICATORIA

A mis padres Fabián y Angélica, de los cuales me siento muy orgulloso, ya que en todo este tiempo han sido un pilar fundamental, brindándome el cariño, paciencia y apoyo necesario para culminar esta etapa en mí vida.

A la memoria de mi querido abuelito Julio Cesar Alcocer Díaz, que, a base de su sabiduría y consejos, me ha impulsado a ser una persona con valores y responsable.

A mi hermano Carlos, que siempre ha tenido la predisposición de apoyarme en todas las dificultades que se me han presentado a lo largo de mi vida y por estar pendiente de mí en todo momento.

Andrés Alcocer B.

DEDICATORIA

A mis padres, Jorge y María, que son el motor fundamental de mi vida, quienes me han ayudado en todo momento y me han brindado sus manos desinteresadamente siempre que lo he necesitado. Gracias por su compresión, paciencia, sacrificio y a sus palabras de aliento, he podido culminar esta importante etapa de mi vida.

A mi hermana que, a pesar de gran diferencia de pensamientos, siempre me ha apoyado en los momentos que han sido necesarios.

Tatiana Chiguano F.

AGRADECIMIENTO

Agradezco primeramente a Dios por permitirme terminar esta meta, a mis padres y familia por ser los promotores e inspiración de todos mis sueños.

A la Escuela Politécnica Nacional y al personal docente por impartirme el conocimiento para conseguir una sólida formación profesional. En especial a mi profesor y director M.Sc. Xavier Calderón, quien con su esfuerzo y tiempo me ha encaminado a realizar este proyecto.

A el Ph.D. Fernando Carrera, por su desinteresada ayuda para que este proyecto inicie y culmine de la mejor manera.

A mi compañera de tesis por los conocimientos compartidos, por la colaboración y paciencia entregados a lo largo del desarrollo de este proyecto y en general todos mis amigos que fueron parte de esta bonita etapa en la universidad.

Andrés Alcocer B.

AGRADECIMIENTO

A mis padres por su apoyo incondicional.

A M.Sc. Xavier Calderón, por su tiempo, apoyo y correcciones brindadas a lo largo del desarrollo de este plan de titulación.

A Ph.D. Fernando Carrera por su predisposición y ayuda.

A mi compañero de tesis por sus conocimientos, paciencia, esfuerzo, dedicación y compromiso.

A la Escuela Politécnica Nacional por haberme acogido en sus aulas.

A cada uno de los docentes por sus enseñanzas y su tiempo brindado.

Y finalmente a aquellos verdaderos amigos que me acompañaron en esta etapa.

Cuando sientes que el mundo se derrumba, no hay mejor refugio que tu familia y los verdaderos amigos.

Tatiana Chiguano F.

ÍNDICE DE CONTENIDO

RESUMEN

El presente trabajo de titulación tiene como objetivo desarrollar un prototipo para el monitoreo y control de acceso de las unidades de transporte estudiantil de la EPN mediante un aplicativo web y móvil.

Para el monitoreo y control de acceso se utilizan varios módulos que capturan información a través de una placa Raspberry Pi, esta información será enviada a una base de datos para finalmente presentarlos en los aplicativos expuestos, además, cuenta con un sistema de videovigilancia otorgado por el software ZoneMinder.

El aplicativo web será de uso exclusivo para la persona encargada de la administración de servicio estudiantil, estará codificado en HTML y JavaScript mientras el aplicativo móvil será orientado tanto al administrador como a los estudiantes y se utilizará XML y Java para su codificación.

El documento estará conformado por cinco capítulos:

El primer capítulo presenta conceptos básicos y fundamentales sobre: Raspberry Pi métodos de acceso, sistemas de geolocalización, sistemas de videovigilancia, base de datos NoSQL, plataforma Firebase, plataforma Dataplicity, lenguajes de programación que intervienen en el prototipo y metodología Kanban.

El segundo capítulo contiene la metodología utilizada para desarrollar el prototipo, este capítulo está dividido en dos partes: la parte de diseño que comprende a la recolección de requerimientos y diseño de los componentes que conforman el prototipo.

El tercer capítulo contiene la implementación del prototipo como la conexión, instalación, configuración y codificación de los diferentes componentes.

El cuarto capítulo presenta los resultados obtenidos de las pruebas realizadas. Además, se presenta los resultados de las encuestas de satisfacción realizadas al personal que interviene en el servicio de transporte estudiantil, choferes, estudiantes y administrador.

El quinto capítulo enlista las conclusiones y recomendaciones obtenidas durante el desarrollo del prototipo. Y finalmente se presentan los anexos.

PALABRAS CLAVE: Raspberry Pi, ZoneMinder, Firebase, JavaScript y Java.

ABSTRACT

The current document's objective is showing the development of a prototype that will be able to monitor and control the access of the EPN transport units through a web and mobile app.

Some modules are used to monitor and control the access; that modules capture the information using a Raspberry Pi module. This information is sent to a data base to finally display it in the mentioned applicative; besides, it has a Close Circuit Television Service (CCTV) given by the ZoneMinder software.

The web applicative will be used only by the person that manages the students transport service; this applicative will be coded in HTML & JavaScript. Meanwhile, the mobile applicative will be used by both the manager and the students; and it will be coded in XML and Java.

This document is made by five chapters:

The first chapter shows basic and essential concepts about: Raspberry Pi, access ways, geolocation systems, video surveillance systems, NoSQL data bases, Firebase platform, Dataplicity platform, programmation languages that intervene in the prototype; and the Kanban methodology.

The second chapter contains the methodology used to develop the prototype. This chapter is splited in two parts: the design part that has the recollection of requirements and comprises design that includes the prototype.

The third chapter contains implementation of prototype as the connection, installation, configuration and codification the different components.

The fourth chapter shows the results that were gotten by the tests made. Further, it shows the satisfaction poll results made to the people that intervene in the student transport service: drivers, students and the manager.

The last chapter lists the conclusions and recommendations that were gotten during the prototype development. Finally, annexed information is shown.

KEY WORD: Raspberry Pi, ZoneMinder, Firebase, JavaScript y Java.

1. INTRODUCCIÓN

Los sistemas de monitoreo de unidades de transporte público en ciertos países, en su mayoría de Europa, han evolucionado a la par con la tecnología que se va desarrollando, es así como estas unidades actualmente proporcionan información como: la ubicación de los buses, conteo de pasajeros y video vigilancia. El envió y/o recepción de esta información en un tiempo definido genera una comunicación de datos útil para quien monitorea y/o controla la unidad de transporte [1]. En nuestro país se han usado ciertos mecanismos como sensores infrarrojos para determinar el número de pasajeros dentro del transporte, siendo esta una tecnología limitada ya que no ha evolucionado con los años en cuanto al control de pasajeros [2]. Así mismo para el monitoreo de eventos y respuesta inmediata para emergencias dentro de las unidades de transporte público se ha implementado el Servicio Integrado de Seguridad ECU 911 [3].

Para el servicio de transporte estudiantil dentro del país solo se han resuelto los problemas de seguridad vial y mecánicos que estas unidades pueden llegar a tener, pero no se ha tomado en consideración aspectos que de una u otra forma ayudan a brindar una mejor experiencia tanto para el usuario, chofer y/o administrador [4].

Es así que para mejorar el control y proporcionar una mejor experiencia a los estudiantes y al administrador, que hacen uso del servicio de transporte estudiantil de la Escuela Politécnica Nacional (PoliBus) se propone desarrollar un prototipo que realice: el monitoreo de las unidades de transporte, el control de acceso de los estudiantes, además, un control de asistencia para choferes de estas unidades.

Actualmente en el mercado ecuatoriano no existen dispositivos capaces de solventar estos requerimientos para este tipo de escenario. Con la ausencia de estos dispositivos el administrador de estas unidades de transporte continuará realizando estos procesos de forma manual.

1.1. OBJETIVOS

El objetivo general de este Proyecto Técnico es:

Desarrollar un prototipo para el monitoreo y control de acceso de las unidades de transporte estudiantil de la EPN mediante un aplicativo web y un aplicativo móvil.

Los objetivos específicos de este Proyecto Técnico son:

- Analizar los fundamentos teóricos de las funciones de cada módulo a integrarse en la Raspberry Pi 3 modelo B+, Firebase Realtime Database y de los diferentes lenguajes de programación.
- Diseñar el sistema de control, aplicativos web y móvil para Android del sistema.
- Implementar los sistemas de control y comunicaciones del prototipo.
- Analizar los resultados de las pruebas de funcionamiento del prototipo.

1.2. ALCANCE

El presente proyecto técnico consiste en desarrollar un prototipo para el monitoreo de las unidades de transporte estudiantil de la Escuela Politécnica Nacional (PoliBus) y control de acceso de los estudiantes a estas unidades mediante un aplicativo web y un aplicativo móvil para Android, esta propuesta consta de dos sistemas: el sistema de control y el sistema de comunicación.

El sistema de control estará compuesto por cuatro módulos: módulo GPS que contará con un dispositivo GPS Ublox NEO 6M, módulo de control electrónico para la interacción del chofer con el prototipo, módulo huella constituido por un sensor biométrico cuya finalidad será realizar el control de acceso de los estudiantes y el módulo cámara que permitirá capturar y grabar video sin audio, estas grabaciones serán almacenadas en la placa Raspberry Pi 3 Modelo B+ y podrán ser visualizados a través de la plataforma ZoneMinder. Cada uno de los módulos serán integrados a una placa Raspberry Pi 3 Modelo B+, adicionalmente dispondrá de un modem 3G para enviar cada uno de los datos generados por los módulos programados en Python, hacia una base de datos alojada en la nube de Google.

El sistema de comunicación estará conformado por la unidad remota que será monitoreada desde un aplicativo web y un aplicativo móvil para Android, un módulo registro que permitirá enrolar estudiantes al servicio de transporte a través del aplicativo web y este constará de un sensor biométrico que será integrado a una placa Raspberry Pi 3 modelo B+ y

finalmente el prototipo hará uso de la plataforma Firebase, principalmente de los servicios de: Firebase Realtime Database, donde se almacenará la base de datos, Firebase Authentication para la autenticación de los usuarios y Firebase Hosting para el alojamiento del aplicativo web.

El aplicativo web será dedicado para el perfil del administrador del servicio de transporte y el aplicativo móvil tendrá dos perfiles: estudiante, donde se podrá ver el estado actual de su unidad de transporte como asientos disponibles y ubicación de su unidad en el mapa y administrador, el cual también mostrará la información del perfil estudiante, además, se contará con video vigilancia y se visualizarán los registros de estudiantes y chofer.

1.3. MARCO TEÓRICO

Dentro de este capítulo se abordan los fundamentos teóricos necesarios para el desarrollo e implementación del trabajo de titulación.

1.3.1. PLACA RASPBERRY PI

Es un mini ordenador de bajo costo, tamaño reducido y gran capacidad de procesamiento; creado por Raspberry Pi Foundation, fundada por Eben Upton desde 2009 y está ubicada en Reino Unido [5], la principal intensión de la creación de esta placa fue enseñar informática a los niños de escuelas de países en desarrollo, desde el año 2012 está disponible para su compra ya que ha sido el motor para el desarrollo de varios proyectos.

Con el pasar de los años se han añadido mejoras en las diferentes versiones de la placa, tanto a nivel de hardware como de software, añadiéndole características como Bluetooth, wifi, puertos periféricos, aumento del número de pines de entrada y salida, cuenta con su propio sistema operativo de código abierto, soporta diferentes sistemas operativos GNU/Linux incluso versiones de Windows 10, por ello y debido a su bajo costo actualmente se ha usado en el desarrollo de proyectos de electrónica, robótica, investigación, IoT [1] entre otras.

En el mercado ecuatoriano actualmente se dispone de las placas Raspberry Pi Model B, Raspberry Pi 2 Model B y Raspberry Pi 3 Model B+. En la Tabla 1.1. se detallan las especificaciones técnicas de las placas mencionadas.

[1] IoT: (Internet of Things), se refiere a la interconexión de dispositivos u objetos a la red “internet” e interactuar con estos. [43]

Tabla 1.1. Especificaciones técnicas de las placas Raspberry Pi B, 2 B y 3 B+ [6]

	RASPBERRY PI		
Especificación	**Model B**	**2 Model B**	**3 Model B+**
SoC	Broadcom BCM2835	Broadcom BCM2836	Broadcom BCM2837
CPU	700 MHz ARM 1176JZF-S	900MHz ARM-Cortex-A7	1.4 GHz Cortex -A53
GPU	VideoCore IV, OpenGL ES 2.0, MPEG-2 y VC-1, 1080p30 H.264 / MPEG-4 AVC	VideoCore IV, OpenGL ES 2.0, MPEG-2 y VC-1, 1080p30 H.264 / MPEG-4 AVC	VideoCore IV, OpenGL ES 2.0, MPEG-2 y VC-1, 1080p30 H.264 / MPEG-4 AVC
RAM	512 MB	1 GB	1 GB
Consumo	700mA / 3.5W / 5V	800 mA / 4W / 5V	2.5A / 12.5 W / 5V
GPIO	8	17	17
USB 2.0	2 puertos	4 puertos	4 puertos
Ethernet	10 Mbps / 100 Mbps	10 Mbps / 100 Mbps	10 Mbps / 100 Mbps
Wi-Fi	No	No	2.4 GHz y 5GHz IEEE 802.11.b/g/n/ac.
Bluetooth	No	No	Bluetooth 4,2 BLE.
Video	RCA, HDMI	Jack, HDMI	Jack, HDMI
Audio	Jack, HDMI	Jack, HDMI	Jack, HDMI
CSI	Si	Si	Si
DSI	Si	Si	Si
Micro-SD	Si	Si	Si

1.1.1.1. SISTEMA OPERATIVO RASPBIAN

Para el desarrollo del prototipo se ha utilizado el sistema operativo oficial de las placas Raspberry Pi, llamado Raspbian que está basado en Debian GNU/Linux, libre y con licencia, este sistema operativo cuenta con más de 35.000 paquetes que permiten optimizar el rendimiento de la placa Raspberry Pi [7].

Este sistema cuenta con una página oficial que brinda soporte y ofrece actualizaciones gratuitas.

- **Python**

Python es un lenguaje de programación de código abierto, interpretado, multiplataforma, orientado a objetos, fácil de aprender y viene por defecto instalado en el sistema operativo Raspbian en sus dos versiones: Python 2 y Python 3.

Este lenguaje nos permite: importar fácilmente librerías, integrar módulos externos, codificar y controlar los pines de entrada/salida y puertos con la finalidad de obtener datos relevantes para luego ser enviados a la base de datos.

1.3.1.1 MÉTODOS DE CONTROL DE ACCESO

Con una Raspberry Pi se pueden implementar diferentes métodos de control acceso, los cuales son desarrollados y utilizados dependiendo de los requerimientos y circunstancias en las que se necesita realizar este control. Dentro de estos métodos se tiene a los más destacados:

- **Huella dactilar**

Este método es uno de los más eficientes debido a que mediante la huella dactilar, la cual es única e intransferible se tiene un muy buen control de acceso. Si se necesita implementar este control de acceso mediante la Raspberry Pi se debe también contar con un sensor biométrico para capturar la información de las huellas y para posteriormente validar su identidad.

- **Reconocimiento facial**

Mediante el reconocimiento fácil al igual que la huella dactilar son métodos infalibles por lo que se tiene una probabilidad muy baja de que si se implementa este control de acceso los usuarios puedan vulnerar esta seguridad. Para obtener un control de acceso de este tipo con una Raspberry Pi se necesita tener adicionalmente un cámara para capturar la imagen y posteriormente realizar el procesamiento de imagen adecuado.

- **Clave**

Este método puede ser el más vulnerable dentro de los anteriormente mencionados puesto que una clave puede ser transferible y mal usada de ser el caso. Para la implementación de este tipo de control de acceso se necesita adicionalmente de un teclado, con el cual se captura la información “personal” del usuario que intenta autenticarse.

- **Tarjetas RFID**

Al igual que el método con clave, el uso de tarjetas RFID es un control de acceso que también puede ser vulnerado si la tarjeta es extraviada o transferida a otra persona. Mediante un sensor y una tarjeta RFID la Raspberry Pi puede procesar esta información y determinar si la tarjeta está o no autenticada en el sistema para posteriormente brindar el acceso.

1.3.1.2 SISTEMAS DE GEOLOCALIZACIÓN

Para brindar un servicio de geolocalización con una Raspberry Pi se necesita de un módulo receptor GPS (Sistema de Posicionamiento Global), el cual mediante su antena recepta la

señal de al menos 4 satélites GPS que están a kilómetros de la atmósfera terrestre para posteriormente procesar esta información y mediante algoritmos matemáticos del propio módulo lograr determinar el retardo de esta señal y así dar la ubicación del dispositivo [8].

Para conseguir una ubicación más precisa se utiliza un sistema GPS asistido, el cual consiste en la implementación de segmentos de control, los cuales son estaciones en la tierra que reciben la señal satelital, teniendo así rapidez y mayor precisión. Este sistema es el más utilizado en los teléfonos celulares o dispositivos móviles, es por lo que estos dispositivos tienen una mejor señal de GPS que un sistema GPS por sí solo. Mediante la Figura 1.1 se puede observar el esquema de un sistema GPS asistido [8].

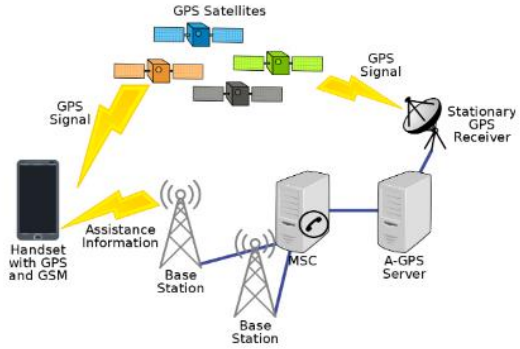


Figura 1.1. Esquema de sistema GPS asistido [8]

1.3.1.3 SISTEMAS DE VIDEOVIGILANCIA

Un sistema de videovigilancia permite visualizar el flujo de imágenes en directo o de eventos pasados a través de una cámara por un ordenador, teléfono celular o tablet. Estos sistemas se pueden implementar en una Raspberry Pi con los siguientes elementos:

1.3.1.3.1. Tipos de cámara

Actualmente en el mercado existen 2 tipos de cámaras de video que son compatibles con las Raspberry Pi, las cuales tiene como principal diferencia el tipo de conexión. Dentro de los tipos de conexión se tiene:

- **Cámara IP**

Este tipo de cámaras deben estar conectadas en la misma red que la Raspberry Pi, ya que mediante la red local se puede observar lo que estas cámaras capturan. Estas cámaras tienen alta resolución de imagen, es decir pueden capturar mayor cantidad de detalles presentes en el cuadro de imagen [9].

- **Cámara Raspberry (RaspiCam)**

RaspiCam es la cámara oficial que el proyecto Raspberry lanzó al mercado, la conectividad es mucho más sencilla y directa, debido a que la Raspberry tiene un puerto dedicado a este elemento [10].

1.3.1.3.2. Servidor de video

Un servidor de video permite procesar y visualizar las imágenes y/o video que una cámara está capturando para posteriormente almacenarla. Dentro de los servidores de video más estables en el mercado se tiene:

- **ZoneMinder**

ZoneMinder tiene un conjunto de herramientas que permiten configurar un sistema de videovigilancia, ya que: captura, graba, analiza y monitorea los eventos mediante las cámaras que están asociadas a este servidor. Una de las principales características de ZoneMinder es que es un producto gratuito y compatible exclusivamente con Linux, por lo que es muy usado en las placas Raspberry Pi [11].

ZoneMinder requiere de ciertos requisitos importantes:

- Base de datos MySQL o MaríaDB.
- Servidor Apache.
- Librerías PHP.
- Librerías libjpeg.

Este software está formado por varios componentes: archivos binarios encargados de realizar el procesamiento de video (ver Tabla 1.2), scripts PERL que al ser compilados interactúan con el sistema operativo, página web o con el usuario (ver Tabla 1.3) y finalmente scripts PHP que consisten en dar el estilo CSS a la interfaz web del ZoneMinder (ver Tabla 1.4) [11].

Tabla 1.2. Archivos binarios de ZoneMinder [12]

Nombre	Función
ZMC (Captura de ZoneMinder)	Permite monitorear un dispositivo y captura cuadros a la velocidad más rápida posible.
ZMA (Análisis de ZoneMinder)	Verifica si existen movimientos en los cuadros capturados para poder generar una alarma o evento.
ZMF (Frame de ZoneMinder)	Trabaja conjuntamente con el archivo ZMA para almacenar en la base de datos los cuadros grabados.
ZMS (Streaming Server de ZoneMinder)	Permite mostrar en la interfaz web el video en tiempo real o video almacenado.
ZMU (Utilidad de ZoneMinder)	Es una interfaz de línea de comandos para depurar problemas dentro del video y es utilizado por la página web.

Tabla 1.3. Scripts PERL de ZoneMinder [12]

Nombre	Función
zmpkg.pl.	Utilizado por la interfaz web y los diferentes scripts para controlar la ejecución del software.
zmdc.pl.	Utilizado por la interfaz web y el script zmpkg.pl para mantener la ejecución de ZMC y ZMA.
zmfilter.pl.	Controla los filtros y es iniciado/detenido desde la interfaz web.
zmaudit.pl.	Elimina eventos huérfanos y también verifica la coherencia de los datos generados por cada uno de los eventos.
zmwatch.pl.	monitorea a ZMC y lo reinicia en el caso de que este se detenga o deje de responder.
zmupdate.pl.	Busca y actualiza a todos los elementos del software.
zmvideo.pl.	Permite generar diferentes tipos de formatos de video desde la interfaz web.
zmcontrol.pl.	Traduce los comandos utilizados para controlar los dispositivos de video en un protocolo que puedan entender.
zm.	Detiene, inicia o reinicia ZoneMinder.

Tabla 1.4. Scripts PHP de ZoneMinder [12]

Nombre	Función
Classic	Estilo clásico
Flat	Estilo moderno

- **Blue Iris**

Blue Iris es un servidor de video que opera exclusivamente sobre sistemas operativos Windows, lo que lo hace incompatible con las placas Raspberry Pi. Además, es un software de paga, es decir necesita una licencia para ser activado en un ordenador. La característica que destaca sobre el resto de los servidores de video es que brinda la publicación de las cámaras configuradas en el sistema de videovigilancia hacia el internet por medio de servidores propietarios de Blue Iris [13].

1.3.1.4 LIBRERIAS RELACIONADAS

- **Pynmea2**

Esta es una librería elemental para los sistemas de geolocalización ya que todos los dispositivos que proporcionan coordenadas GPS, las obtienen en un formato común (formato NMEA) por lo que se necesita de esta librería, para a partir de esta información se realice un análisis a tal punto de obtener los valores como: latitud, longitud, altitud y entre otros [14].

- **PyFingerprint**

Es una librería para sistemas operativos Python, usada conjuntamente con los sensores biométricos en los sistemas de control de acceso. El proceso que realiza es la captura y

detección de los puntos más representativos de la huella, es decir, con un sensor biométrico y mediante esta librería se puede comparar y extraer las huellas que se necesiten para realizar sistemas de control de acceso [15].

1.3.2. BASES DE DATOS NOSQL

Son sistemas de almacenamiento de información que no cumplen con el esquema entidad-relación, no manejan una estructura en forma de tablas, sino que, para el almacenamiento hacen uso de otros tipos como clave-valor, mapeo de columnas o grafos y documentales [16].

Ventajas de las bases de datos NoSQL.

- Están basadas en clave-valor.
- Pueden manipular un gran conjunto de información.
- Su expansión es más fácil ya que se realiza un escalado horizontal y se distribuye la carga por todos los nodos [17].

1.3.2.1 TIPOS DE BASE DE DATOS

A continuación, se detallan los tipos de base de datos NoSQL más importantes:

- CLAVE-VALOR. - cada elemento tiene un identificador con una key única, permitiendo recuperar información de manera muy rápida. Estas bases son utilizadas en juegos, tecnología publicitaria e IoT [18].
- GRAFOS. - la información se almacena en estructuras grafo, cuyos elementos son: nodos que representan a las entidades, propiedades que hacen referencia a la información de entidades y líneas que serán las conexiones entre entidades. Son frecuentemente utilizadas en redes sociales, motores de recomendaciones, detección de fraude y gráficos de conocimiento [18].
- DOCUMENTOS. - Los datos se representan como objetos o en un documento JSON o XML y se basan en un modelo de clave-valor, este único documento almacena los datos relacionados con una clave específica [17].

1.3.3. PLATAFORMA FIREBASE

Esta plataforma antes llamada Envolve nació como una API[2] la cual permitía integrar servicios de chat dentro de páginas web, pero durante su manejo se dieron cuenta que los desarrolladores enviaban gran cantidad de información de partidas de juegos por ello decidieron separarla en dos funcionalidades, el sistema de chat y el sistema de arquitectura en tiempo real, dando lugar a la creación de Firebase, en el año 2012 esta plataforma fue comprada por Google y desde entonces se han añadido diferentes funcionalidades y servicios.

En la actualidad Firebase brinda la facilidad de desarrollar aplicaciones de alta calidad las cuales soportan elevadas cantidades de usuarios, los servicios que esta plataforma son los siguientes:

- Firebase Cloud Firestore.
- Firebase Authentication.
- Firebase Cloud Functions
- Firebase Realtime Database.
- Firebase Hosting.
- Firebase Cloud Storage.

Para el desarrollo del prototipo se optó trabajar con las siguientes herramientas que ofrece Firebase: Realtime Database, Authentication y Hosting.

Para llevar a cabo una transferencia de información mediante esta plataforma, el cliente y el servidor intercambian mensajes mediante el protocolo TLS[3], con el cual realiza una negociación del método de compresión y el tipo de cifrado con los que cada paquete será procesado en el envío y recepción de información entre el cliente y el servidor (ver Figura 1.2).

[2] API: (Application Programming Interface), es un conjunto de funciones, protocolos y comandos que permiten a los desarrolladores crear programas sin necesidad de comenzar desde cero [45].

[3] TLS: (Transport Layer Security), es la versión más actualizada de SSL, este protocolo establece una negociación entre cliente-servidor para así verificar su identidad y posteriormente establecer una comunicación segura [48].

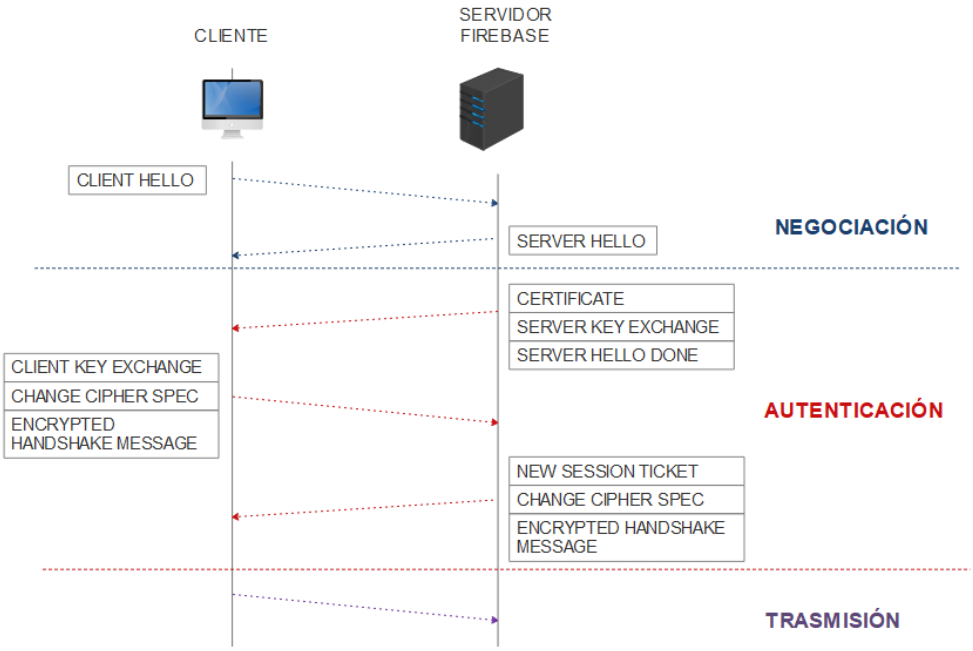


Figura 1.2. Comunicación Firebase

A continuación, se muestra una breve reseña de los pasos de una conexión entre el cliente y el servidor usando el protocolo TLS.

El cliente siempre es el que inicia la comunicación, por lo que envía un mensaje del tipo *Client Hello*, el cual realiza una oferta con una lista de tipos de cifrado y los métodos de compresión que éste soporta, además, de la versión del protocolo SSL[4] más alta permitida. Como respuesta a este primer mensaje el servidor es el encargado de escoger estos parámetros en el mensaje *Server Hello*.

Una vez realizadas las configuraciones para la comunicación entre cliente y servidor, estas entidades realizan un intercambio de mensajes: *Client Key Exchange - Cipher Spec - Encrypted Handshake Message* por parte del Cliente y *Certificate – Server Key Exchange – Server Hello Done* por parte del Servidor para con ello validar la conexión.

Si la autenticación con el servidor es correcta, este envía un mensaje *New Session Ticket -Change Cipher Spec – Encrypted Handshake Message* con el que da inicio a la transferencia de información entre el cliente y el servidor a través del protocolo HTTPS[5].

[4] SSL: (Secure Sockets Layer), permite cifrar el tráfico de datos en una comunicación, con lo que se garantiza que la información viaje segura a su destino [46].

[5] HTTPS: (Hyper Text Transfer Protocol Secure), este protocolo protege la información de un sitio web a través de certificados SSL o TLS [46].

1.3.3.1 Firebase Realtime Database

Es una base de datos en tiempo real NoSQL que se aloja en la nube, los datos son almacenados en formato JSON, se puede acceder, modificar y detectar cambios en los datos contenidos en cada nodo, permite añadir reglas para proteger los datos, es compatible con las plataformas de iOS, Android y JavaScript [19].

1.3.3.2 Firebase Authentication

Este servicio permite identificar la identidad de un usuario, así podrá interactuar con las aplicaciones desarrolladas de forma segura y poder leer y/o escribir datos en Firebase Realtime Database, permite la autenticación mediante contraseñas, números de teléfono, redes sociales como: Facebook, Google, Twitter entre otros [20].

1.3.3.3 Firebase Hosting

Es un servicio de hosting que permite alojar sitios web con contenido dinámico o estático en una CDN[6], de forma segura y así los usuarios pueden acceder rápidamente a su contenido.

Firebase trabaja juntamente con Firebase CLI, una herramienta que contiene una serie de comandos que permiten implementar archivos HTML y JavaScript desde una PC local a la CDN mediante una conexión SSL, permite definir varios sitios por proyecto, otorga dos subdominios gratuitos *web.app* y *firebaseapp.com,* en caso de no requerir estos subdominios se puede conectar el sitio a un propio dominio [21].

Firebase Hosting en el modo gratuito permite: 1GB de datos almacenados, 10GB/mes de datos transferidos, dominio personalizado y SSL, además, permite definir varios sitios por proyecto [22].

1.3.4. PLATAFORMA DATAPLICITY

Es una plataforma que provee soporte remoto para dispositivos compatibles con tecnología IoT mediante Dataplicity Wormhole, incluso si estos se desplazan entre redes celulares, satelitales y fijas, para lo cual estos dispositivos requieren principalmente una conexión a internet y tener una cuenta en esta plataforma. Es decir, mediante esta plataforma se puede publicar, administrar y acceder en la nube a cualquier servicio que se encuentre alojado en

[6] CDN: (Content Delivery Network), es una infraestructura que consiste en ubicar varios servidores en diferentes lugares geográficos que se comunican entre sí, con la finalidad de obtener una conectividad más rápida para el usuario [47].

un dispositivo, por ejemplo, un servidor web en una placa Raspberry Pi será visible y administrable de forma remota a través de la web. Cabe destacar que esta plataforma soporta a solo un dispositivo de forma gratuita, por lo que si se agregan más dispositivos se debe agregar una forma de pago en la cuenta. A continuación, en la Figura 1.3 se presenta el funcionamiento de Dataplicity [23].

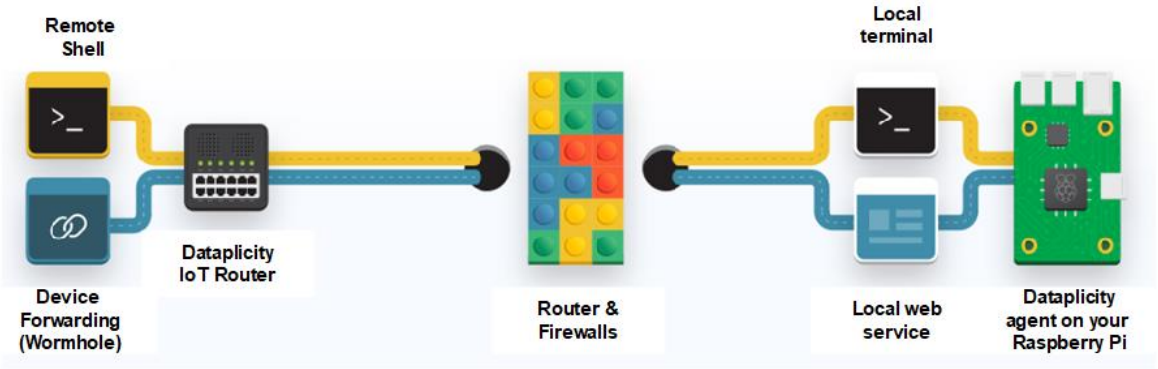


Figura 1.3. Funcionamiento de la plataforma Dataplicity [23]

1.3.5. HERRAMIENTAS PARA EL APLICATIVO WEB

1.3.5.1 Visual Studio Code

La herramienta seleccionada para el desarrollo del aplicativo web es el editor de código fuente Visual Studio Code, un editor multiplataforma, gratuito, ligero y potente creado por Microsoft [24], compatible con la mayoría de los lenguajes de programación ya que tan solo descargando las extensiones del lenguaje deseado ya podemos desarrollar el código, dentro de sus características se destacan:

- Permite autocompletar el código.
- Permite vincular con GitHub para almacenar y extraer repositorios.
- Permite realizar un programa de depurado.
- Permite instalar extensiones que facilitan la codificación en los diferentes lenguajes de programación (HTML, JavaScript, Angular, Python entre otras).

1.3.5.2 HiperText Markup Language HTML

Es un lenguaje que permite crear o modelar estructuras para sitios web y sirve para indicarle al navegador la forma en que debe presentar el sitio web, o a su vez indicar que elemento se tiene en una página siento este: una imagen, un párrafo, un video, entre otros, es compatible con diferentes navegadores y plataformas y actualmente se encuentra en su quinta versión llamada HTML5 [25].

1.3.5.3 JavaScript

Es un lenguaje de programación interpretado utilizado para el desarrollo y diseño de sitios web dinámicos que se ejecutan en el lado del cliente. Actualmente existen dos tipos de JavaScript: Navigator JavaScript del lado del cliente y LiveWire JavaScript para el lado del servidor.

Para incluir código JavaScript dentro de un documento HTML existen dos formas: la primera es agregar el código JavaScript dentro de la etiqueta <script> en la cabecera del documento (<head>) y la segunda forma es enlazando un documento con un archivo externo del tipo *nombre.js*, para esta forma en el documento se agrega la etiqueta <script> junto con dos atributos: *type* cuyo valor sería igual a *text/javascript* y *src* que corresponde al URL del archivo *nombre.js* [26].

Se eligió este lenguaje de programación para el desarrollo del aplicativo web debido a que existe una amplia guía dentro de la documentación de Firebase para permite integrar y configurar fácilmente cada uno de los servicios que ofrece Firebase, de esta forma el aplicativo web podrá leer/modificar datos dentro de la base de datos Firebase Realtime Database, acceder a las opciones de autenticación que ofrece Firebase Authentication y alojar el aplicativo web con la ayudad de Firebase Hosting.

1.3.5.4 Bootstrap

Es un framework CSS creado por Twitter en el año 2011, útil para crear interfaces de sitios web, cuenta con librerías CSS que permiten agregar estilos, botones, tablas, menús y otros elementos visuales a aplicaciones web adaptables a cualquier tipo de pantalla y dispositivo, es compatible con la mayor parte de navegadores web que existen actualmente [27].

1.3.6. HERRAMIENTAS PARA EL APLICATIVO MÓVIL

1.3.6.1 Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para sistemas operativos Android, usa el editor de códigos y las herramientas de IntelliJ. Además, incluye varias funcionalidades que ayudan a mejorar la productividad cuando se desarrollan aplicaciones, de las cuales cabe destacar las siguientes: [28]

- Un sistema de compilación basado en Gradle.
- Un emulador compatible con todas las versiones de Android.
- Integración con repositorios en GitHub y plantillas de código para compilar funciones que son comunes entre aplicaciones.

- Integración de un asistente realiza la configuración con todos los servicios de Firebase.
- Compatibilidad integrada para Google Cloud Plataform, la misma que facilita la integración con Google Cloud Messaging y App Engine [28].

1.3.6.2 Java

Java se creó en 1991 y se publicó en 1995 por Microsystem con el principal objetivo de ser un lenguaje de programación de estructura sencilla, el cual tendría la capacidad de ser ejecutado en diversos sistemas operativos. Este es un lenguaje de programación orientado a objetos con el cual se tienen características como: rapidez, seguridad y fiabilidad [29].

1.3.6.3 XML

XML es un metalenguaje, el cual es fácilmente procesable, libre de licencia, es estandarizado y es diseñado para cualquier lenguaje y alfabeto de programación. XML permite representar toda la información estructurada en la web, por lo que esta información puede ser visualizada en los diferentes aplicativos existentes [30].

1.3.7. OTRAS HERRAMIENTAS

1.3.7.1 Putty

Es una herramienta de conectividad que además de ser sencilla, gratuita e intuitiva, realiza conexiones SSH[7] o Telnet[8], ya sea con dispositivos que estén dentro de la red local o de una red remota, con el fin de configurar o monitorear estos dispositivos mediante un terminal. Para que esta conexión se realice correctamente se necesita instalar y/o configurar el servidor SSH o Telnet en el dispositivo al cual se desea acceder, el mismo que necesita estar encendido durante la conexión puesto que la arquitectura que utiliza es cliente-servidor.

1.3.8. METODOLOGÍA KANBAN

Es una metodología ágil que busca gestionar el flujo de procesos y tareas que se necesitan realizar en un determinado proyecto a través de la representación de tarjetas, teniendo así

[7] SSH: (Secure Shell), es un protocolo de administración remota que permite al usuario acceder a dispositivos en una red local o en la nube a través de un mecanismo de autenticación [50].

[8] Telnet: (Telecomunication Network), es un protocolo de administración remota que permite al usuario acceder a dispositivos en una red local o en la nube sin mecanismo de autenticación, lo que implica que toda la comunicación no es segura [49].

una metodología visual y fácil de entenderla e implementarla, puesto que las personas que participen o no en el proyecto pueden conocer el estado del mismo [31].

Para usar esta metodología es necesario conocer y manejar la herramienta conocida como tablero Kanban. Este tablero es una representación visual de todas las tareas, las cuales se dividen por lo general en 3 categorías: por hacer “To do”, en proceso “Doing” y realizadas “Done” (ver Figura 1.4). Con ello se tiene un orden de ejecución de cada proceso dentro del proyecto [32].

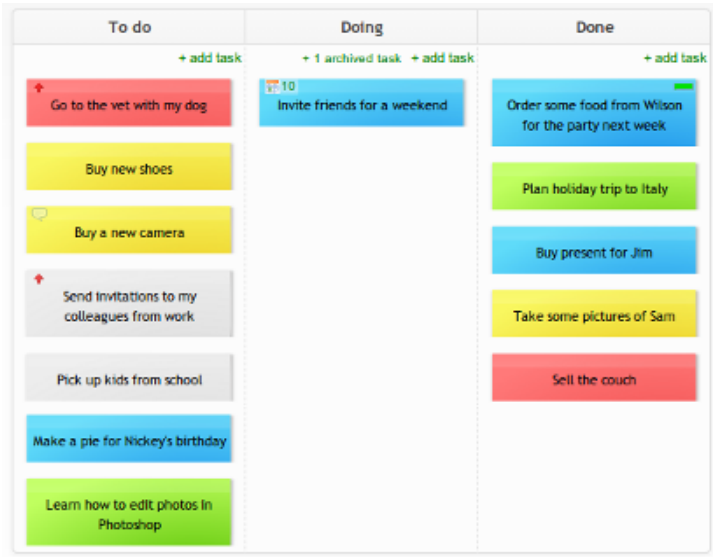


Figura 1.4. Representación visual del tablero Kanban [32]

La metodología Kanban se basa en los siguientes principios [31]:

- **Calidad garantizada**

Expresa la necesidad que las tareas se realicen bien en el primer intento, puesto que se utiliza más tiempo y recursos en arreglar algo que no cumple con los requerimientos planteados que en hacerlo bien a la primera. Por lo tanto, bajo este principio en la metodología Kanban no importa lo rápido que se realicen las tareas, sino que se realicen bien.

- **Reducción del desperdicio**

Este principio trata de que no es necesario hacer algo extra para que la tarea se realice correctamente, sino solo lo justo y necesario, con ello se garantiza que esta metodología tiene una optimización de recursos.

- **Mejora continua**

Se enfoca en mejorar los procesos pendientes con un sistema de mejora continua, teniendo en cuenta el cómo se realizaron las tareas anteriores.

- **Flexibilidad**

Brinda una alta capacidad de respuesta ante circunstancias o tareas que no se tomaron en consideración en la planificación, con lo que se tiene una cola de espera, en la cual debe existir prioridades para ir realizando las tareas dentro de esta.

2. METODOLOGÍA

En el presente capítulo se detallan las etapas que se utilizaron para el diseño e implementación del prototipo, se detallará las tareas en el tablero Kanban, la recopilación de los requerimientos obtenidos de los usuarios que hacen uso del servicio de transporte de la EPN, el diseño de los sistemas junto con los módulos definidos y finalmente se describe la implementación de estos.

2.1. TABLERO KANBAN

En la figura 2.1 se aprecia el tablero Kanban con la lista de tareas a realizarse para el diseño del prototipo el mismo que permitirá mejorar el flujo de trabajo.

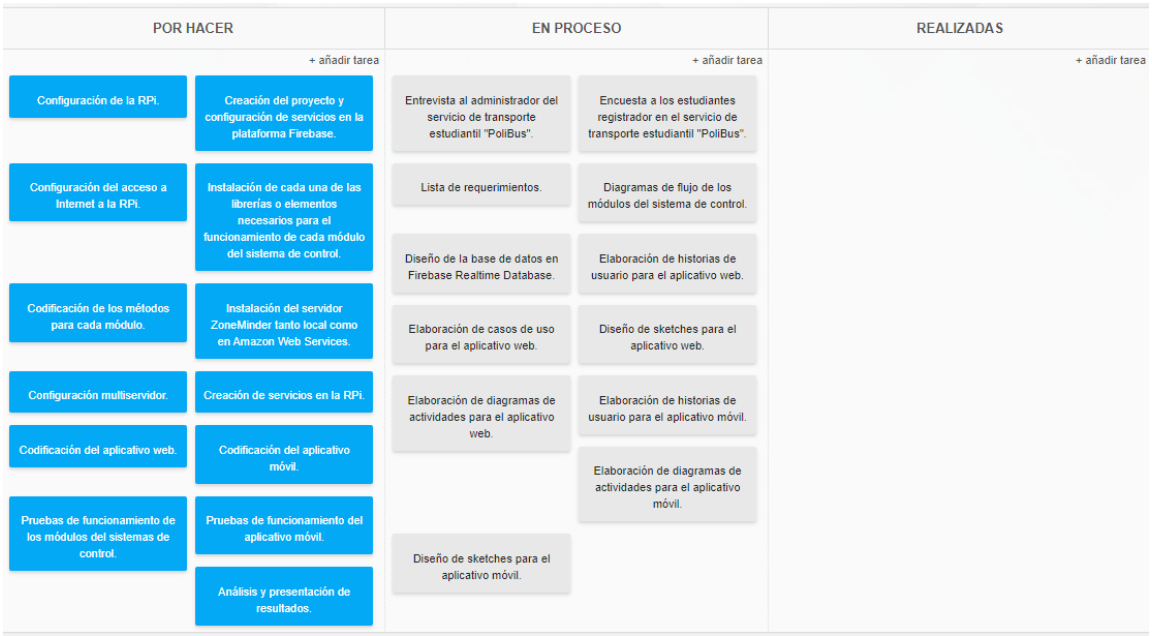


Figura 2.1. Tablero Kanban de la fase de diseño

2.2. REQUERIMIENTOS DEL PROTOTIPO

Para el diseño y desarrollo del prototipo se obtuvieron los requerimientos de las personas involucradas en el servicio de transporte estudiantil de la EPN, es decir: estudiantes y administrador, así como también se han adoptado ciertas características de los aplicativos que se encuentran en el mercado internacional tal es el caso de NUBE BUS [33] y SCHOOL BUS TRACKER [34].

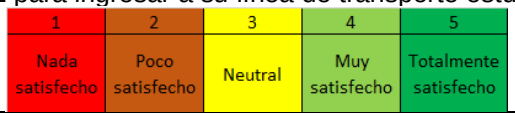
2.2.1. ENTREVISTA AL ADMINISTRADOR

Se realizó una entrevista al Ph.D. Fernando Carrera, administrador del servicio de transporte estudiantil “PoliBus”, donde se obtuvo una lista de requerimientos que el administrador deseaba integrar con el fin de poder realizar un control de acceso de los estudiantes y choferes, así como monitorear cada unidad de transporte, esta lista detallada podrá ser encontrada en el ANEXO A.

2.2.2. ENCUESTA A LOS ESTUDIANTES

Se realizaron 40 encuestas que contienen 10 preguntas para los estudiantes que usan el servicio de transporte estudiantil “PoliBus” con estas encuestas se justificarán los requerimientos establecidos en el trabajo de titulación. La encuesta y los resultados las mismas se encuentran en el ANEXO B. A continuación, se resumen los resultados de las preguntas elaboradas en la Tabla 2.1.

Tabla 2.1. Resultados de la Encuesta dirigida a los estudiantes

No.	PREGUNTA	RESPUESTA (%)
1.	¿Dispone de un teléfono inteligente?	
	Si	97.5
	No	2.5
2.	¿Qué sistema operativo tiene su teléfono inteligente?	
	Android	92.3
	IOS	7.7
3.	¿Cree usted que el servicio de transporte estudiantil PoliBus está siendo usado por personas que NO pertenecen a la EPN?	
	Sí	92.5
	No	7.5
4.	En la escala del 1 al 5, ¿Qué tan satisfecho está usted con el registro de firmas, que se realiza ACTUALMENTE para ingresar a su línea de transporte estudiantil PoliBus? 	
	1	7.5
	2	67.5
	3	17.5
	4	7.5
	5	0
5.	¿Cree usted que se debería hacer un control de acceso para el servicio de transporte estudiantil PoliBus?	
	Sí	97.5
	No	2.5
6.	¿Cuál cree usted que sería el mejor método de acceso para las unidades de transporte estudiantil PoliBus?	
	Huella dactilar	76.9
	Cédula de identidad	2.6
	Código QR	5.1
	Carnet estudiantil	15.4

7.	¿Usted alguna vez ha perdido (se ha atrasado) la unidad de transporte estudiantil PoliBus debido a que no conoce la ubicación de la unidad?	
	Sí	90
	No	10
8.	¿Le gustaría dar seguimiento a la línea de transporte estudiantil PoliBus que usa?	
	Sí	90
	No	10
9.	¿Le gustaría que la información del seguimiento a su línea de transporte estudiantil PoliBus sea presentada en una aplicación móvil?	
	Sí	100
	No	0
10.	¿Qué información le gustaría que se muestre en la aplicación móvil?	
	Ubicación actual del transporte.	94.4
	Número de asientos disponibles.	97.2
	Nombre del chofer.	19.4
	Hora en la que se inició la ruta.	58.3

2.2.3. LISTA DE REQUERIMIENTOS

La lista de requerimientos se definió basándose en la entrevista y resultados de las encuestas elaboradas a los usuarios del servicio de transporte “PoliBus”. Después de analizar los requerimientos se ha decidido separarlos en dos tipos:

- Requerimientos Funcionales (RF).
- Requerimientos No Funcionales (RNF).

2.2.3.1 Requerimientos Funcionales

Debido a que el prototipo cuenta con hardware y software se ha decidido diferenciarlos los requerimientos funcionales por: Hardware (RFH) y Software (RFS).

2.2.3.1.1. Hardware

En la Tabla 2.2 se visualiza los requerimientos funcionales definidos para el hardware del prototipo.

Tabla 2.2. Requerimientos Funcionales Hardware

Código	Descripción	Fuente
RFH01	Control de acceso a la unidad de transporte	Encuesta/Entrevista
RFH02	Conocer la ubicación de la unidad de transporte	Encuesta/Entrevista
RFH03	Monitorear de forma remota y respaldar los eventos que sucedan en la unidad de transporte.	Entrevista
RFH04	Permitir iniciar, finalizar o alertar una emergencia en una ruta especifica.	Entrevista

2.2.3.1.2. Software

Debido a que el prototipo consta de dos aplicativos, a continuación, se presenta los requerimientos funcionales para: aplicativo web (RFSW) y aplicativo móvil (RFSM).

A continuación, en la Tabla 2.3 se presenta el formato de las historias de usuario obtenidas para cada aplicativo. Cada historia de usuario tiene los siguientes elementos:

- **ID:** Es el identificador único para cada historia de usuario, el mismo que consta las siglas “HU”, como el prototipo tiene dos aplicativos se las puede diferenciar con las siglas “HUW” para las historias de usuario del aplicativo web y “HUM” para las historias de usuario del aplicativo móvil, seguidamente tienen un número. Por ejemplo: el ID “HUW01” representa la historia de usuario del aplicativo web número 1.
- **Rol:** Indica a que usuario está orientada la historia de usuario. Este elemento puede ser Administrador, Estudiante, Administrador/Estudiante o Ninguno.
- **Prioridad:** Este elemento indica el nivel de importancia que tiene la historia de usuario para el desarrollo del prototipo, por lo que se definen 3 tipo de prioridades: baja, media y alta.
- **Nombre de la historia:** Indica una breve descripción de la historia de usuario.
- **Descripción:** En este elemento se describe la funcionalidad de la historia de usuario.
- **Criterios de aceptación:** Indica en resumen la funcionalidad la historia de usuario y es considerado para realizar las pruebas necesarias de dicha historia.

Tabla 2.3. Formato de Historia de Usuario

HISTORIAS DE USUARIO					
ID:		**Rol:**		**Prioridad:**	
Nombre de la historia:					
Descripción:					
Criterios de aceptación:					

Aplicativo web

Basándose en la entrevista, se elaboraron las historias de usuario cumpliendo cada una de las funcionalidades que debe tener el aplicativo web. A continuación, se presentan cada una de las historias de usuario:

Tabla 2.4. Historia de usuario del aplicativo web 01

HISTORIAS DE USUARIO					
I**D:**	HUW01	**Rol:**	Administrador	**Prioridad:**	Alta
Nombre de la historia: Iniciar sesión.					
Descripción: El administrador podrá iniciar sesión siempre y cuando este registrado, mediante el ingreso de una dirección de correo electrónico y contraseña.					
Criterios de aceptación: Permite iniciar sesión y acceder al aplicativo web.					

Tabla 2.5. Historia de usuario del aplicativo web 02

HISTORIAS DE USUARIO					
I**D:**	HUW02	**Rol:**	Administrador	**Prioridad:**	Baja
Nombre de la historia: Recuperación de contraseña.					
Descripción: El administrador que este registrado y olvide su contraseña podrá restablecer la contraseña ingresando la dirección de correo electrónico.					
Criterios de aceptación: El aplicativo permite al administrador ingresar con una nueva contraseña.					

Tabla 2.6. Historia de usuario del aplicativo web 03

HISTORIAS DE USUARIO					
I**D:**	HUW03	**Rol:**	Administrador	**Prioridad:**	Alta
Nombre de la historia: Visualización de la página principal.					
Descripción: El administrador luego de iniciar sesión podrá: 1. Visualizar el formulario principal y también tendrá acceso a los diferentes formularios del aplicativo. 2. Seleccionar la línea de la unidad de transporte y seguidamente se mostrará la información actual de la unidad de transporte en el caso de que se esté realizando una ruta.					
Criterios de aceptación: El aplicativo permite visualizar los diferentes formularios.					

Tabla 2.7. Historia de usuario del aplicativo web 04

HISTORIAS DE USUARIO					
I**D:**	HUW04	**Rol:**	Administrador	**Prioridad:**	Baja
Nombre de la historia: Registro de asistencia chofer.					
Descripción: El administrador podrá generar por semana un registro de asistencia de los choferes, donde F significa “falta” y A “asistencia”. También se podrá descargar los registros en formato PDF.					
Criterios de aceptación: El aplicativo permite al administrador generar los registros semanales de asistencia de choferes.					

Tabla 2.8. Historia de usuario del aplicativo web 05

HISTORIAS DE USUARIO					
I**D:**	HUW05	**Rol:**	Administrador	**Prioridad:**	Baja
Nombre de la historia: Registro de asistencia estudiante.					
Descripción: El administrador podrá generar por semana o diario un registro de asistencia de los estudiantes. También se podrá descargar los registros en formato PDF. Diario: El registro generado contendrá la siguiente información: dirección de subida en la mañana y dirección de bajada en la noche del estudiante, además, de sus datos personales. Semanal: El registro generado contendrá a todos los estudiantes que utilizan una unidad de transporte en particular donde por día ya sea mañana o noche se indicara con las letras A de “asistencia” y F de “falta”.					
Criterios de aceptación: El aplicativo permite al administrador generar los registros semanales o diarios de asistencia para los estudiantes.					

Tabla 2.9. Historia de usuario del aplicativo web 06

HISTORIAS DE USUARIO					
I**D:**	HUW06	**Rol:**	Administrador	**Prioridad:**	Alta
Nombre de la historia: Administración de usuarios.					
Descripción: El administrador podrá: 1. Ver lista de usuarios registrados. 2. Crear nuevos usuarios. 3. Actualizar la información de los usuarios. 4. Eliminar usuarios.					
Criterios de aceptación: El aplicativo permite la gestión de los usuarios.					

Tabla 2.10. Historia de usuario del aplicativo web 07

HISTORIAS DE USUARIO					
I**D:**	HUW07	**Rol:**	Administrador	**Prioridad:**	Alta
Nombre de la historia: Administración de buses.					
Descripción: El administrador podrá: 1. Ver lista de unidades de transporte registradas. 2. Crear nuevas unidades de transporte. 3. Actualizar la información de las unidades de transporte. 4. Eliminar unidades de transporte.					
Criterios de aceptación: El aplicativo permite la gestión de las unidades de transporte.					

Tabla 2.11. Historia de usuario del aplicativo web 08

HISTORIAS DE USUARIO					
I**D:**	HUW08	**Rol:**	Administrador	**Prioridad:**	Alta
Nombre de la historia: Administración de carreras.					
Descripción: El administrador podrá: 1. Ver un listado de carreras registradas. 2. Crear nuevas carreras. 3. Actualizar la descripción de la carrera. 4. Eliminar carreras.					
Criterios de aceptación: El aplicativo permite la gestión de las carreras.					

Tabla 2.12. Historia de usuario del aplicativo web 09

HISTORIAS DE USUARIO					
I**D:**	HUW09	**Rol:**	Administrador	**Prioridad:**	Alta
Nombre de la historia: Ingreso a ZoneMinder.					
Descripción: El administrador podrá acceder al software ZoneMinder y también: • Visualizar las cámaras de cada una de las unidades de transporte. • Agregar nuevas cámaras.					
Criterios de aceptación: El aplicativo permite el ingreso a ZoneMinder.					

Tabla 2.13. Historia de usuario del aplicativo web 10

HISTORIAS DE USUARIO					
I**D:**	HUW10	**Rol:**	Administrador	**Prioridad:**	Alta
Nombre de la historia: Cerrar sesión.					
Descripción: El administrador podrá cerrar su sesión en el aplicativo.					
Criterios de aceptación: El aplicativo permite cerrar sesión.					

En la Tabla 2.14 se muestran la lista de requerimientos funcionales software para el aplicativo web.

Tabla 2.14. Requerimientos Funcionales Software del aplicativo web

Historia de Usuario	**Código**	**Descripción**
HUW01	RFSW01	Iniciar sesión.
HUW02	RFSW02	Recuperar contraseña.
HUW03	RFSW03	Visualizar información actual de la unidad de transporte seleccionada.
HUW04	RFSW04	Obtener registros de asistencia del chofer.
HUW05	RFSW05	Obtener registros de asistencia del estudiante.
HUW06	RFSW06	Administrar usuarios.
HUW07	RFSW07	Administrar unidades de transporte.
HUW08	RFSW08	Administrar carreras.
HUW09	RFSW09	Ingresar a ZoneMinder.
HUW10	RFSW10	Cerrar sesión.

Aplicativo Móvil

A continuación, se observan las historias de usuario que cumplen con cada una de las funcionalidades que debe tener el aplicativo móvil.

Tabla 2.15. Historia de usuario del aplicativo móvil 01

HISTORIAS DE USUARIO					
I**D:**	HUM01	**Rol:**	Administrador o estudiante	**Prioridad:**	Alta
Nombre de la historia: Iniciar sesión.					
Descripción: El administrador o estudiante podrá iniciar sesión (siempre y cuando este registrado) mediante el ingreso de una dirección de correo electrónico y contraseña.					
Criterios de aceptación: Permite iniciar sesión y acceder al aplicativo móvil.					

Tabla 2.16. Historia de usuario del aplicativo móvil 02

HISTORIAS DE USUARIO					
I**D:**	HUM02	**Rol:**	Administrador o estudiante	**Prioridad:**	Baja
Nombre de la historia: Recuperación de contraseña.					
Descripción: El administrador o estudiante podrá recuperar la contraseña ingresando la dirección de correo electrónico.					
Criterios de aceptación: El aplicativo permite al administrador o estudiante cambiar su contraseña actual.					

Tabla 2.17. Historia de usuario del aplicativo móvil 03

HISTORIAS DE USUARIO					
I**D:**	HUM03	**Rol:**	Administrador o estudiante	**Prioridad:**	Alta
Nombre de la historia: Visualización de la pantalla de inicio.					
Descripción: El administrador o estudiante al abrir el aplicativo podrá visualizar la pantalla inicio, mientras válida el inicio de sesión y posteriormente abrirá la pantalla información o control dependiendo del rol del usuario.					
Criterios de aceptación: El aplicativo válida el inicio de sesión y dirige al usuario a la pantalla correspondiente.					

Tabla 2.18. Historia de usuario del aplicativo móvil 04

HISTORIAS DE USUARIO					
I**D:**	HUM04	**Rol:**	Estudiante	**Prioridad:**	Alta
Nombre de la historia: Visualización de información.					
Descripción: El estudiante podrá visualizar la pantalla información, la cual mostrará la información actual de la unidad de transporte en la que está registrado, siempre y cuando la unidad esté realizando una ruta.					
Criterios de aceptación: El aplicativo permite la visualización de la pantalla información.					

Tabla 2.19. Historia de usuario del aplicativo móvil 05

HISTORIAS DE USUARIO					
I**D:**	HUM05	**Rol:**	Administrador	**Prioridad:**	Alta
Nombre de la historia: Visualización de control.					
Descripción: El administrador podrá visualizar la pantalla control en la que se le permitirá seleccionar una unidad de transporte para obtener la información actual de la unidad de transporte en el caso de que se esté realizando una ruta.					
Criterios de aceptación: El aplicativo permite la visualización de la pantalla control.					

Tabla 2.20. Historia de usuario del aplicativo móvil 06

HISTORIAS DE USUARIO					
I**D:**	HUM06	**Rol:**	Administrador	**Prioridad:**	Alta
Nombre de la historia: Ingreso a ZoneMinder.					
Descripción: El administrador podrá navegar en el sitio ZoneMinder AWS y visualizar las cámaras añadidas correspondientes a cada unidad de transporte.					
Criterios de aceptación: El aplicativo permite la navegación en el sitio ZoneMinder AWS.					

Tabla 2.21. Historia de usuario del aplicativo móvil 07

HISTORIAS DE USUARIO					
I**D:**	HU07	**Rol:**	Administrador	**Prioridad:**	Alta
Nombre de la historia: Registro actual de estudiantes.					
Descripción: El administrador podrá visualizar el registro de los estudiantes que están presentes en cada ruta.					
Criterios de aceptación: El aplicativo permite la visualización del registro actual en cada unidad de transporte.					

Tabla 2.22. Historia de usuario del aplicativo móvil 08

HISTORIAS DE USUARIO					
I**D:**	HUM08	**Rol:**	Administrador o estudiante	**Prioridad:**	Alta
Nombre de la historia: Visualización de emergencias.					
Descripción: El administrador podrá ver y/o eliminar las emergencias que ocurrieron en la unidad de transporte seleccionada, para el caso del estudiante cuando reciba una notificación en esta pantalla podrá ver la información de la última emergencia ocurrida en la unidad suscrita.					
Criterios de aceptación: El aplicativo permite ver las emergencias que ocurrieron la unidad de transporte.					

En la Tabla 2.23 se muestran la lista de requerimientos funcionales software para el aplicativo móvil.

Tabla 2.23. Requerimientos Funcionales Software del aplicativo móvil

Historia de Usuario	**Código**	**Descripción de requerimiento**
HUM01	RFSM01	Iniciar sesión.
HUM02	RFSM02	Recuperar contraseña.
HUM03	RFSM03	Validar rol.
HUM04	RFSM04	Visualizar información de la unidad de transporte.
HUM05	RFSM05	Visualizar información de las unidades de transporte.
HUM06	RFSM06	Ingresar a ZoneMinder.
HUM07	RFSM07	Visualizar registro actual de estudiantes.
HUM08	RFSM08	Visualizar notificaciones de emergencia.

2.2.3.2 Requerimientos No Funcionales

A continuación, se muestran las historias de usuario obtenidas para los Requerimientos No Funcionales.

Tabla 2.24. Historia de usuario 01

HISTORIAS DE USUARIO					
I**D:**	HU01	**Rol:**	Ninguno	**Prioridad:**	Alta
Nombre de la historia: Disponibilidad a conectarse a Internet.					
Descripción: El prototipo deberá conectarse a Internet para guardar y obtener la información en la base.					
Criterios de aceptación: El prototipo tiene conexión a Internet.					

Tabla 2.25. Historia de usuario 02

HISTORIAS DE USUARIO					
I**D:**	HU02	**Rol:**	Ninguno	**Prioridad:**	Alta
Nombre de la historia: Escalabilidad.					
Descripción: El prototipo deberá ser implementado en una plataforma que permita almacenar toda la información generada por el mismo en el transcurso del tiempo.					
Criterios de aceptación: El prototipo permitirá el almacenamiento de toda la información generada.					

Tabla 2.26. Historia de usuario 03

HISTORIAS DE USUARIO					
ID:	HU03	**Rol:**	Administrador o estudiante	**Prioridad:**	Alta
Nombre de la historia: Acceso a los aplicativos.					
Descripción: Los usuarios tendrán acceso a los aplicativos mediante correo electrónico y contraseña.					
Criterios de aceptación: Solo el usuario registrado podrá acceder y usar los aplicativos.					

Tabla 2.27. Historia de usuario 04

HISTORIAS DE USUARIO					
ID:	HU04	**Rol:**	Ninguno	**Prioridad:**	Alta
Nombre de la historia: Dispositivo.					
Descripción: El prototipo deberá tener un dispositivo capaz de obtener información generada por ciertos periféricos para poder procesarla y guardarla.					
Criterios de aceptación: El dispositivo procesa información.					

Tabla 2.28. Historia de usuario 05

HISTORIAS DE USUARIO					
ID:	HU05	**Rol:**	Ninguno	**Prioridad:**	Alta
Nombre de la historia: Comercialización.					
Descripción: Los dispositivos que conformen el prototipo deberán ser de bajo costo y encontrados en el mercado ecuatoriano.					
Criterios de aceptación: Prototipo a un bajo costo.					

En la Tabla 2.29 se observa los requerimientos no funcionales definidos para el prototipo.

Tabla 2.29. Requerimientos no funcionales

Historia de Usuario	Código	Descripción de requerimiento
HU01	RNF01	Conexión a Internet.
HU02	RNF02	Escalabilidad.
HU03	RNF03	Acceso a los aplicativos.
HU04	RNF04	Dispositivo.
HU05	RNF05	Comercialización.

2.3. ESTRUCTURA DEL PROTOTIPO

El diseño de la estructura del prototipo está conformado por dos sistemas: sistema de control y sistema de comunicación. El primer sistema corresponde al hardware del prototipo y está compuesto por los siguientes módulos:

- Módulo GPS.
- Módulo huella.
- Módulo de control electrónico.
- Módulo cámara.

El objetivo de los módulos anteriormente mencionados es obtener información a través de los diferentes elementos integrados a ellos, para luego ser procesados por la RPi.

Finalmente, el sistema de comunicación tiene:

- Un aplicativo web con un único perfil dedicado al administrador del servicio de transporte estudiantil, este aplicativo presenta los datos obtenidos del sistema de control. Adicionalmente constará de un módulo registro, el cual permitirá enrolar a los estudiantes mediante su huella dactilar.

- Un aplicativo móvil con dos perfiles: administrador y estudiante que de igual manera presenta los valores obtenidos por el sistema de control.
- Y los servicios que integra la base de datos, hosting y autenticación.

En la Figura 2.2 se puede apreciar el diagrama del prototipo junto con los sistemas y módulos que conforman la estructura del prototipo.

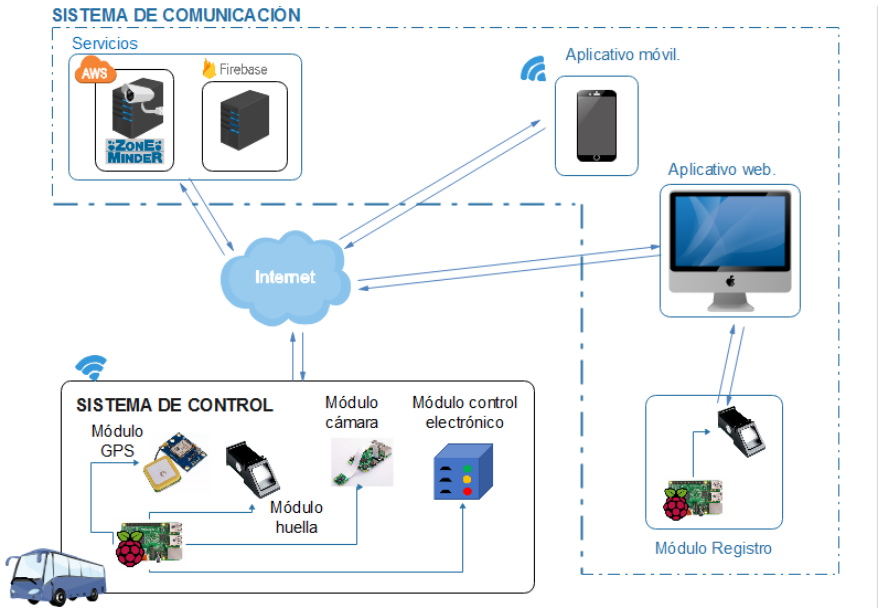


Figura 2.2. Diagrama del prototipo

2.4. SELECCION DE DISPOSITIVOS PARA EL SISTEMA DE CONTROL

En esta sección se indica los diferentes dispositivos elegidos para cada módulo correspondiente al sistema de control.

2.4.1. RASPBERRY PI 3 B+

Debido al requerimiento RNF04 de la Tabla 2.29 para el desarrollo del prototipo se eligió la Raspberry Pi 3B+ debido a que es la última versión existente en el mercado, también por el mayor número de puertos/pines ya que se integrarán varios elementos electrónicos (sensores, leds, pulsadores entre otros) a la RPi, se instalará y configurará la versión más reciente de sistema operativo llamado Stretch.

2.4.2. MÓDULO GPS

Debido al requerimiento RFH02 de la Tabla 2.2 se integró este módulo que será el encargado de obtener los valores de coordenadas de latitud y longitud, está conformado

por un dispositivo GPS que será controlado por la RPi y trabajará conjuntamente con la librería pynmea2, los valores obtenidos serán enviados a una base de datos para luego ser mostrados en los diferentes aplicativos.

En la actualidad existen innumerables dispositivos GPS, pero no todos son compatibles con la placa RPi 3 B+, varios de estos dispositivos ofrecen adicionalmente otros servicios (GPRS y GSM), dependiendo de los servicios que posea el costo del dispositivo es mayor. Por tal razón debido al requerimiento RNF05 de la Tabla 2.29 para el prototipo sea ha seleccionado un dispositivo que solo cuente con el servicio de GPS y de bajo costo. A continuación, se presenta un resumen de dispositivo GPS seleccionado.

2.4.2.1 GPS Ublox NEO 6M

Este dispositivo está formando por un EEPROM[9] que permite guardar configuración de parámetros, además, contiene al circuito NEO-6M y una antena cerámica como se puede observar en la Figura 2.3, este dispositivo es compatible con la librería pynmea2.

Figura 2.3. Ublox NEO 6M [35]

El dispositivo cuenta con 4 pines, en la Tabla 2.30 se describe cada uno de ellos.

Tabla 2.30. Pines del sensor GPS Ublox NEO 6M

Pines	Descripción
Vcc	5V
Rx	Recepción
Tx	Transmisión
GND	Tierra

2.4.2.2 Diagrama de flujo del programa en la RPi para el módulo GPS

A continuación, en la Figura 2.4 se presenta el diagrama de flujo del programa en la RPi para el módulo GPS, el cual inicia obteniendo las coordenadas NMEA, las cuales se evalúan si son válidas, luego a partir de las coordenadas NMEA se puedan obtener la

[9] EEPROM (Electrically Erasable Programmable Read-Only Memory): es un chip de memoria no volátil que puede borrarse desde un computador o externamente. [44]

latitud y longitud de la ubicación de la unidad remota para posteriormente guardar esta información en la base de datos.

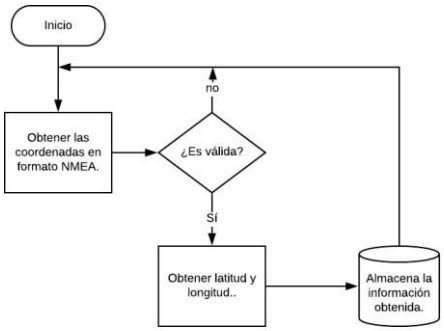


Figura 2.4. Diagrama de flujo del programa en la RPi para el módulo GPS

2.4.3. MÓDULO HUELLA

Para cumplir el requerimiento RFH01 de la Tabla 2.2, se agregó este módulo con la finalidad de realizar un control de acceso a los estudiantes, está conformado por un sensor biométrico compatible con RPi 3B+. Este módulo se encargará de escanear las huellas dactilares y posteriormente enviarlas a una base de datos a través de la RPi, con la información y otros datos adicionales se podrán obtener los registros de asistencia, además, se conocerá que personas se encuentran en la ruta actual.

En cuanto a la selección del dispositivo que conformará este módulo se debe recalcar que no existe variedad debido a la irregularidad de stock de sensores biométricos en el mercado ecuatoriano, ya que depende de la importación que se realiza cada cierto tiempo. Por lo que solo se encontró el sensor R305 AS601. A continuación, se presenta un breve resumen de dispositivo.

2.4.3.1 Sensor biométrico R305 AS601

La función de este sensor óptico es escanear huellas dactilares, utiliza el chip AS601 de alta potencia que se encarga de interpretar, ejecutar el cálculo de la imagen escaneada que permitirá realizar la identificación, detección y además buscar huellas. Cada huella escaneada genera una plantilla diferente que se almacena dentro de una memoria interna, que puede almacenar hasta 1000 plantillas o huellas. En la Figura 2.5 se presenta el sensor.

Figura 2.5. Sensor Biométrico AS601 [36]

Este sensor posee 4 pines, los cuales se detallan en la Tabla 2.31.

Tabla 2.31. Pines del Sensor FPM10A

Pines	Descripción
Tx	Trasmisión
Rx	Recepción
Vcc	5 V
GND	Tierra

2.4.3.2 Conexión del módulo huella a la RPi

Mediante el software de simulación Proteus[10] y con los requerimientos previamente obtenidos, se realizó la conexión para el módulo huella, el cual se presenta a continuación, en la Figura 2.6.

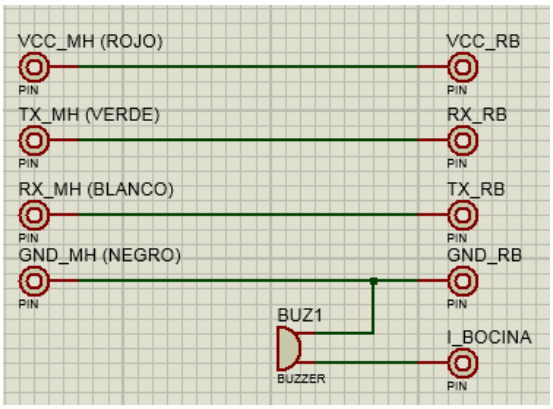


Figura 2.6. Conexión del módulo huella a la RPi

A continuación, en la Tabla 2.31 se presenta la función de cada elemento que forma parte del módulo huella.

[10] **Proteus:** software de diseño electrónico.

Tabla 2.32. Funciones de los elementos del circuito del módulo huella

Elemento	Nomenclatura	Función
PIN	GND_MH (NEGRO)	Polariza el módulo huella.
	VCC_MH (ROJO)	
	TX_MH (VERDE)	Comunica (transmisión) del módulo huella.
	RX_MH (BLANCO)	Comunica (recepción) del módulo huella.
	VCC_RB	Polariza el módulo huella desde la RPi.
	GND_RB	
	TX_RB	Comunicación (transmisión) que ofrece la RPi
	RX_RB	Comunicación (recepción) que ofrece la RPi.
	I_BOCINA	Conexión desde la RPi, para encender la bocina.
BOCINA	BUZ1	Indica si la RPi ha comprobado la huella colocada está registrada o no.

2.4.3.3 Diagrama de flujo del programa en la RPi para el módulo huella

A continuación, en la Figura 2.7 se presenta el diagrama de flujo del programa en la RPi para el módulo huella, el cual inicia con el encendido del sensor biométrico, luego se evalúa si un usuario está colocando su huella en el sensor, posteriormente se comprueba si existe una ruta activa, de ser así el caso se captura la huella y se verifica en la base de datos si este usuario esta registrado y si es estudiante o chofer, para con ello registrar el control de asistencia del chofer en la base de datos y en el caso de estudiante registrar si se subió o bajo de la unidad de transporte.

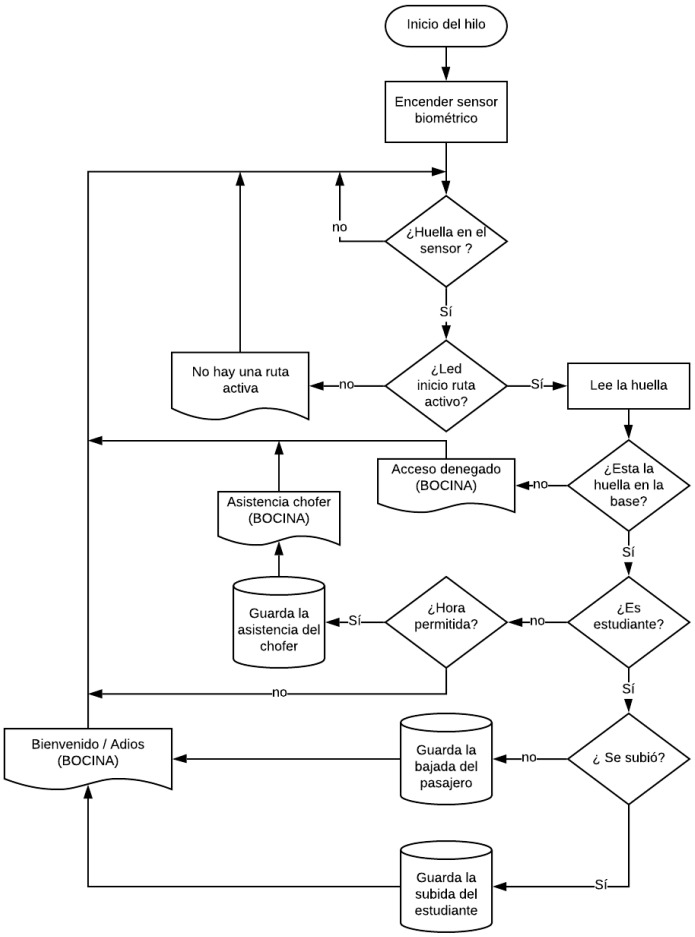


Figura 2.7. Diagrama de flujo del programa en la RPi para el módulo huella

2.4.4. MÓDULO DE CONTROL ELECTRÓNICO

Debido al requerimiento RFH04 de la Tabla 2.2 se ha consideró agregar el módulo denominado control electrónico, que estará integrado a la RPi, el cual a través de pulsadores permitirá enviar información a la base de datos, esta información luego será presentados en los diferentes aplicativos. Además, contará con diodos led con el objetivo de indicar que acción (inicio, fin de ruta o emergencia) fue activada.

2.4.4.1 Conexión del módulo de control electrónico a la RPI

Mediante el software de simulación Proteus y con los requerimientos previamente obtenidos (RFH04), se llevó a cabo la conexión de los dispositivos para el módulo de control electrónico, el cual se presenta a continuación, en la Figura 2.8.

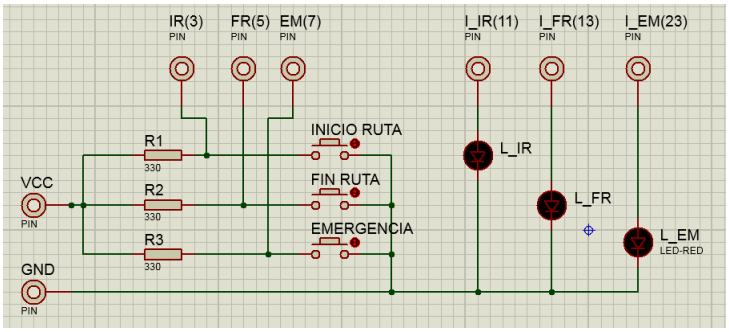


Figura 2.8. Conexión del módulo de control electrónico a la RPi

A continuación, en la Tabla 2.33 se detalla la función de cada elemento electrónico que forma parte de este módulo.

Tabla 2.33. Funciones de los elementos integrados al módulo

Elemento	Nomenclatura	Función
PIN	VCC	Polarización del circuito.
	GND	
	IR	Conexión hacia la RPi para realizar la acción de iniciar la ruta mediante el pulsador directamente conectado.
	FR	Conexión hacia la RPi para realizar la acción de finalizar la ruta mediante el pulsador directamente conectado.
	EM	Conexión hacia la RPi para realizar la acción de notificar una emergencia mediante el pulsador directamente conectado
	I_IR	Conexión desde RPi para encender el led de inicio de ruta.
	I_FR	Conexión desde RPi para encender el led de fin de ruta.
	I_EM	Conexión desde RPi para encender el led de emergencia.
RESISTENCIA	R1	Configuración de pull up para identificar de mejor manera cuando esta activado/desactivado cada pulsador directamente conectado a cada resistencia.
	R2	
	R3	
PULSADOR	INICIO RUTA	Realiza la acción de crear una nueva ruta.
	FIN RUTA	Realiza la acción de finalizar una ruta que estuvo activa.
	EMERGENCIA	Realiza la acción de notificar que ha sucedido una acción.
LED	L_IR	Indica al chofer que la ruta ha iniciado y con ello los estudiantes podrán subirse a la unidad de transporte.
	L_FR	Indica al chofer que la ruta ha finalizado.
	L_EM	Indica al chofer que ha notificado a todos los usuarios de esa unidad que esta tiene una emergencia.

2.4.4.2 Diagrama de flujo del programa en la RPi para el módulo de control electrónico

A continuación, en la Figura 2.9 se presenta el diagrama de flujo del programa en la RPi para el módulo de control electrónico, el cual empieza verificando en la base de datos si la ruta está activa o si ya ha finalizado, esta verificación es importante, debido a que puede existir un corte de energía y con ello se recupera el estado de la ruta. Posteriormente se

guardan los estados de los pulsadores para evaluar si el pulsador de inicio de ruta, fin de ruta o emergencia fueron activados.

Si fue activado el pulsador de emergencia, la RPi almacena en la base de datos la información (estado, hora y fecha) correspondiente a la emergencia y activa su indicador (led), el mismo que estará activo por un determinado tiempo, transcurrido este tiempo el estado de emergencia en la base de datos cambiará y el indicador de emergencia (led) se desactivará.

Si fue activado el pulsador de inicio de ruta o fin de ruta se evalúa la ubicación para determinar si está o no la unidad remota en una parada (sea la de inicio o de fin), posteriormente se crea una ruta y se activa el indicador de inicio de ruta (led) o se finaliza la ruta y se activa el indicador de fin de ruta (led). En el caso de que se inicia una ruta también se comprueba si existe una tabla de asistencia para ese día para que al chofer de esa unidad se le pueda registrar su asistencia, de no estar creada, se crea para ese día una tabla de asistencia.

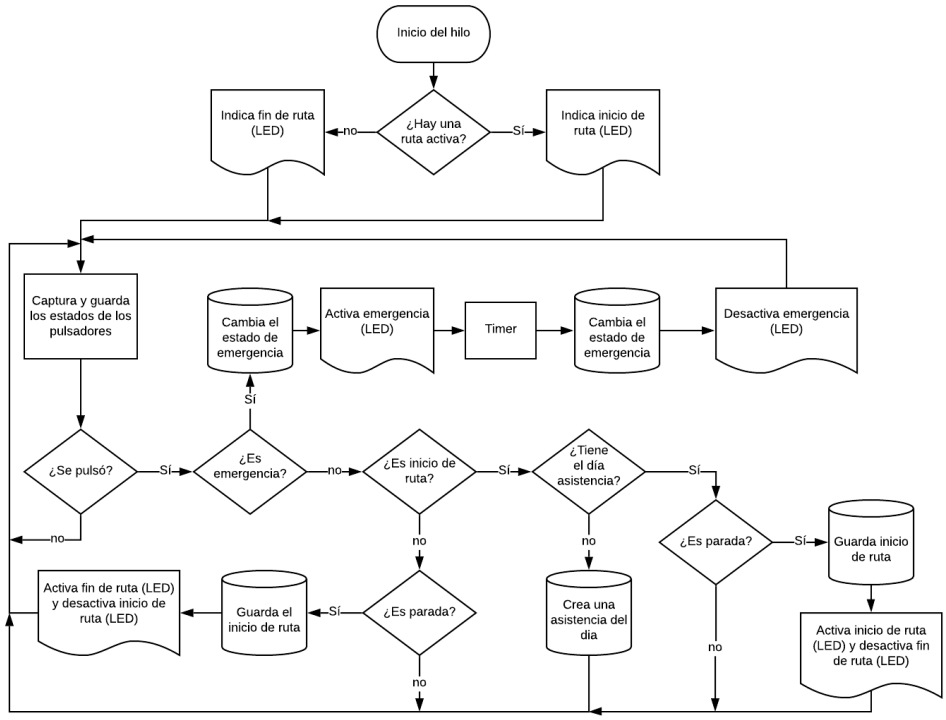


Figura 2.9. Diagrama de flujo del programa en la RPi para el módulo de control electrónico

2.4.5. MÓDULO CÁMARA

Debido al requerimiento RFH03 de la Tabla 2.2 se agregó el módulo cámara, el cual se encargará de grabar los eventos sin audio que acontezcan dentro de la unidad de transporte en la memoria de la RPi, también se instalará y configurará el software ZoneMinder, software que permitirá controlar y monitorear de forma local/remota la cámara. Se debe mencionar que para monitorear de forma remota se configurará el software en modo multiservidor.

Por el requerimiento RNF05 de la Tabla 2.29 se seleccionó la cámara RaspiCam Pi NoIR Camera V2 y además por su compatibilidad con la RPi. A continuación, se presenta un breve resumen del dispositivo seleccionado.

2.4.5.1 RaspiCam Pi NoIR Camera V2

Es una cámara diseñada especialmente para la RPi (ver Figura 2.10) la cual permite tomar fotografías y grabar videos con resolución, tanto en el día como en la noche [10].

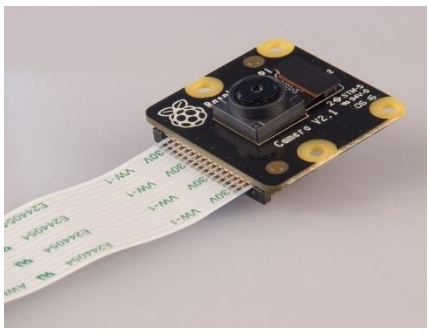


Figura 2.10. Pi NoIR Camera V2 [10]

2.4.5.2 Software ZoneMinder

Debido al requerimiento RFH03 de la tabla 2.2 se eligió este software para el monitoreo y control de la cámara debido a la configuración del modo multiservidor, este modo permitirá acceder a las cámaras instaladas en cada unidad de transporte de forma remota desde un servidor centralizado, es gratuito, tiene una interfaz amigable, fácil de usar y es compatible con la RaspiCam y el sistema operativo Raspbian.

2.4.5.2.1. Multiservidor en ZoneMinder

Como se mencionó anteriormente está opción de configuración permite ejecutar múltiples servidores ZoneMinder en diferentes dispositivos y poder administrarlos desde un solo servidor centralizado, cada servidor debe estar conectado a un único servidor de base de datos de esta manera comparte el almacenamiento de archivos de datos de eventos [37].

Para la configuración de multiservidor se ha planteado crear una máquina virtual en AWS[11], esta máquina será el servidor centralizado al cual se conectarán todos los servidores ZoneMinder de cada unidad de transporte logrando de esta manera visualizar video Streaming o grabaciones de eventos almacenados en cada RPi (ver Figura 2.11).

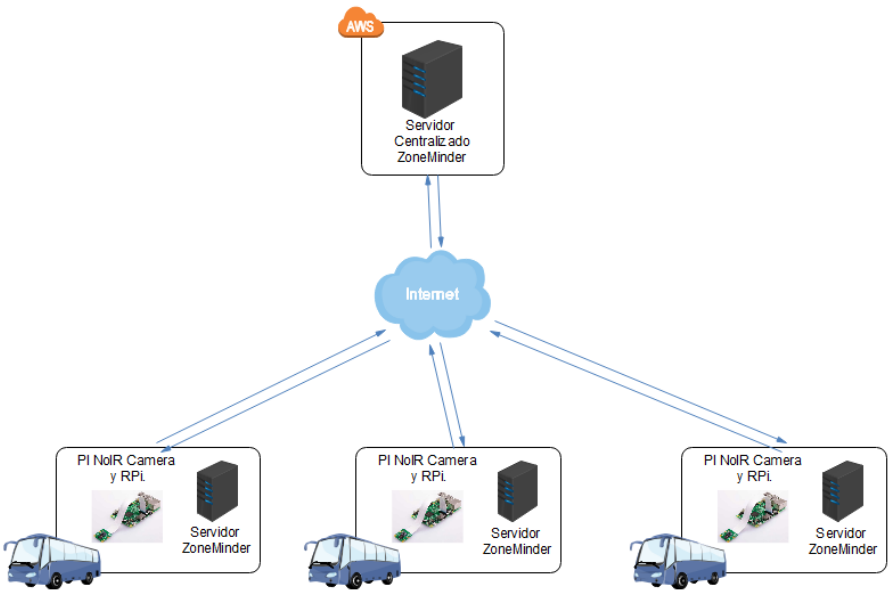


Figura 2.11. Topología ZoneMinder Multiservidor

2.5. MÓDULOS DEL PROTOTIPO

Como se ha mencionado el prototipo contará con dos aplicativos, a continuación, se definen los módulos correspondientes a cada uno.

2.5.1. APLICATIVO WEB

Los módulos se definieron en base a los requerimientos y contará con los siguientes módulos:

- Login: permite al administrador ingresar al sistema mediante el método de autenticación otorgado por el servicio Firebase Authentication.
- Principal: permite al administrador visualizar detalles acerca de cada una de las unidades de transporte en el caso que se encuentren realizando una ruta en ese preciso momento.

[11] AWS: (Amazon Web Services), plataforma que ofrece un conjunto de servicios de computación en la nube ofrecida por www.amazon.com.

- Administración: permite al administrador ver información o crear, estudiantes, unidades de transporte, carreras y choferes para poder modificarlos o eliminarlos.
- Registro: permite al administrador visualizar los registros de asistencia de los estudiantes, como para los choferes.
- Cámara: la cual enlaza hacia el servidor centralizado ZoneMinder.
- Cerrar sesión: permite al administrador cerrar el aplicativo web.

La Tabla 2.34 muestra las historias de usuario que corresponden a cada módulo.

Tabla 2.34. Módulos del aplicativo web

Módulo	Historia de usuario	Descripción de historia
Login	HUW01	Iniciar sesión.
	HUW02	Recuperación de contraseña.
Home	HUW03	Visualización de la página principal.
Registro	HUW04	Registros de asistencia chofer.
	HUW05	Registros de asistencia estudiante.
Administración	HUW06	Administración de usuarios.
	HUW07	Administración de bus.
	HUW08	Administración de carreras.
Cámara	HUW09	Ingreso a ZoneMinder.
Cerrar sesión	HUW10	Cerrar sesión.

2.5.2. APLICATIVO MÓVIL

Para el diseño de las pantallas (módulos) que tendrá el aplicativo móvil se tomó en consideración los requerimientos por los usuarios (administrador y estudiantes) que harán uso de este aplicativo. Por lo tanto, se consideran las siguientes pantallas:

- Login: permite ingresar al estudiante o administrador al sistema mediante Firebase Authentication.
- Inicio: comprueba el inicio de sesión del usuario y redirige al usuario a su pantalla correspondiente.
- Información: permite al estudiante visualizar la descripción de la unidad de transporte, la ubicación de la unidad tanto en el mapa de Google, la fecha y hora en la cual la unidad de transporte inicio su recorrido y el número de asientos disponibles que aún existen en dicha unidad.
- Control: permite al administrador monitorear a todas las unidades de transporte y visualizar los eventos que ocurren en la unidad a través de la cámara.
- Estudiantes: muestra un registro de los estudiantes actuales dentro de la ruta, esta pantalla es únicamente para el perfil de administrador.

- Emergencias: permite al administrador ver un registro de emergencias ocurridas en la unidad, además, eliminar estas si así fuera el caso.
- ZoneMinder: permite la navegación que el software ZoneMinder ofrece mediante la AWS.

La Tabla 2.35 muestra las historias de usuario que corresponden a cada módulo.

Tabla 2.35. Pantallas del aplicativo móvil

Pantalla	Historia de usuario	Descripción de historia
Login	HUM01	Iniciar sesión.
	HUM02	Recuperación de contraseña.
Inicio	HUM03	Visualización de la página inicio.
Información	HUM04	Visualización de información.
Control	HUM05	Visualización de control.
ZoneMinder	HUM06	Ingreso a ZoneMinder.
Estudiantes	HUM07	Registro actual de estudiantes.
Emergencias	HUM08	Visualización de emergencia.

2.6. DISEÑO DEL SISTEMA DE COMUNICACIÓN

Debido al requerimiento RNF02 de la Tabla 2.29, a continuación, se describe y se muestra el diseño para la plataforma Firebase en la cual se albergará la base de datos, además se presenta el diseño de los aplicativos definidos en el plan de titulación.

2.6.1. CONFIGURACIÓN EN FIREBASE

Se describen los servicios de la plataforma Firebase que intervienen en el prototipo, la selección de tipo de autenticación, el diseño del árbol JSON con los nodos que serán agregados en la base de datos.

Lo primero que se deberá hacer es crear una cuenta en la plataforma Firebase, para esto es necesario poseer una cuenta de Gmail, luego se podrán crear los proyectos los cuales tendrán acceso a los servicios otorgados por la plataforma.

2.6.1.1 Diseño de la base de datos en Firebase Realtime Database

Una vez creado el proyecto en la plataforma como siguiente paso será crear la base de datos dentro del servicio Realtime Database, luego configurar las reglas de seguridad. El modo elegido será modo de prueba ya que para el proyecto va a existir una constante lectura y escritura de datos de terceros y a partir de esto ya se puede crear el árbol JSON.

Para poder manipular los datos obtenidos desde la RPi por los aplicativos, se han creado los siguientes nodos:

- **BUS. -** es un nodo principal que contendrá varios nodos secundarios los cuales corresponderán a cada una de las unidades de transporte (suponiendo que se instalen varios prototipos), la clave para cada nodo secundario denominada *ID_BUS* será la dirección MAC de la RPi. Cada nodo secundario se ramifica de la siguiente forma:
    - ***PARADA. -*** es un nodo secundario que guardará los datos que permitirán comprobar que la unidad se encuentre en la parada de inicio o fin de ruta y tendrá las siguientes ramificaciones:
        - **latitudInicio. -** valor de la latitud de la parada inicial, este campo será enviado al crear un bus en el aplicativo web y su valor será constante e igual a la coordenada de latitud que se genera en la dirección de la parada ubicada en la EPN.
        - **longitudInicio. -** valor de la longitud de la parada inicial, este campo será enviado al crear un bus en el aplicativo web y su valor será constante e igual a la coordenada de longitud que se genera en la dirección de la parada ubicada en la EPN.
        - **latitudFin. -** valor de la latitud de la parada final, este campo será enviado al crear un bus en el aplicativo web.
        - **longitudFin. -** valor de la longitud de la parada final, este campo será enviado al crear un bus en el aplicativo web.
    - **capacidad. -** este valor contiene el número de asientos que tiene la unidad de transporte y será agregado cuando se cree un bus desde el aplicativo web.
    - **línea. -** el valor corresponderá al número de línea, el cual identifica a la unidad de transporte y será almacenada cuando se cree un bus desde el aplicativo web.
    - **descripción. -** el valor de esta ramificación corresponde al nombre del lugar a donde se realiza la ruta y será enviado desde el aplicativo web al crear un bus.
    - **latitud. -** este valor va a ir cambiando según marque el módulo GPS y almacenará la coordenada de latitud la cual permitirá ubicar al bus en la posición actual dentro del mapa Google que se encontrará en los aplicativos.

    - **longitud. -** este valor va a ir cambiando según marque el módulo GPS y almacenará la coordenada de longitud la cual permitirá ubicar al bus en la posición actual dentro del mapa Google que se encontrará en los aplicativos.
- **CARRERA. –** es un nodo principal y dentro de este nodo se ubicarán los nodos secundarios que corresponden al nombre de cada facultad que serán creadas desde el aplicativo web, la clave *ID_CARRERA* para cada nodo será un identificador numérico, cada nodo tendrá como única ramificación a:
    - **nombreCarrera. -** el valor de esta ramificación será el nombre de la carrera.
- **HUELLA. -** es un nodo principal que contiene varios nodos secundarios que almacenará las huellas de los estudiantes (para realizar el control de acceso) de los choferes (para realizar el control de asistencia), cada nodo secundario tendrá como clave *ID_USUARIO* la cual corresponde a la cédula del estudiante o chofer y tendrá como ramificación a:
    - ***CODIGOHUELLA. -*** este será el nombre a este nodo donde se almacenará el código de la huella dactilar generado por el módulo registro desde el aplicativo web al enrolar a un estudiante o chofer, el código que se genera se presentado en una matriz por esa razón este nodo tendrá 511 ramificaciones o números ([i]).
        - [i]. - cada número [i] corresponde a un elemento de la matriz generada por el sensor biométrico, esta matriz es única y corresponde al código generado de la imagen escaneada de la huella dactilar.
- **MODULOREGISTRO. -** este nodo permitirá activar el módulo registro desde la aplicación web para poder enrolar a los estudiantes y contiene las siguientes ramificaciones:
    - **estadoHuella. -** este valor permitirá notificar al aplicativo web si la huella a enrolar se ha escaneado correctamente.
    - **opcionEnrolar. -** este valor contendrá un número 0 o 1 seguido de una coma junto con la cédula del estudiante o chofer a ser enrolado.
- **USUARIO. -** este nodo almacenará a los estudiantes, administradores que utilizarán los aplicativos y a los choferes para realizar el control de asistencia, se llenará cuando se agreguen los usuarios desde el aplicativo web y tendrá como clave a *ID_USUARIO* que corresponde a la cédula del usuario. Además, estará conformado por las siguientes ramificaciones:
    - **nombres. -** cuyo valor será los nombres del usuario.

- **apellidos. -** cuyo valor será los apellidos del usuario.
- **correo. -** cuyo valor será el correo del usuario.
- **carrera. -** este valor corresponde a la carrera a la cual pertenece el estudiante, este valor será útil para realizar los formularios de control de acceso, para el rol “Administrador” y “Chofer” este valor será igual a “ninguno”.
- **línea. -** cuyo valor será el número de línea de la unidad transporte correspondiente del “Estudiante”, o corresponderá a la línea que maneja en el caso del “Chofer” y si el usuario es “Administrador” este valor será igual a “ninguno”.
- **rol. -** el valor de esta ramificación podrá ser “Administrador”, “Estudiante” o “Chofer”, y servirá para acceder al correspondiente perfil dentro del aplicativo móvil para el caso de los estudiantes y el administrador, pero para el rol “Chofer” permitirá realizar los registros de control de asistencia en el aplicativo web.

- **RUTA. -** nodo principal que almacenará las rutas junto sus correspondientes pasajeros. De este nodo principal se desprende en primer lugar un nodo con clave *ID_BUS* que corresponde a la MAC del RPi este nodo permitirá a las aplicaciones conocer con claridad a que dirección se debe apuntar para leer los datos guardados en la base, ya que a cada unidad de transporte le corresponde una RPi. Bajo el nodo con clave *ID_BUS* se agregarán los nodos de las rutas que se realicen, estos nodos de ruta con clave *ID_RUTA* se crearán desde el módulo de control electrónico cuando se inicie una ruta al presionar el botón “Inicio” con clave generada por el mismo Firebase, este nodo de rutas tendrá como ramificaciones:
  - ***PASAJERO. -*** este nodo permitirá identificar que pasajeros se han subido a la unidad de transporte en esa ruta específica, bajo este nodo PASAJERO se agregarán varios nodos que corresponderán a los pasajeros estos nodos cuya clave *ID_USUARIO* al cual será la cédula de identidad del pasajero y tendrán las siguientes ramificaciones:
    - ***estado. -*** cuyo valor será False cuando el pasajero se haya bajado de la unidad o True cuando aún se encuentre dentro de la unidad y permitirán conocer número de pasajeros que se han subido y número de pasajeros que se han bajado de la unidad, estos números serán presentados en los aplicativos.
    - ***direccionBajada.*** - cuyo valor será la dirección del lugar donde se bajó el pasajero.

        - ***direccionSubida. -*** cuyo valor será la dirección del lugar donde se subió el pasajero.
    - **fechaFinRuta. -** cuyo valor será igual a la hora y fecha en la que se finalizó la ruta específica.
    - **fechaInicioRuta. -** cuyo valor será igual a la hora y fecha en la que se inició la ruta específica.
- **ASISTENCIA.** - este nodo almacenará los datos que permitirán realizar los registros de control de asistencia de los choferes de las unidades de transporte, la clave denominada *ID_USUARIO* para los nodos secundarios será la cédula de identidad del chofer y tendrá las siguientes ramificaciones:
    - **fecha.** - cuyo valor será la fecha cuando timbro.
    - **hora1**. - cuyo valor será igual a la hora en la cual timbro al empezar a realizar la ruta en la mañana
    - **hora2.** - cuyo valor será igual a la hora en la cual timbro al finalizar la ruta en la mañana.
    - **hora3**. - cuyo valor será igual a la hora en la cual timbro al empezar a realizar la ruta en la noche.
    - **hora4**. - cuyo valor será igual a la hora en la cual timbro al finalizar la ruta en la noche.
    - **línea**. - cuyo valor será el número de línea de la unidad de transporte.
- **EMERGENCIA.** - con la ayuda de este nodo principal se podrá generar las notificaciones de emergencia en el aplicativo móvil, la clave para los nodos secundarios es *ID_BUS* que corresponde a la dirección MAC de RPi que permitirá identificar que unidad de transporte presionó el botón de emergencia en el módulo de control electrónico, cada nodo correspondiente a emergencia tendrá como clave *ID_EMERGENCIA* las cuales son autogeneradas, estos nodos tendrán las siguientes ramificaciones:
    - **estado.** - su valor será “False” o “True”, el valor igual a True permitirá activar una notificación.
    - **fecha. -** cuyo valor será igual a la fecha y hora en la cual se generó la notificación.
- **HORARIO.** - nodo principal que almacenará los horarios de la mañana y noche de cada uno de los buses, se creará un nodo secundario por bus, cuya clave será *ID_BUS* el cual corresponde la MAC de la RPi y cada uno tendrá las siguientes ramificaciones:

- **hora1**. - hora de inicio de ruta en la mañana
- **hora2**. - hora de fin de ruta en la mañana.
- **hora3**. - hora de inicio de ruta en la noche.
- **hora4**. - hora de fin de ruta en la noche.

En la Figura 2.12 se puede apreciar el diseño del árbol JSON para la base de datos.

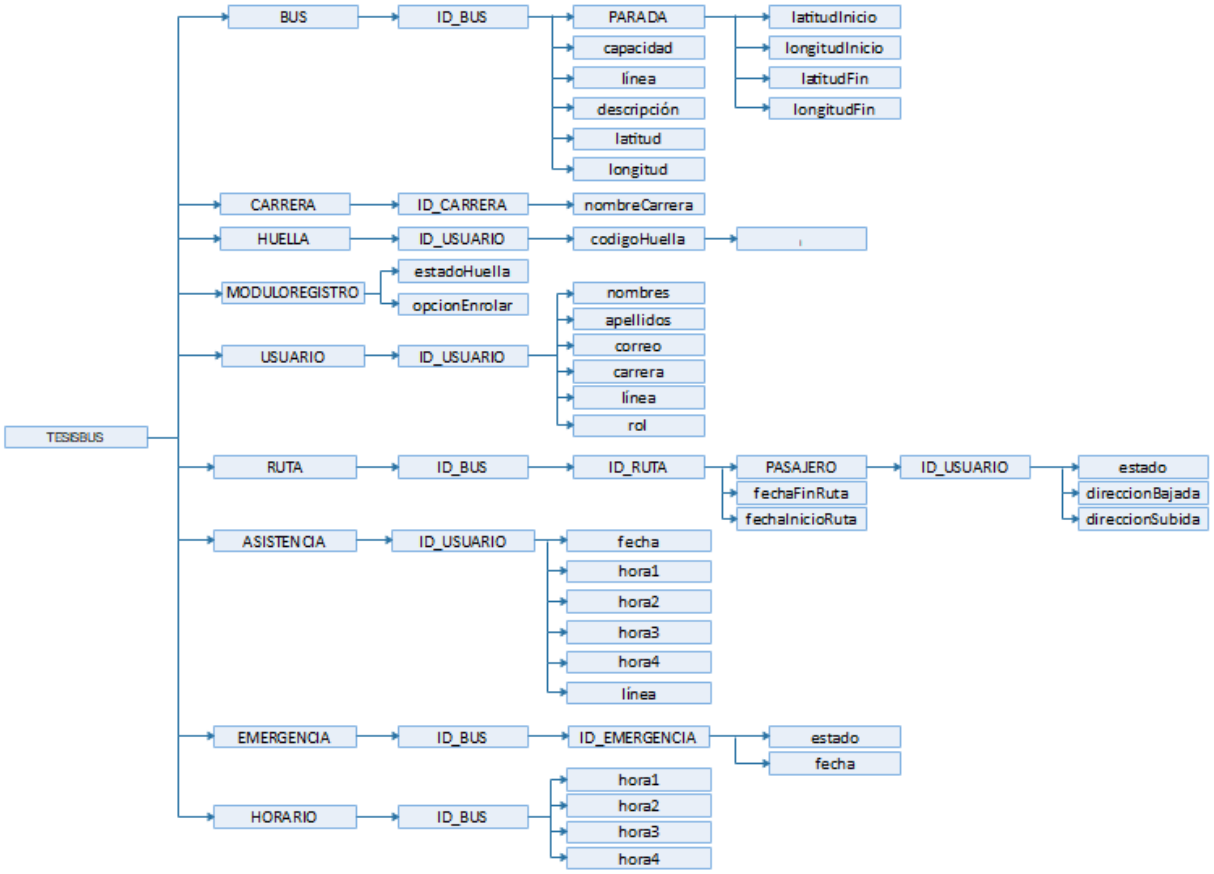


Figura 2.12. Diseño Árbol JSON del prototipo

2.6.1.2 Selección de tipo de autenticación de Firebase Authentication

Para realizar la autenticación se ha elegido hacerlo mediante el método de correo electrónico y contraseña, para lo cual es necesario añadir una interfaz de inicio de sesión que permita ingresar estos datos en los dos aplicativos definidos para este proyecto.

Se debe habilitar el servicio de autenticación desde consola en el proyecto creado, luego configurar el método elegido de inicio de sesión donde existen varias opciones como teléfono, a través de redes sociales, cuentas de correo, GitHub, entre otras, una vez realizado esto, ya se puede gestionar usuarios ya sea desde consola o desde los aplicativos, para lo cual es necesario crear usuarios con un correo válido y una contraseña

cuyos caracteres sean mayor o igual a 8 y de esta forma el usuario podrá usar los aplicativos definidos para el desarrollo del prototipo.

2.6.1.3 Firebase Hosting

Debido a que el prototipo desarrolla un aplicativo web se ha decido utilizar el servicio de Firebase Hosting de esta manera se otorgará un dominio gratuito al que podrán acceder los usuarios, para ello primero se debe habilitar, instalar y configurar Firebase en el servidor web y añadir los SDK que se usarán dentro del código del aplicativo, una vez hecho esto, ya se puede alojar los formularios HTML y script JavaScript dentro del proyecto de Firebase.

2.6.2. DISEÑO DEL APLICATIVO WEB

En esta sección se describirá los diagramas de casos de uso del aplicativo web, diagramas de actividades y sus correspondientes sketches.

2.6.2.1 Diagrama de caso de uso

Luego de los análisis de requerimientos de la Tabla 2.14, se define un solo actor para el aplicativo web. A continuación, se detallan los diagramas de caso de uso para el aplicativo.

Los casos de uso se los clasifica dependiendo de los módulos definidos anteriormente para el aplicativo web, el título de cada caso de uso corresponde al nombre del requerimiento funcional.

Los casos de uso correspondientes a **"Módulo Login"**, que permite autentificar y recuperar contraseña, son los siguientes:

La Figura 2.13 muestra el caso de uso para iniciar sesión, el administrador puede ingresar a la aplicación web cuando se verifiquen sus datos.

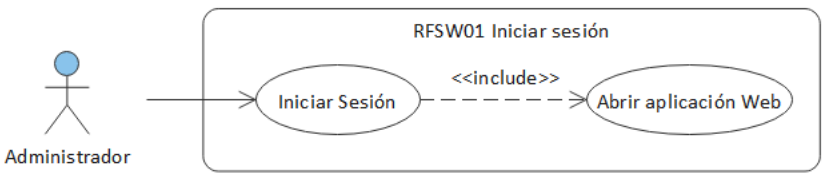


Figura 2.13. Casos de uso iniciar sesión.

La Figura 2.14 muestra el caso de uso para recuperar contraseña como usuario administrador.

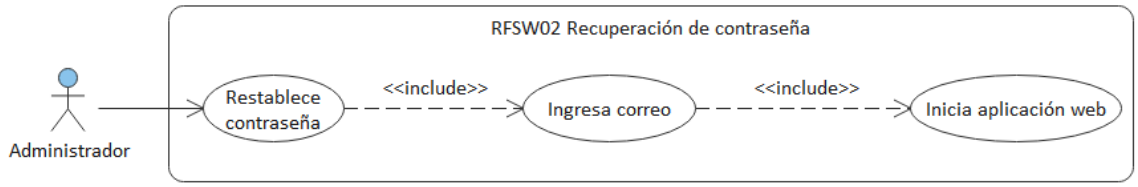


Figura 2.14. Caso de uso recuperación de contraseña

El caso de uso correspondiente al **"Módulo Home"**, el cual permite visualizar la información actual de la unidad de transporte, es el siguiente:

La Figura 2.15 muestra el caso de uso para visualizar la página principal que permite al administrador conocer por unidad de transporte cual está realizando una ruta, cuantos pasajeros se encuentran actualmente en esa ruta, la ubicación en tiempo real de la unidad de transporte.

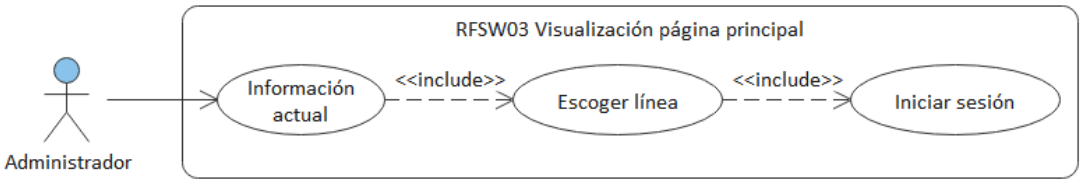


Figura 2.15. Caso de uso visualización página principal.

Los casos de uso correspondientes para el **"Módulo Registro",** el cual permite generar y descargar registros de asistencia, son los siguientes:

La Figura 2.16 permite al administrador generar un registro diario de asistencia para los choferes y descargar estos registros en formato .pdf.

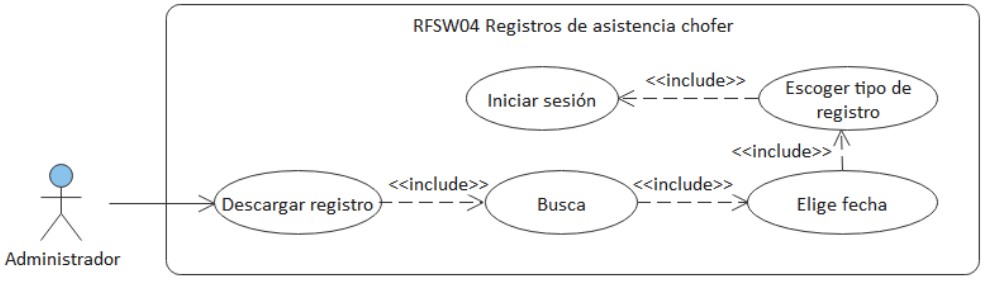


Figura 2.16. Caso de uso registro de asistencia chofer

La Figura 2.17 muestra el caso de uso para registro de asistencia de estudiante, permite al administrador generar un registro diario o semanal de los estudiantes que utilizaron la unidad de transporte para luego descargarlo en formato .pdf.

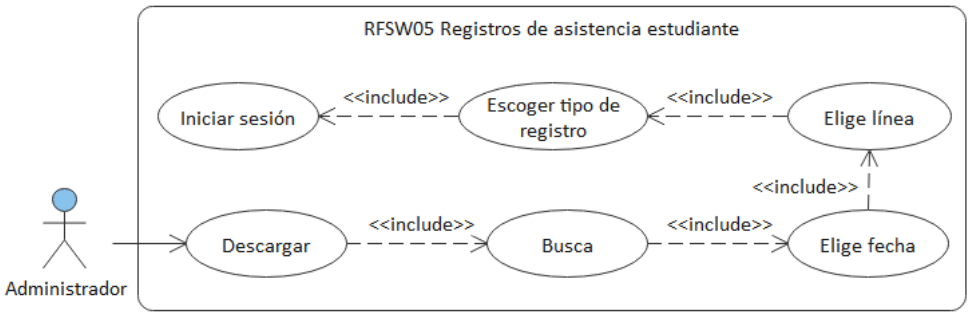


Figura 2.17. Caso de uso registro de asistencia estudiante

Los casos de uso correspondientes para el "**Módulo Administración**", el cual permite gestionar a los usuarios, unidades de transporte y carrera, son los siguientes.

La Figura 2.18 muestra el caso de uso para administrar usuarios, permite al administrador gestionar los usuarios, esto es crear, eliminar o editar usuarios.

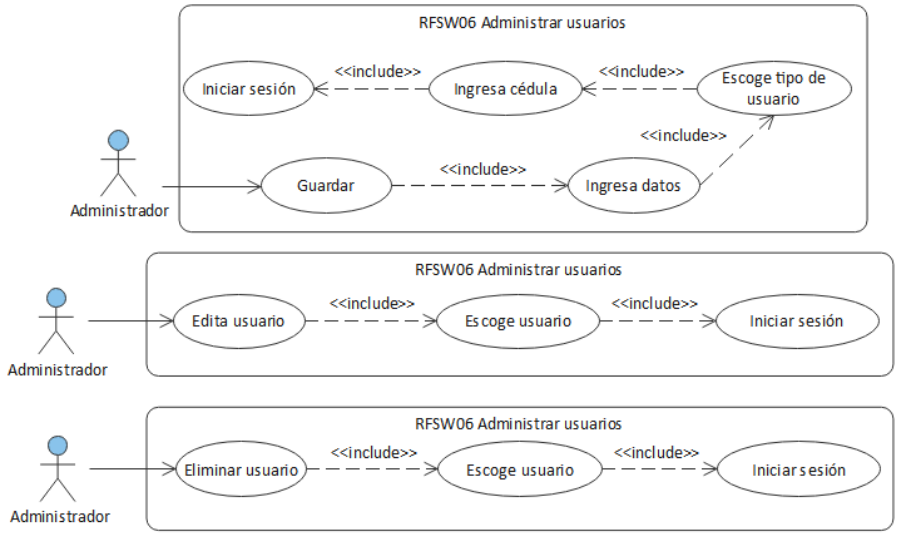


Figura 2.18. Casos de uso administrar usuarios

La Figura 2.19 muestra el caso de uso para administrar buses, permite al administrador gestionar los buses, esto es crear, eliminar o editar buses.

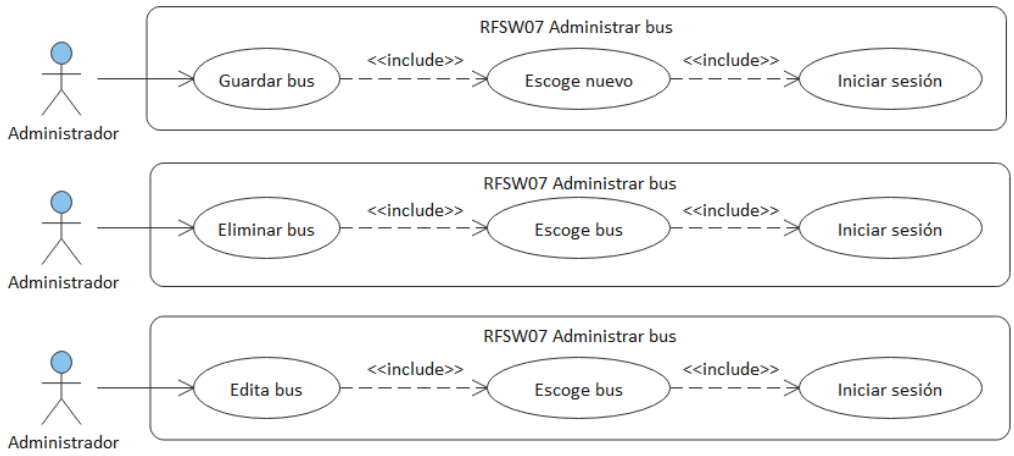


Figura 2.19. Caso de uso administrar bus

La Figura 2.20 muestra el caso de uso para administrar carreras, permite al administrador gestionar las carreras, esto es crear, eliminar o editar carreras.

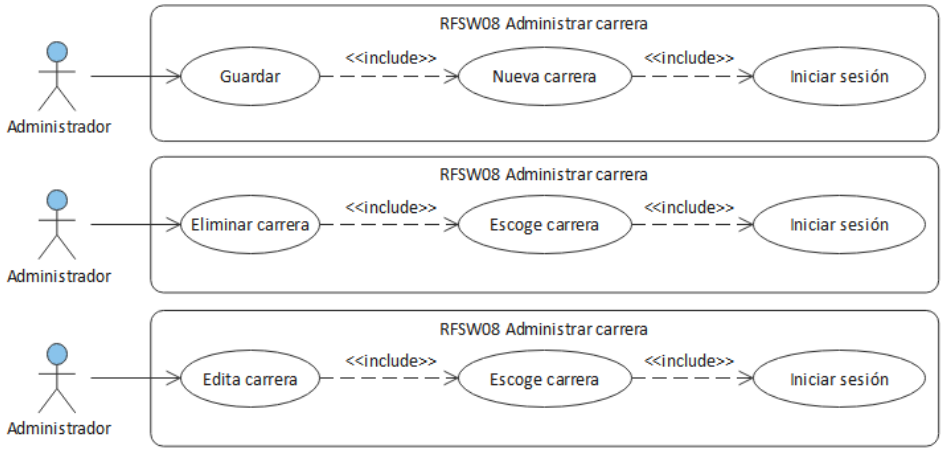


Figura 2.20. Casos de uso para administrar carrera

El caso de uso para el **"Módulo Cámara",** el cual permite visualizar las imágenes captadas por la cámara, es el siguiente:

La Figura 2.21 muestra el caso de uso para ingresar a ZoneMinder, permite al administrador enlazarse con el software ZoneMinder para ver las grabaciones efectuadas en cada unidad de transporte.

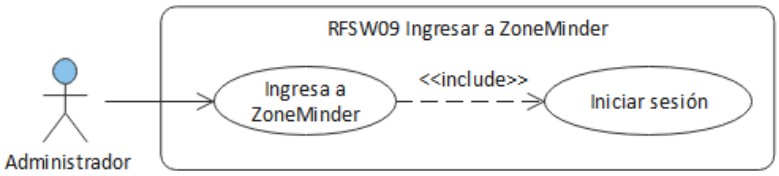


Figura 2.21. Caso de uso para ingresar a ZoneMinder

2.6.2.2 Diseño de los sketches

Para la elaboración de los sketches se ha utilizado la herramienta Pencil Proyect de código abierto, la cual nos permite crear sketch de interfaz de usuario (UI). La Figura 2.22 muestra la interfaz de la herramienta, cuenta con elementos de páginas web, entre otros.

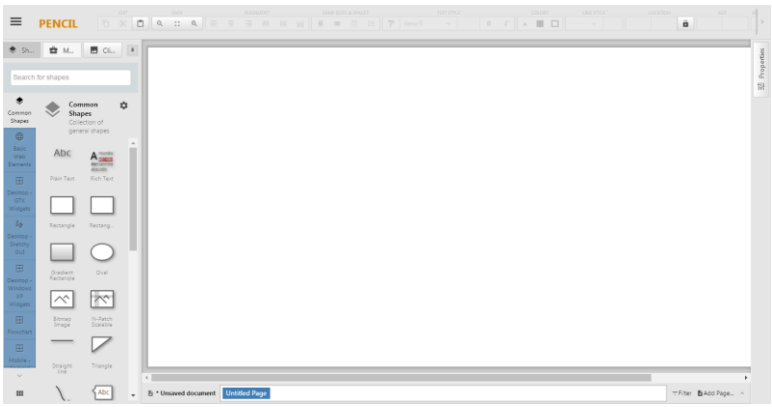


Figura 2.22. Interfaz de Pencil Proyect

Los sketches se elaboraron tomando en cuenta los módulos junto con las historias de usuario y la descripción de los requerimientos de la Tabla 2.14, estos sketches permitirán diseñar las UI en el aplicativo web. Como siguiente paso a realizar fue enlistar los formularios HTML a integrar a cada módulo del aplicativo web, estos son los siguientes:

Para el módulo Login se definió un solo formulario:

- índex: formulario de inicio de sesión.

Para el módulo Home se dispondrá de un solo formulario:

- principal: formulario que contiene información actual de cada unidad de transporte.

Para el módulo Registro se tendrán los siguientes formularios:

- regEstSemanal: formulario que contiene los registros de asistencia por semana de los estudiantes de la misma unidad de transporte.

- regEstDiario: formulario que contiene los registros de asistencia diario de los estudiantes pertenecientes a la misma unidad de transporte.
- registroChofer: formulario que contiene los registros semanales de asistencia de los choferes.

Para el módulo Administración se tendrán los siguientes formularios:

- administrarUsuario: formulario que permite crear, eliminar o modificar usuarios.
- administrarBus: formulario que permite crear, eliminar o modificar unidades de transporte.
- administrarCarrera: formulario que permite crear, eliminar o modificar facultades.

Definidos los formularios HTML se procedió a realizar el diseño de UI de cada formulario. A continuación, se enlista se presentan los sketches de cada formulario:

En la Figura 2.23 se presenta el diseño del sketch del formulario Login, el cual permitirá al administrador acceder al aplicativo web, este formulario contiene varios label y una imagen para que la UI sea amigable, los principales elementos son los input que corresponde al ingreso del correo electrónico y contraseña estos valores serán comparados con los datos almacenados en Firebase Autenticación al presionar el botón “Iniciar Sesión”, también se ha agregado un link “Restablecer contraseña” que permitirá ingresar el correo electrónico del administrador que desea restablecer su contraseña.

Figura 2.23. Sketch del formulario Login

Después de formulario Login siempre y cuando exista el usuario en la base, se mostrará el formulario principal y a su vez se podrá acceder a los diferentes formularios, para ello cada formulario cuenta con la misma barra de navegación la cual contendrá el nombre de los módulos definidos para este aplicativo.

En la Figura 2.24 se presenta el sketch del formulario principal, este formulario será el primero en verse luego de iniciar sesión, cada formulario tendrá una barra de navegación cuyas divisiones corresponden a cada módulo del aplicativo web.

El formulario principal permite ver la información actual de las unidades de transporte que estén realizando una ruta especifica en ese momento, permitirá seleccionar la unidad de transporte que se desea monitorear y a continuación, mostrará la ubicación de esta en el mapa Google, además, se cargarán los inputs con los valores de hora de inicio de la ruta,

número de estudiantes que se encuentran actualmente en el bus y número de asientos disponibles de ser el caso.

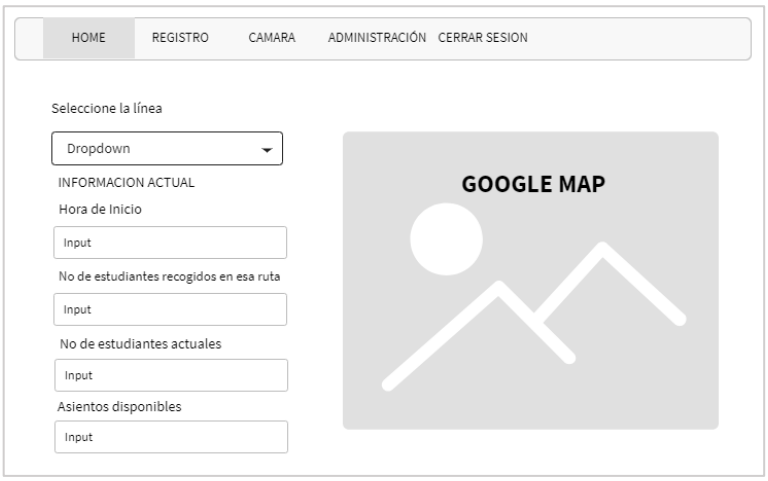


Figura 2.24. Sketch del formulario home

En la Figura 2.25 se presenta el sketch del formulario regEstDiario, para este formulario se deben ingresar dos datos: la unidad de transporte y el día para su selección se ha colocado un datepicker, una vez ingresado los datos se podrán ver los registros para ello se dispone del botón “Buscar” que permitirá enlistar los datos y presentarlos en una tabla, por último se colocará un imagen de PDF en un input que permitirá descargar en un documento PDF la tabla con los datos.

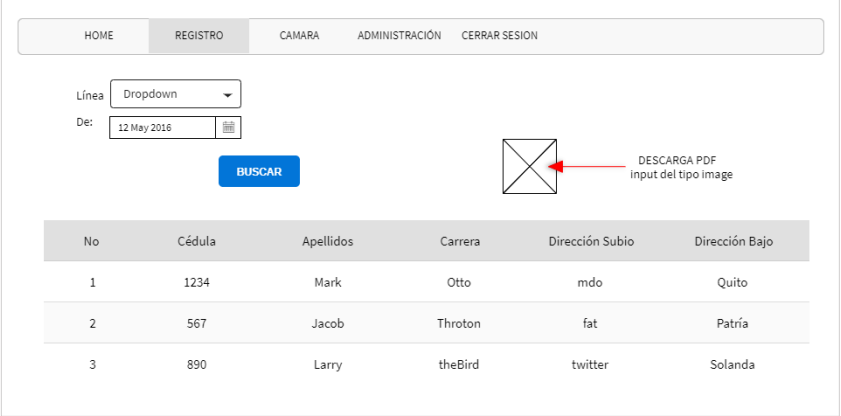


Figura 2.25. Sketch del formulario regEstDiario

A continuación, se presenta el sketch de los formularios de registroChofer (ver Figura 2.26), y regEstSemanal ,para estos formularios solo se ingresará el día en un datepicker para generar un registro de asistencia para los choferes o estudiantes, dependiendo del formulario, una vez ingresada la fecha se podrán ver los registros para ello se dispone del botón “Buscar” que permite enlistar los datos correspondientes y presentarlos en una tabla, únicamente para el caso del formulario regEstSemanal se adicionará una tabla resumen que mostrará el número de estudiantes que asistieron (ver Figura 2.27) y finalmente se colocará una imagen de PDF que permitirá obtener un documento en formato PDF de los registros.

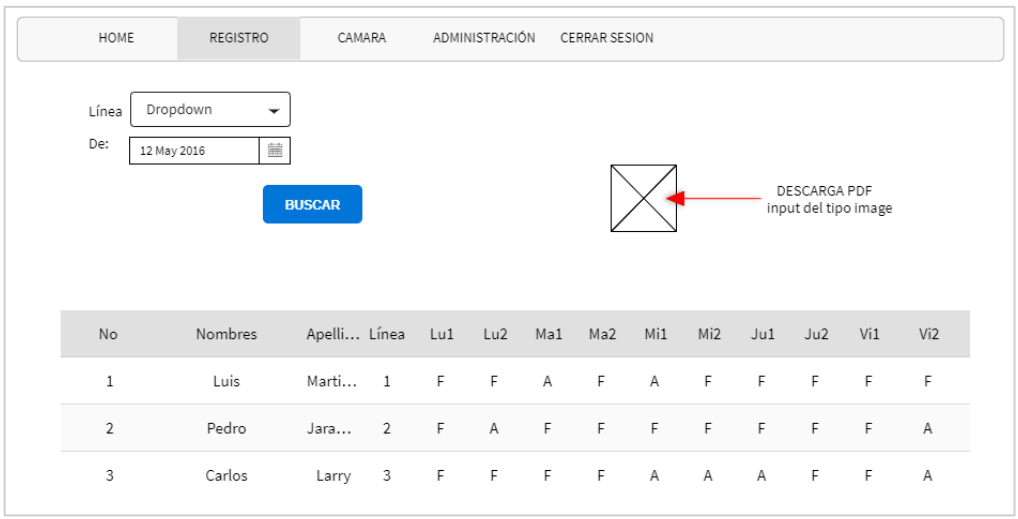


Figura 2.26. Sketch del formulario registroChofer

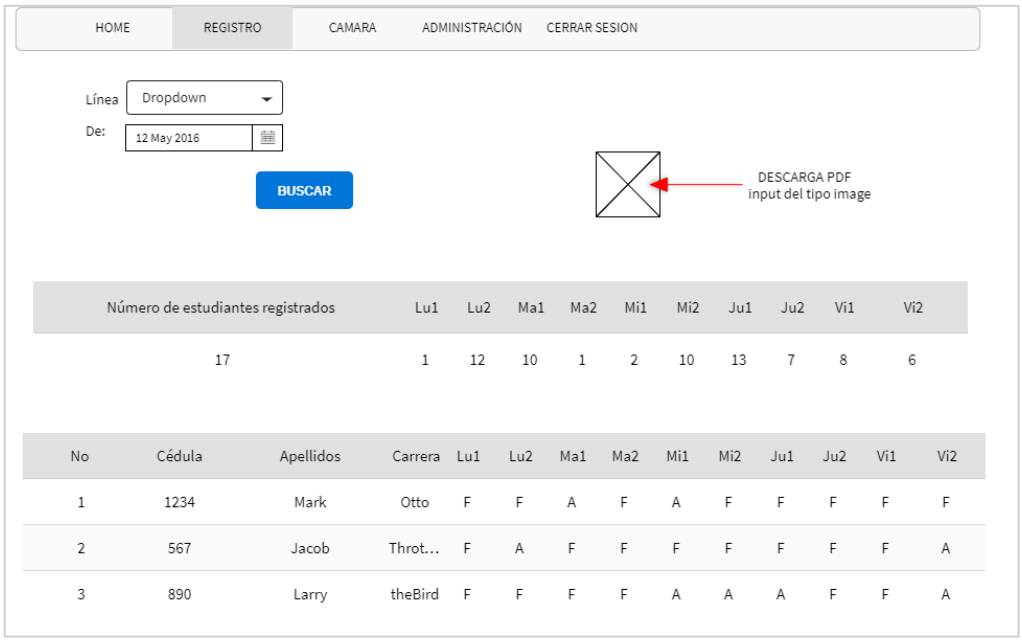


Figura 2.27. Sketch del formulario regEstSemanal

La Figura 2.28 muestra el sketch para el formulario administrarUsuario, este formulario permitirá agregar nuevos usuarios a la base de datos, tendrá un input donde se ingresará la cédula de usuario a agregar, se colocará un elemento select para elegir el rol del nuevo usuario previamente este elemento tendrá cargados los valores de “Administrador”, “Estudiante” y “Chofer”, tendrá un botón que activará al módulo registro para ingresar la nueva huella y el último elemento será una tabla que enlistará a los usuarios registrados, cada fila de la tabla tendrá dos input de tipo imagen estos servirán para eliminar o editar al usuario seleccionado.

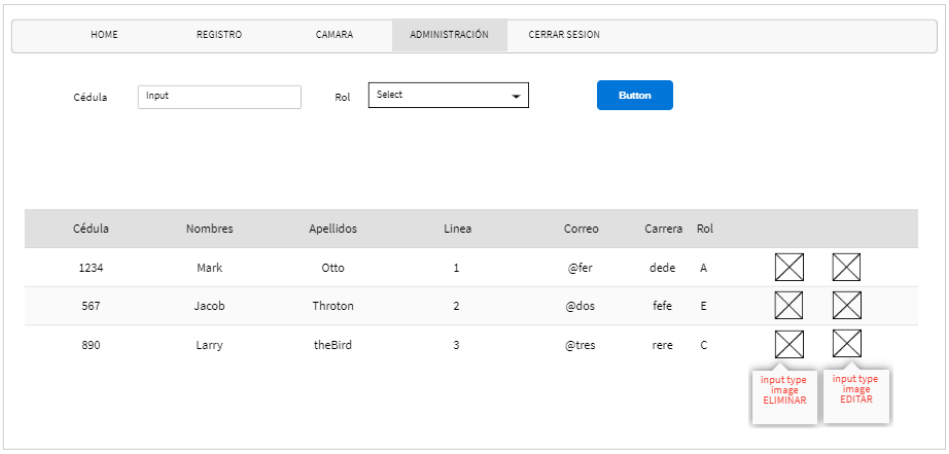


Figura 2.28. Sketch del formulario administrarUsuario

La Figura 2.29 muestra el sketh para el formulario administrarBus, este formulario tendrá un botón para agregar un bus este botón abrirá una ventana emergente (ver Figura 2.30)

en la cual se agregaron varios input que permitirán ingresar la información necesaria para crear un nuevo bus, a esta ventana emergente se le ha colocado un mapa, este mapa permitirá elegir la dirección de la parada final del bus y para mayor facilidad en cuanto a la búsqueda de la dirección parada final, se han agregado un input junto con un botón “Buscar” para mostrar en el mapa la dirección, también consta de cuatro datepickers del tipo time que permitirán registrar la asistencia del chofer. Al igual que las demás tablas de los otros formularios estos enlistaran los buses almacenados en la base y en cada fila se colocará dos inputs del tipo imagen que permitirán eliminar el bus seleccionado o editar su información en la base de datos.

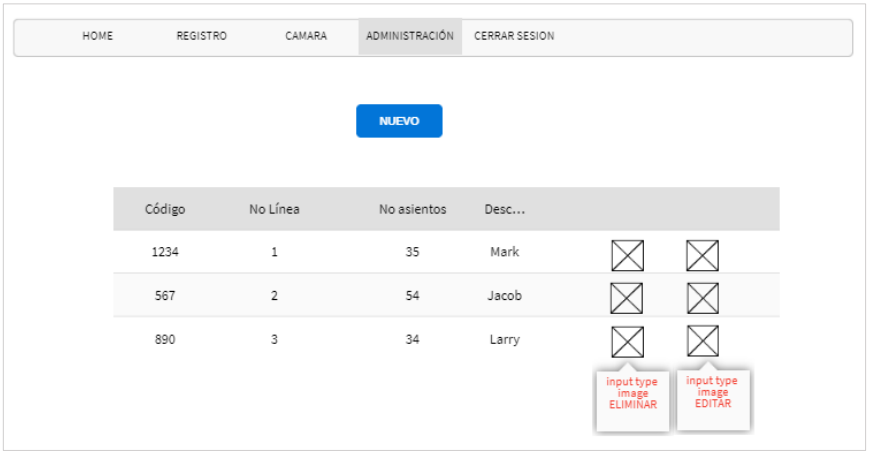


Figura 2.29. Sketch del formulario administrarBus

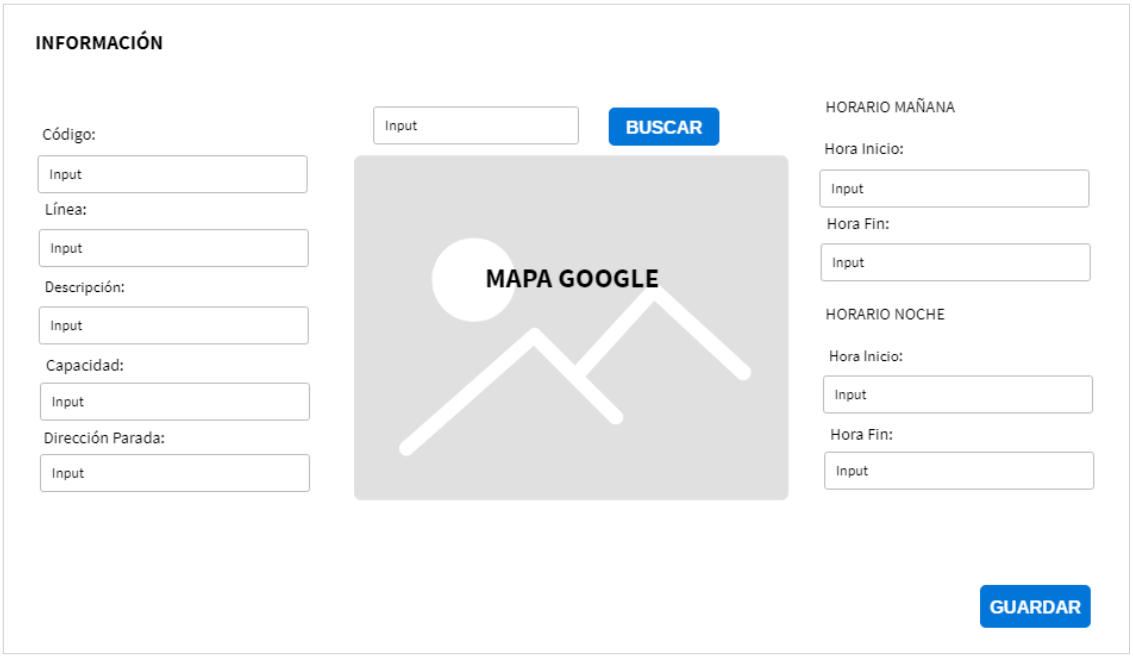


Figura 2.30. Sketch de la ventana emergente para crear un bus

La Figura 2.31 muestra el sketch para el formulario administrarCarrera, cuenta con dos inputs, uno se ellos será para la identificación de la carrera y el otro permitirá ingresar la descripción de la carrera, también tendrá un botón que dependiendo de la acción a realizar puede agregar una nueva carrera en la base, o editar cierta carrera seleccionada de la tabla, esta tabla enlistara las carreras de la base y finalmente cada fila tendrá un input del tipo imagen que permitirá eliminar o editar la carrera seleccionada.

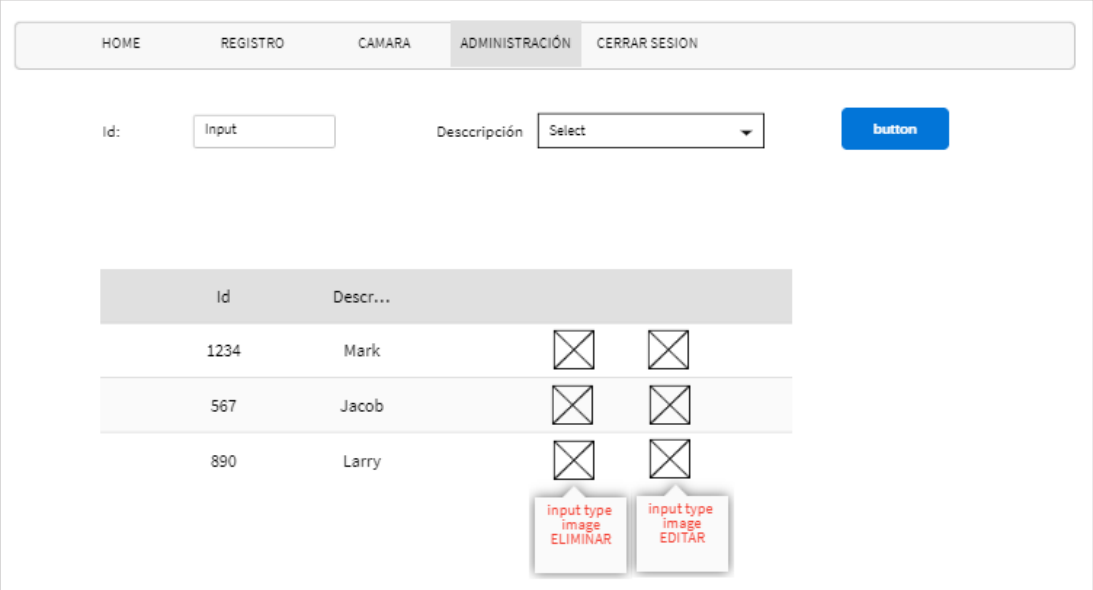


Figura 2.31. Sketch del formulario administrarCarrera

2.6.2.3 Diagramas de actividades

A continuación, se muestran los diagramas de actividades para cada formulario ya descrito anteriormente.

2.6.2.3.1. Formulario index

En la Figura 2.32 se presenta el diagrama de actividades para el formulario Login, el cual inicia evaluando las acciones: Ingresar o recuperar contraseña, cuando existe una acción de ingresar verifica que los campos estén llenos para posteriormente autenticar al administrador y abrir el formulario principal, cuando están vacíos los campos muestra un mensaje que se requieren dichos campos. De ser el caso de recuperar contraseña se verifica que haya un correo para posteriormente mediante Firebase Authentication enviar un link con los pasos a seguir para recuperar la contraseña, cuando no haya un correo, notifica que se requiere ese campo.

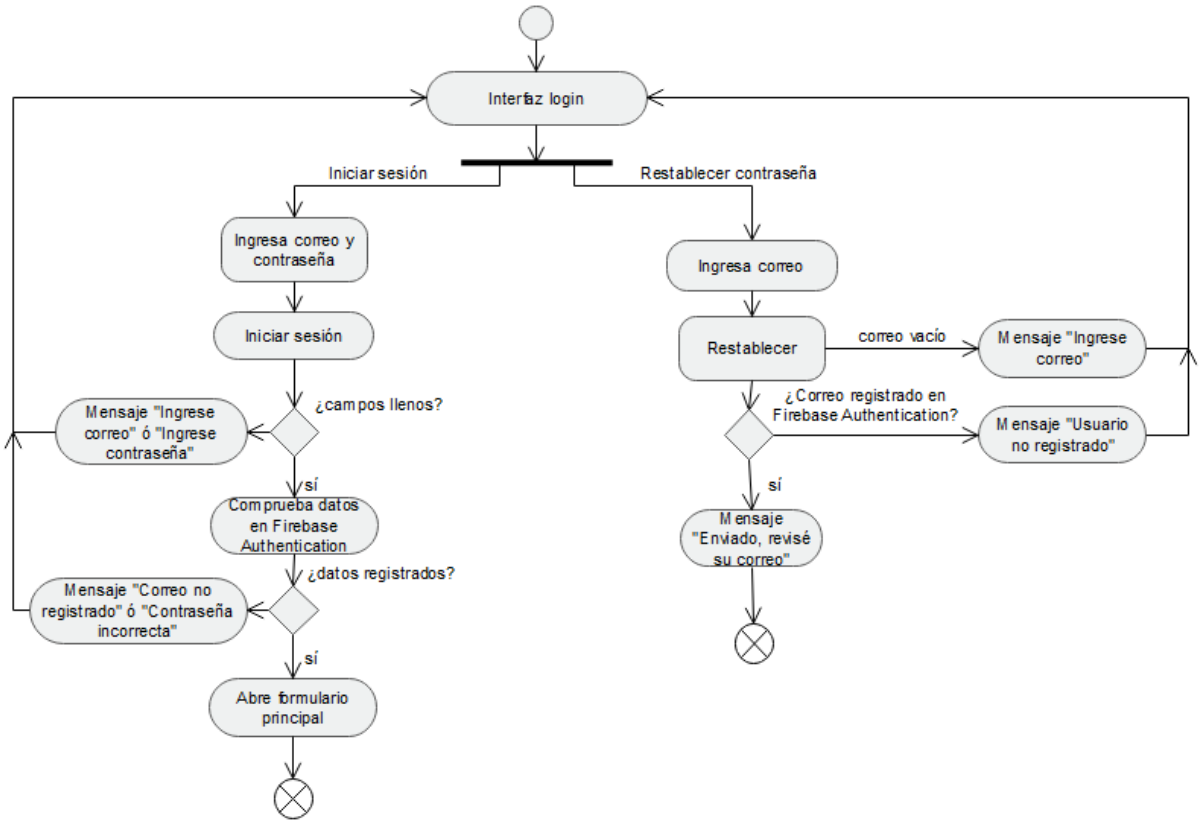


Figura 2.32. Diagrama de actividades para el formulario Login

2.6.2.3.2. Formulario principal

A continuación, en la Figura 2.33 se muestra el diagrama de actividades para el formulario principal, el cual inicia evaluando el valor que se selecciona para el campo línea para posteriormente consultar y mostrar la información correspondiente de esta unidad de transporte.

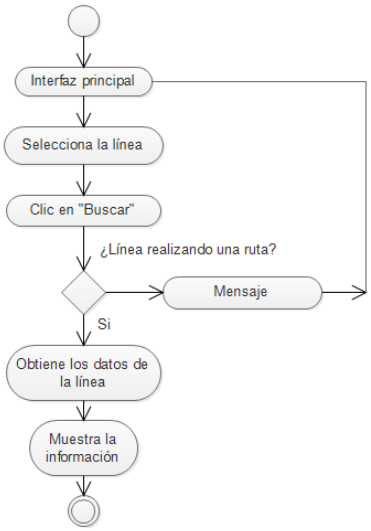


Figura 2.33. Diagrama de actividades para el formulario principal

2.6.2.3.3. Formulario registroEstudiante

La Figura 2.34 representa el diagrama de actividades para los formularios correspondiente a los registros de los estudiantes se iniciarán evaluando el valor seleccionado para el campo línea y selección del día para con ello consultar y mostrar los registros de asistencia de los estudiantes, posteriormente se tiene una evaluación de la acción exportar, la cual realiza el proceso de generar un archivo de formato PDF de todo este registro mostrado anteriormente.

Figura 2.34. Diagrama de actividades para el formulario registroEstudiante

2.6.2.3.4. Formulario registroChofer

A continuación, en la Figura 2.35 se tiene el diagrama de actividades para el formulario registroChofer, el cual inicia evaluando el valor seleccionado para el campo día luego consulta y muestra la información correspondiente a este registro. Así mismo se tiene la evaluación a la acción exportar PDF, con la que se realiza el proceso de obtener un archivo de formato PDF de este archivo.

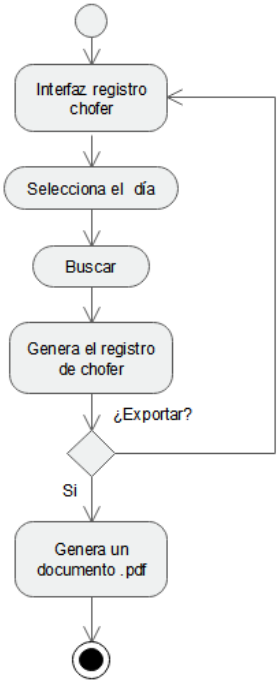


Figura 2.35. Diagrama de actividades para el formulario registroChofer

2.6.2.3.5. Formulario administrarUsuario

En la Figura 2.36 se presenta el diagrama de actividades para el formulario administrarUsuario, el mismo que inicia consultando la información actual de los usuarios registrados en el sistema, a la vez evalúa si ingresó una cédula y rol en los campos correspondientes, luego comprueba si el usuario a registrar es administrador, estudiante o chofer. En el caso de que es administrador se abre un modal y se llenan los campos para guardar la información de este usuario. Para el caso del estudiante y chofer el formulario envía las órdenes de encendido al módulo registro a través de la base de datos una vez se haya finalizado el registrado de huella, el formulario abre un modal para llenar la información correspondiente de cada usuario, luego esta información se guarda en la base de datos y el formulario está listo para realizar este proceso indefinidamente. Así mismo el formulario está evaluando la acción de editar o eliminar a un usuario, en el caso de editar,

el formulario abre un modal y carga la información correspondiente y para el caso de eliminar el formulario eliminar al usuario de la base de datos.

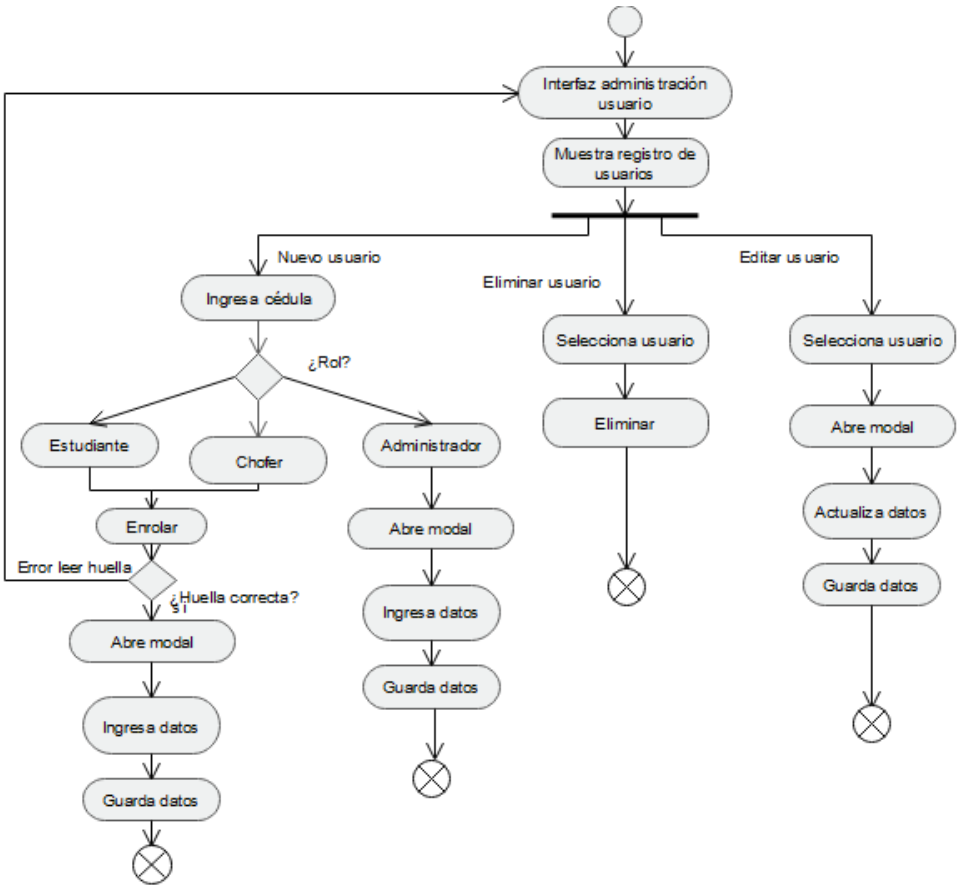


Figura 2.36. Diagrama de actividades del formulario administrarUsuario

2.6.2.3.6. Formulario administrarCarrera

En la Figura 2.37 se tiene el diagrama de actividades para el formulario administrarCarrera, el cual inicia mostrando la información actual de las carreras agregadas en el sistema, a la vez evalúa si existe una descripción para capturar esa información y guardarla en la base de datos. Así mismo el formulario evalúa la acción de editar o eliminar una carrera, cuando se realiza la acción de editar se captura la nueva información correspondiente a la carrera y para eliminar simplemente elimina esta carrera de la base y procede a actualizar la información de este formulario.

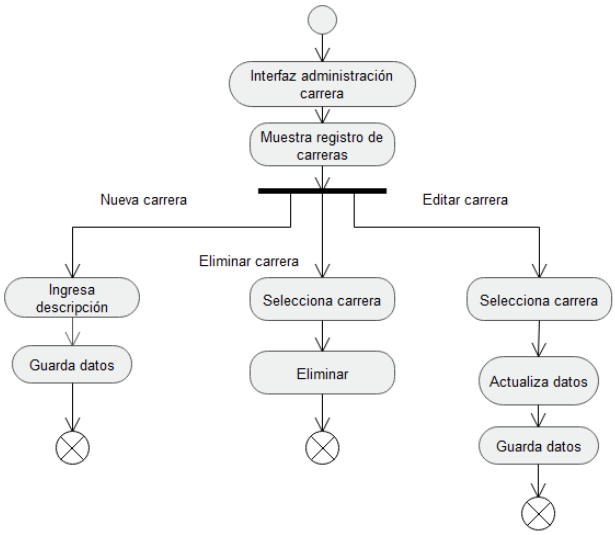


Figura 2.37. Diagrama de actividades del formulario administrarCarrera

2.6.2.3.7. Formulario administrarBus

A continuación, se presenta en la Figura 2.38 el diagrama de actividades para el formulario administrarBus, el cual inicia consultando y mostrando la información correspondiente a las unidades de transporte que son parte del sistema. Adicionalmente evalúa si se ha realizado la acción “nuevo” para abrir un modal, capturar y almacenar la información ingresada en el modal para agregar un nuevo bus, posteriormente actualizar la información correspondiente en la tabla. Este formulario también evalúa constantemente la acción editar y eliminar, en el caso de editar captura la nueva información por medio de un modal para posteriormente actualizar la base de datos y se actualiza la tabla, para el caso de eliminar, el formulario elimina y actualiza la base de datos.

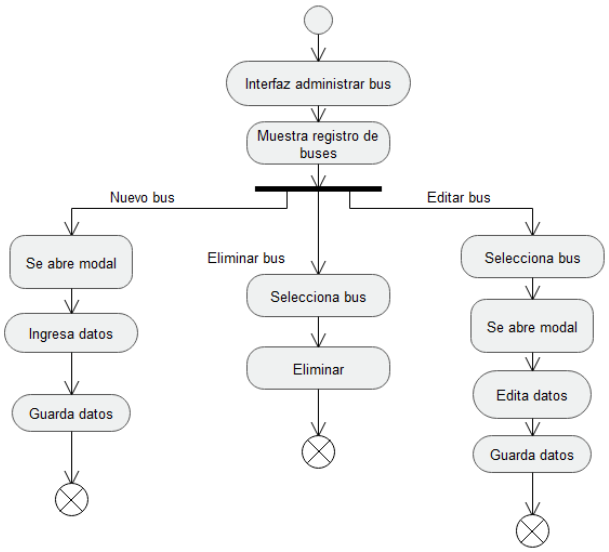


Figura 2.38. Diagrama de actividades del formulario administrarBus

2.6.2.4 Diagrama de flujo del módulo registro

En la Figura 2.39 se presenta el diagrama de flujo para el módulo registro, el cual inicia obteniendo las órdenes que serán enviadas desde el aplicativo web para encender o no al sensor biométrico, luego lee y válida la huella que por dos ocasiones será solicitada para así almacenarla en la base de datos y resetear las órdenes para que se apague el sensor hasta una nueva petición por parte del aplicativo web.

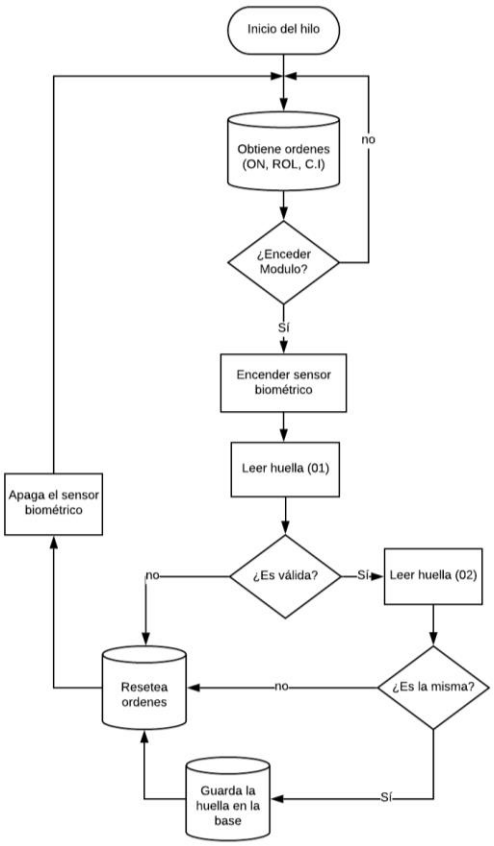


Figura 2.39. Diagrama de flujo del módulo registro

2.6.3. DISEÑO DEL APLICATIVO ANDROID

A continuación, en esta sección se describirá los diagramas de casos de uso del aplicativo móvil, además se visualizarán los diagramas de actividades y los sketches.

2.6.3.1 Diagrama de casos de uso

Debido a los requerimientos de la Tabla 2.23 se tiene para el aplicativo móvil dos actores: estudiante y administrador.

Cada requerimiento funcional definido para el aplicativo móvil es representado por un diagrama de caso de uso, por ello el título de cada diagrama corresponde al requerimiento

y a la vez los casos de uso se han clasificado dependiendo de los módulos definidos para el aplicativo.

Los casos de uso correspondientes para el **"Módulo Login"**, el cual permite iniciar sesión y recuperar contraseña, son los siguientes:

La Figura 2.40 muestra el caso de uso para iniciar sesión, donde tanto administrador como estudiante pueden iniciar sesión en la aplicación móvil.

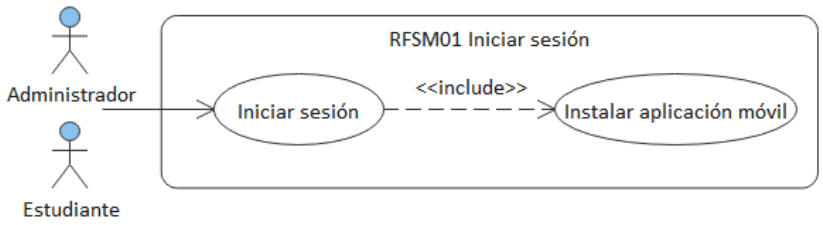


Figura 2.40. Caso de uso iniciar sesión

La Figura 2.41 muestra el caso de uso para recuperar contraseña, permite a los actores generar una nueva clave de ingreso a la aplicación móvil en caso de olvidarla.

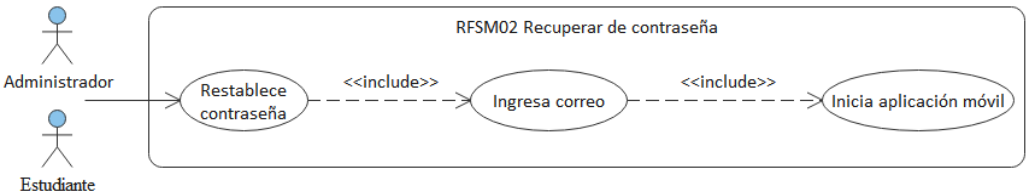


Figura 2.41. Caso de uso recuperar contraseña

El caso de uso para el **"Módulo Inicio",** permite identificar el rol del usuario logueado y es el siguiente:

La Figura 2.42 muestra el caso de uso para validar rol, permite identificar que actor inicia sesión para dirigirlo a su pantalla correspondiente.

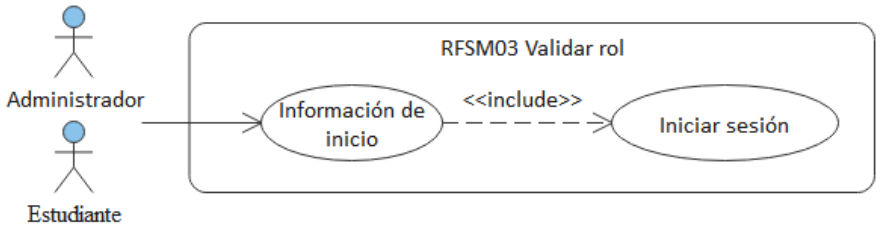


Figura 2.42. Validar rol

El caso de uso para el **"Módulo Información",** es el siguiente:

La Figura 2.43 muestra el caso de uso para visualizar información de su unidad de transporte, permite al actor estudiante ver información actual de la unidad a la cual se encuentra registrado.

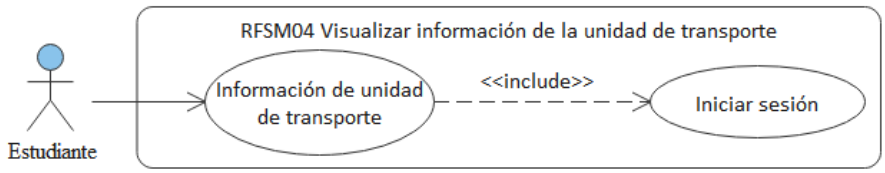


Figura 2.43. Caso de uso visualizar información de su unidad de transporte

El caso de uso para el **"Módulo Control",** es el siguiente:

La Figura 2.44 muestra el caso de uso para visualizar información de cada una de las unidades de transporte, permite al actor administrador poder conocer el estado actual de todas las unidades de transporte registradas.

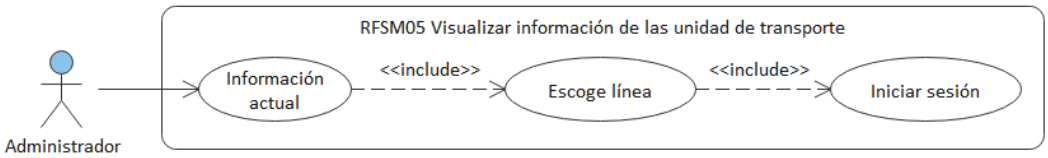


Figura 2.44. Caso de uso visualizar información de las unidades de transporte

El caso de uso para el "**Módulo ZoneMinder**", el cual permite visualizar las imágenes capturas por la cámara, es el siguiente:

La Figura 2.45 muestra el caso de uso para ingresar al ZoneMinder, permite al actor administrador ingresar al software ZoneMinder y poder visualizar cada una de las cámaras de las unidades de transporte registradas.

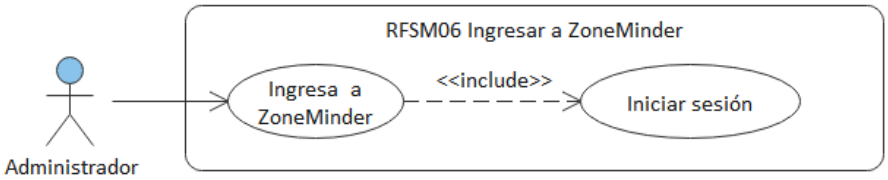


Figura 2.45. Caso de uso ingresar a ZoneMinder

El caso de uso para el "**Módulo Estudiantes**", el cual permite listar los estudiantes presentes es el siguiente:

La Figura 2.46 muestra el caso de uso para visualizar registro actual de estudiante, permite al actor administrador tener un registro actual de los estudiantes que se encuentran dentro de la unidad de transporte en ese momento en una ruta específica.

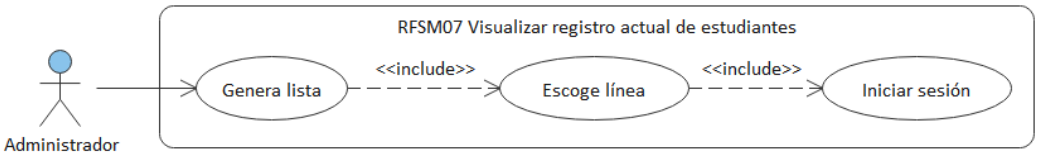


Figura 2.46. Caso de uso visualizar registro actual de estudiantes

Los casos de uso para el "**Módulo Emergencias",** el cual permite visualizar y eliminar las notificaciones de emergencia, son los siguientes:

La Figura 2.47 muestra los casos de uso para visualización de emergencias, permite al actor administrador y estudiante recibir notificaciones generadas desde la unidad de transporte en el caso de existir alguna emergencia y también permite al administrador eliminar las emergencias generadas.

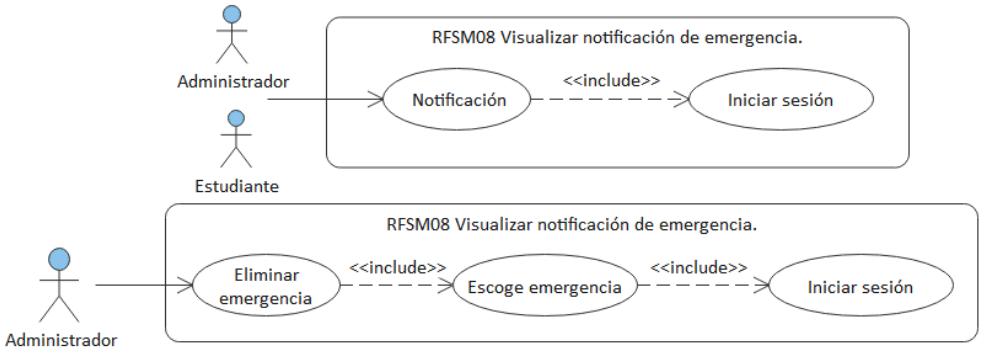


Figura 2.47. Visualizar notificaciones de emergencia

2.6.3.2 Diseño de los sketches

Para el diseño de los respectivos sketches para el aplicativo móvil se utilizó el software Lucid chart, este es un software de paga, pero permite tener una versión de prueba por un tiempo y/o número de trabajos (sketches) realizados, además, tiene la facilidad de trabajar directamente en la web o con una aplicación que se instala en IOS o en Android si es que así se quiere [38]. En la Figura 2.48 presenta la interfaz de Lucid chart.

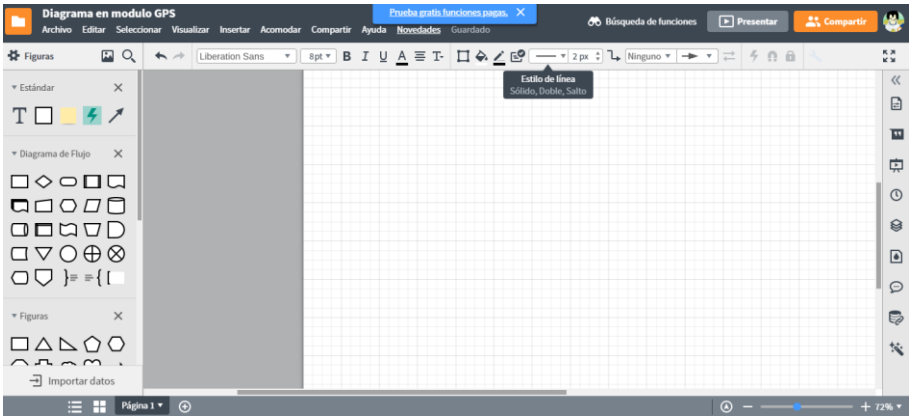


Figura 2.48. Interfaz de Lucid chart

En la Figura 2.49 se presenta el sketch correspondiente a la pantalla de inicio, esta pantalla válida si el usuario inició sesión en la aplicación para luego dirigirlo a la pantalla correspondiente, es decir, verifica si el usuario que se identificó es estudiante o administrador para así dirigirlo a la pantalla de información o control dependiendo de su rol. En el caso que no iniciar sesión, se pide al usuario identificarse mediante la pantalla Login. Todo este proceso lo realiza mediante internet, por lo que mientras realiza esta validación muestra las normas de uso del servicio de transporte estudiantil de la Escuela Politécnica Nacional.

Figura 2.49. Sketch de la pantalla inicio del aplicativo móvil

La pantalla Login valida mediante correo electrónico y contraseña a un usuario, ya sea administrador o estudiante (ver Figura 2.50). También permite al usuario recuperar su contraseña. Este proceso es ejecutado por el servicio de Firebase Authentication.

Figura 2.50. Sketch de la pantalla login del aplicativo móvil

En la Figura 2.51 se muestra el sketch de la pantalla información la cual es exclusiva para el uso del estudiante, esta indica la ubicación de la unidad de transporte a la que está registrado mediante el mapa de Google. Además, se podrá observar la información sobre el número de asientos disponibles que la unidad de transporte tiene actualmente y la fecha y hora en la que esta unidad inicio su ruta. En esta pantalla el estudiante también podrá cerrar su sesión si es que este así lo desea.

Figura 2.51. Sketch de la pantalla información del aplicativo móvil

A continuación, se presenta el sketch de la pantalla control (ver Figura 2.52), la misma que permite al administrador dar un seguimiento a cada una de las unidades de transporte. Esta pantalla tiene un spinner, el cual muestra la línea de la unidad de transporte y la descripción del mismo, para así facilitar al administrador la selección de a que unidad desea

dar un seguimiento. La información que presenta esta pantalla es: el número de asientos disponibles, la fecha y hora en la que la unidad de transporte inició la ruta. También cuenta con cuatro botones: el primer botón “VER” permite obtener la lista de estudiantes que están presentes en la unidad de transporte a través de la pantalla estudiantes, el segundo botón “VER” correspondiente al editText “Emergencias”, el cual abre la pantalla emergencias (ver Figura 2.53), la cual muestra el registro de emergencias ocurridas en la unidad de transporte seleccionada, el tercer botón “ELIMINAR” abre la misma pantalla anteriormente mencionada, con la diferencia que en este caso permitirá ver y eliminar una emergencia de la lista, finalmente el botón menú (tres puntos) permite cerrar la sesión y abrir la pantalla ZoneMinder.

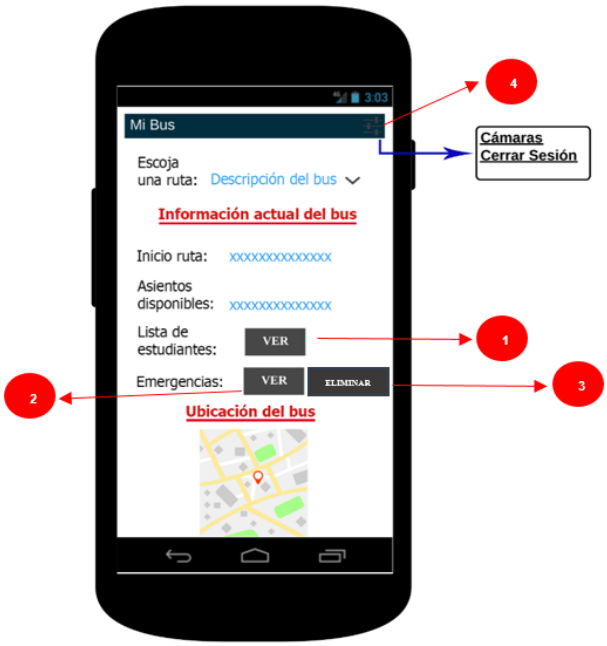


Figura 2.52. Sketch de la pantalla control del aplicativo móvil

Figura 2.53. Sketch de la pantalla emergencias del aplicativo móvil

En la Figura 2.54 se presenta el sketch para la pantalla estudiantes y se mostrará siempre y cuando el administrador haya pulsado el primer botón “VER” de la pantalla control (ver Figura 2.52). En esta pantalla el administrador podrá visualizar un registro actual de todos los estudiantes que están en la unidad de transporte que previamente fue seleccionada en la pantalla control.

Figura 2.54. Sketch de la pantalla estudiantes del aplicativo móvil

Para finalizar con el diseño de los sketches, en la Figura 2.55 se presenta el sketch de la pantalla ZoneMinder, en la cual el administrador tras pulsar el cuarto botón y seleccionar la opción cámaras en la pantalla control (ver Figura 2.52) podrá visualizar mediante un webView el interfaz del software ZoneMinder que se encuentra en Amazon Web Services.

Figura 2.55. Sketch de la pantalla ZoneMinder del aplicativo móvil

2.6.3.3 Diagramas de actividades

En esta sección se presentan los diagramas de actividades correspondientes a cada pantalla del aplicativo móvil y a la vez se describe brevemente cada uno.

2.6.3.3.1. Pantalla inicio

En la Figura 2.56 se presenta el diagrama de actividades para la pantalla inicio, el cual inicialmente verifica el inicio de sesión del usuario, si el usuario es estudiante inicia sesión abre la pantalla información, si el usuario es administrador abre la pantalla control y en el caso que el usuario no haya iniciado la sesión abre la pantalla login.

Esta pantalla es la principal, es decir siempre que se instale el aplicativo móvil o se abra la aplicación será la primera en iniciarse y realizar lo anteriormente expuesto.

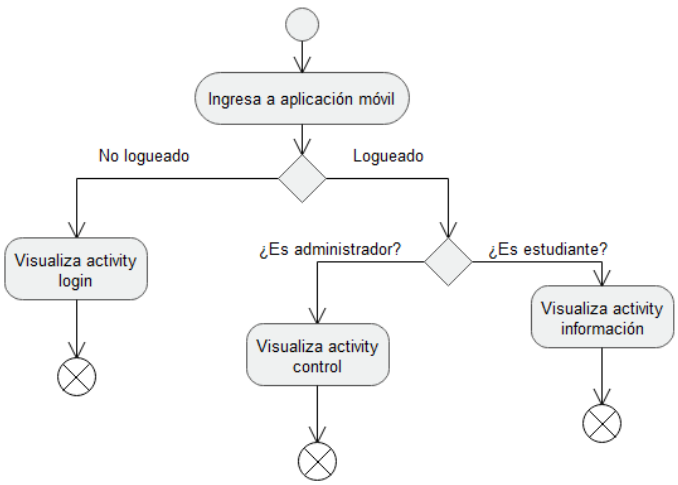


Figura 2.56. Diagrama de actividades de la pantalla inicio

2.6.3.3.2. Pantalla login

La Figura 2.57 presenta el diagrama de actividades para la pantalla login, que está a la espera de una acción como: ingresar o recuperar contraseña.

En el caso de inicio de sesión se comprueba que el usuario ingrese los campos requeridos (correo y contraseña), para posteriormente validar su identidad mediante el servicio de Firebase Authentication. Si el usuario está registrado, se evalúa si es estudiante o administrador, en el caso de sea estudiante abre la pantalla información y para el caso de administrador abre la pantalla control. Cuando se realiza la acción de recuperar contraseña se evalúa que el usuario ingrese su correo en el campo correspondiente, para así mediante Firebase Authentication enviar un link al correo anteriormente capturado con los pasos para restablecer su contraseña.

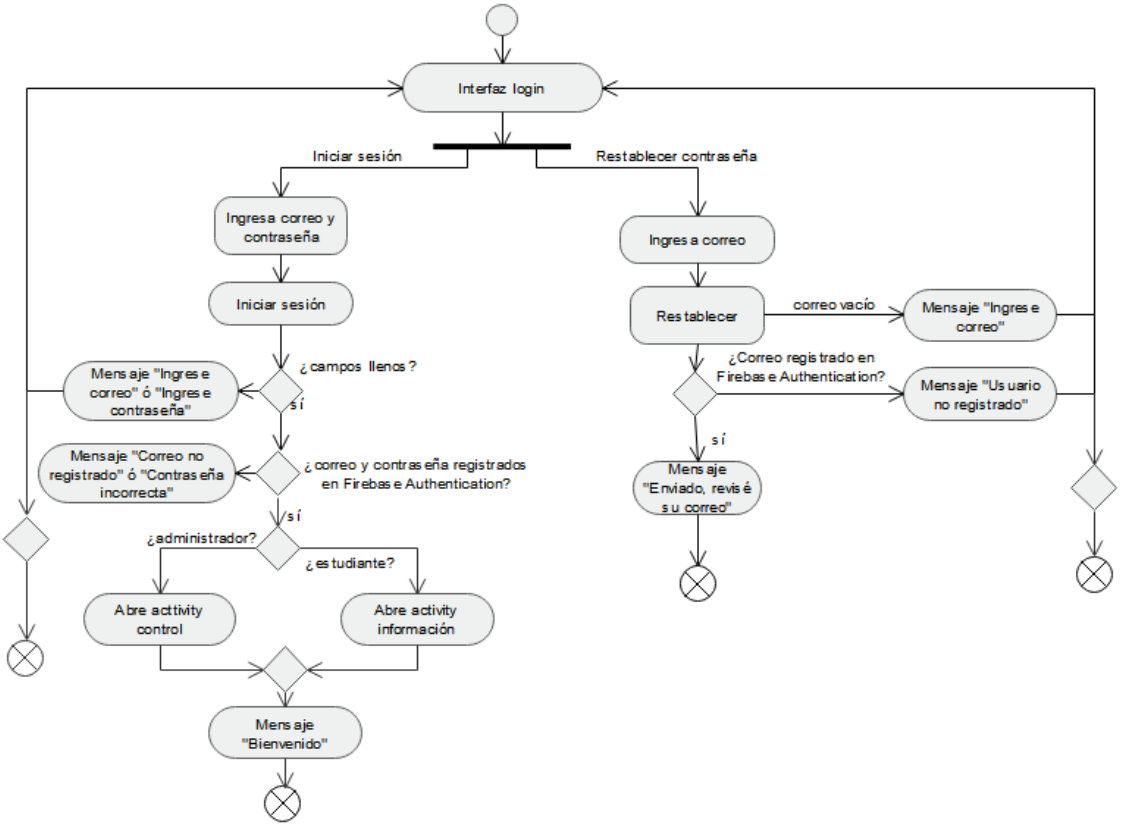


Figura 2.57. Diagrama de actividades de la pantalla login

2.6.3.3.3. Pantalla información

En la Figura 2.58 se muestra el diagrama de actividades correspondiente a la pantalla información, el mismo que inicia consultando en la base de datos Firebase y obtiene la información actual de la unidad de transporte, puesto que la información puede variar cada

instante, por ello es necesario mantener un bucle en la consulta para mantener la información actualizada.

A su vez la pantalla inicia un servicio, el cual opera en segundo plano para mostrar notificaciones cuando haya algún cambio en la base de datos, con la acción de cerrar sesión el estudiante regresa a la pantalla login y finalmente se muestra la acción de emergencia la cual permite visualizar las notificaciones de emergencia generadas por la unidad de transporte.

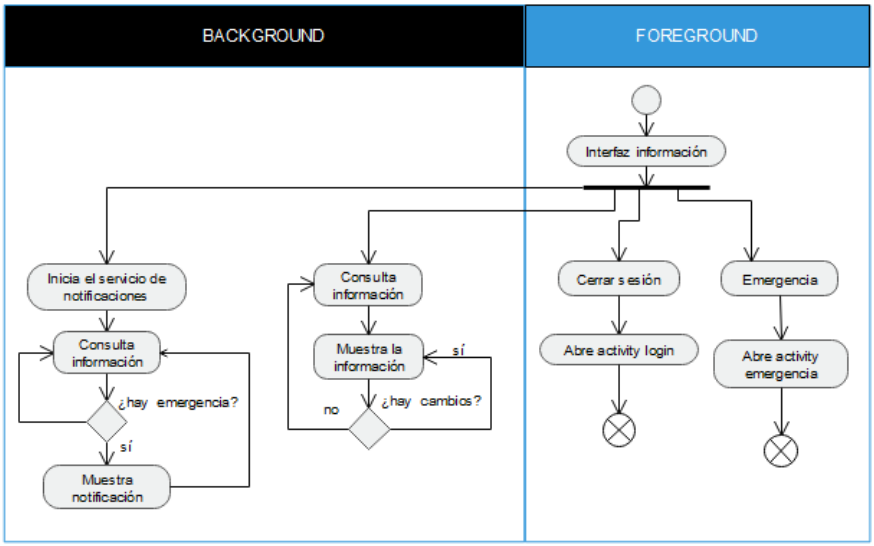


Figura 2.58. Diagrama de actividades de la pantalla información

2.6.3.3.4. Pantalla control

La siguiente Figura 2.59 muestra el diagrama de actividades de la pantalla control, la cual inicia consultando la información en la base de datos, posteriormente muestra esta información y entra a un bucle para mantener los datos siempre actualizados. Posteriormente se evalúa si se realizó las acciones de: “ver” (estudiantes), “ver” (emergencias), “eliminar” (emergencias), “ver cámaras” y “cerrar sesión”, dependiendo de la acción elegida se abrirá su activity o pantalla correspondiente, adicionalmente esta pantalla inicia el servicio de notificaciones.

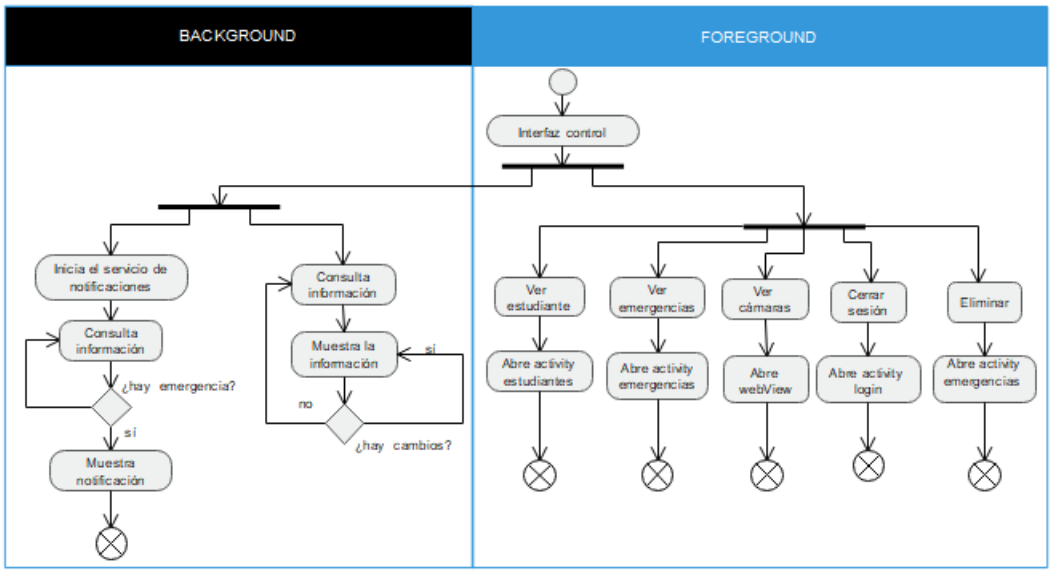


Figura 2.59. Diagrama de actividades de la pantalla control

2.6.3.3.5. Pantalla estudiante

A continuación, en la Figura 2.60 se presenta el diagrama de actividades para la pantalla estudiante, la cual inicia consultando la información en la base de datos para su posterior presentación, esta información es actualizada con cada cambio que suceda en la base por lo que entra a un bucle hasta que el administrador pulse el botón de cerrar sesión y con ello abra la pantalla login y así finalice este ciclo.

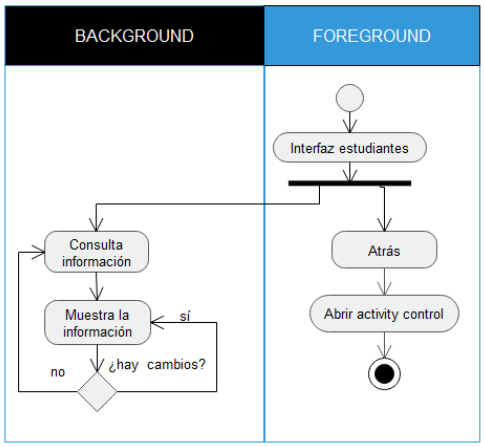


Figura 2.60. Diagrama de actividades de la pantalla estudiante

2.6.3.3.6. Pantalla emergencia

Para esta pantalla se tiene dos diagramas de actividades el primer es similar al diagrama de actividades de la pantalla estudiante, con la diferencia que son emergencias (ver Figura 2.60) debido a que se pulso el botón “ver” en la pantalla de control. El segundo diagrama

(ver Figura 2.61) presenta el proceso correspondiente a la acción del botón “eliminar” de la pantalla control.

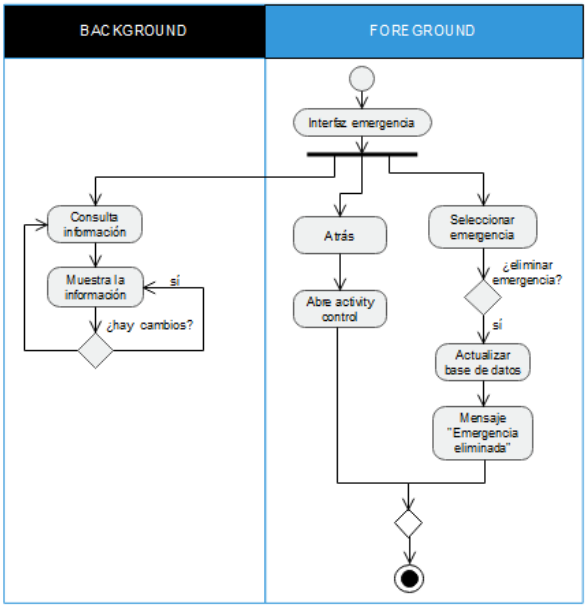


Figura 2.61. Diagrama de actividades de la pantalla emergencias

2.6.4. DIAGRAMA DE CLASES

A continuación, se presenta el diagrama de clases para el aplicativo web (ver Figura 2.62) y aplicativo móvil (ver Figura 2.63) dado que se tiene base de datos NoSQL con nodos independientes para almacenar la información enviada desde el sistema de control y aplicativo web. Sin embargo, la agregación, modificación y eliminación de estos nodos se realizan muchas veces a partir de la información previa que se obtiene de otros nodos. Por ejemplo, cuando un bus realiza una ruta, en el nodo principal RUTA se agregará un nodo secundario con la key “BUS_ID que corresponde a la MAC RPi” del nodo principal BUS y dentro de este nodo secundario se continuarán almacenando las rutas que se realicen por este BUS. Por lo tanto, se puede realizar un diagrama de clases que define la interacción de cada nodo (ver Figura 2.12).

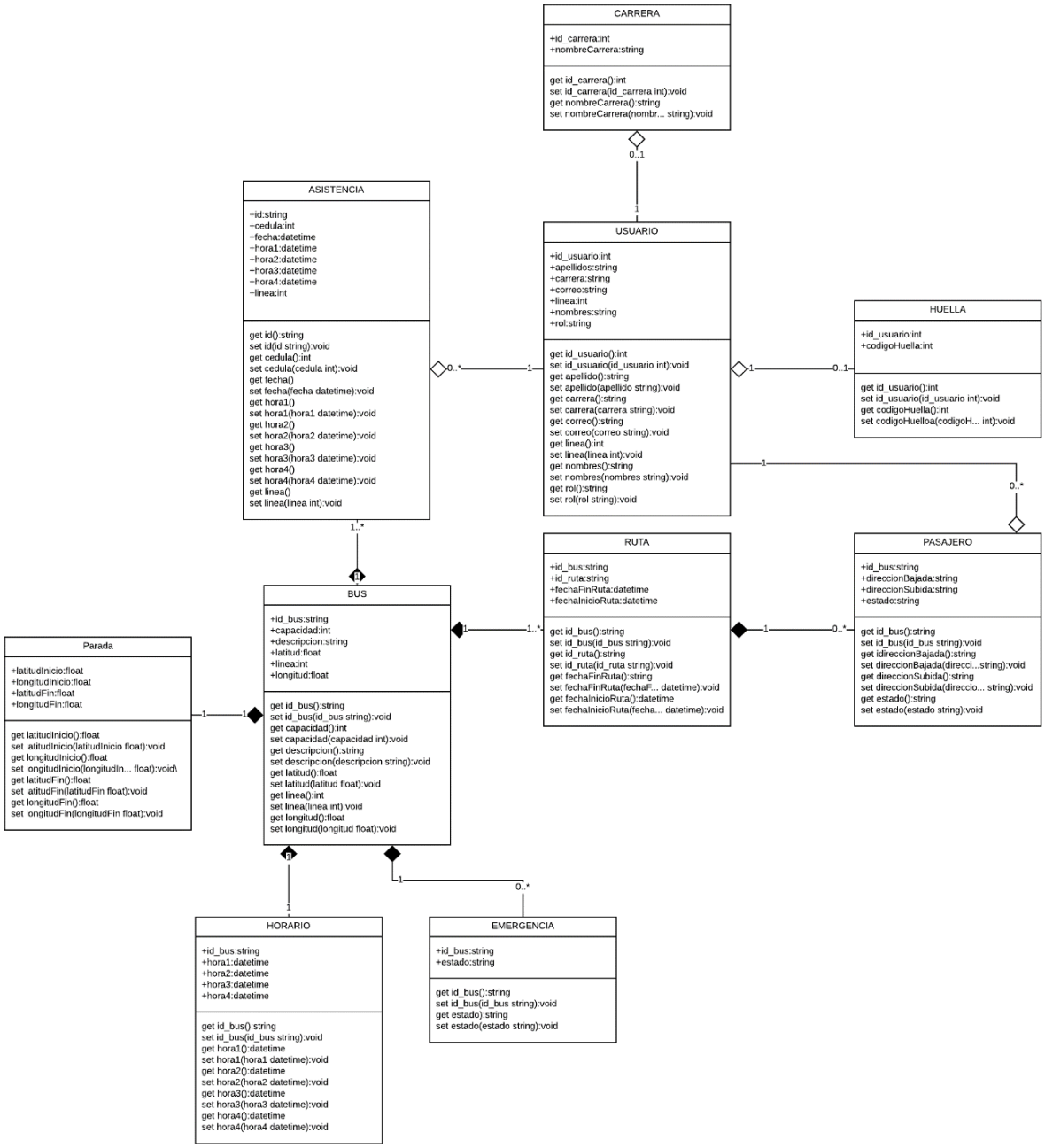


Figura 2.62. Diagrama de clases del aplicativo web

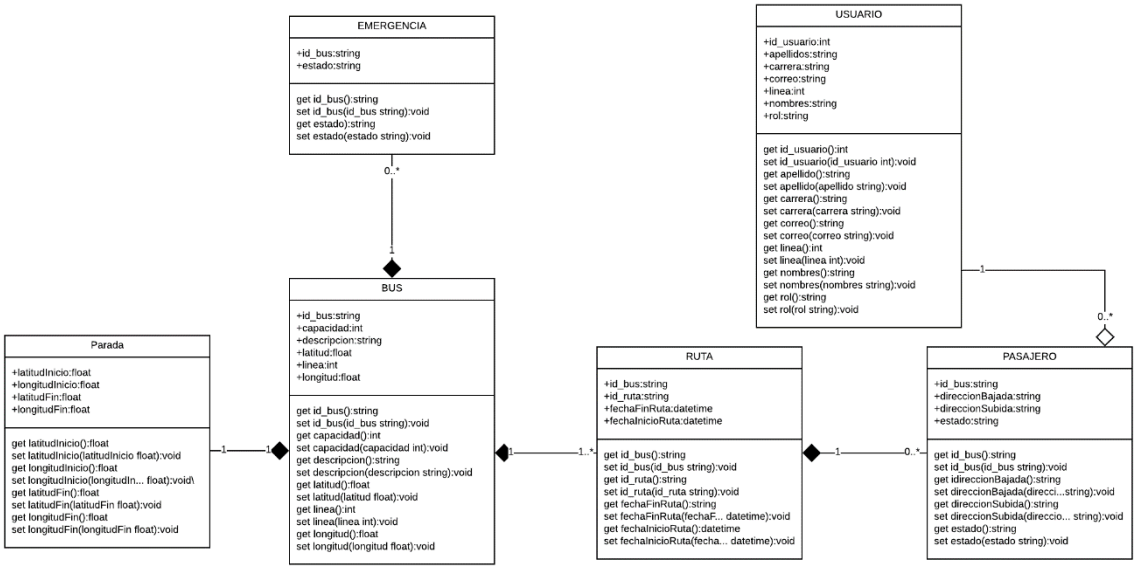


Figura 2.63. Diagrama de clases del aplicativo móvil

3. IMPLEMENTACIÓN

Este capítulo describe la implementación realizada para cada uno de los sistemas que forman parte del prototipo planteado. Se presentará la codificación más relevante, la instalación de componentes necesarios, tanto para los aplicativos como para los módulos que intervienen. Para el caso de los módulos se presentará adicionalmente la conexión realizada hacia la RPi.

3.1. ACTUALIZACIÓN DEL TABLERO KANBAN EN LA FASE DE IMPLEMENTACIÓN

En la Figura 3.1 se puede visualizar la actualización del tablero Kanban, por lo que se puede notar el avance y la realización de las tareas correspondientes al prototipo. Además, se debe destacar que las historias de usuario fueron tomadas en cuenta para la realización de las distintas tareas presentes en el tablero.

Figura 3.1. Tablero Kanban de la fase de implementación

3.2. IMPLEMENTACION DEL SISTEMA DE CONTROL

En esta sección, además, de la instalación y configuración de los módulos pertenecientes al sistema de control también se detalla la configuración del internet y creación de los servicios.

3.2.1. CONFIGURACIÓN PARA EL ACCESO A INTERNET

A la unidad remota se le agregó un modem USB Huawei E303 que permitirá la conexión a internet a través de la red celular. A continuación, se presenta la instalación y configuración de los requerimientos para que la RPi tenga conexión.

Mediante el Comando 3.1 se instala el servicio *pppd* y a la vez los programas *wvdial* y *usb-modeswitch.*

```
pi@raspberrypi:~ $ sudo apt install ppp usb-modeswitch wvdial
```

Comando 3.1. Instalación de servicio y programas para el acceso a internet

Una vez terminada la instalación, se necesita configurar el programa wvdial con el Comando 3.2. Para esta configuración es necesario que el modem tenga un chip de una operadora celular en la ranura SIM y con ello colocar el APN correspondiente de la operadora en el archivo de configuración *etc/wvdial.conf* (ver Figura 3.1).

```
pi@raspberrypi:~ $ sudo nano /etc/wvdial.conf
```

Comando 3.2. Editar el archivo wvdial.conf

```
 GNU nano 2.7.4                File: /etc/wvdial.conf

[Dialer Defaults]
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0
Init3 = AT+CGDCONT=1,"IP","internet.cnt.net.ec"
Stupid Mode = 1
Modem Type = Analog Modem
Baud =9600
New PPPD = yes
Modem = /dev/ttyUSB2
ISDN=0
Phone =*99#
Username = { }
Password = { }
```

Figura 3.2. Configuración del programa wvdial con operadora CNT

Una vez realizada la configuración de wvdial se ejecuta el Comando 3.3, de esta manera se activa el acceso a internet. Una forma de comprobar la conexión a internet es identificar las interfaces de red de la RPi mediante el Comando 3.4, con el cual se muestran las interfaces de red de la RPi con sus respectivas direcciones IP. En la Figura 3.3 se puede apreciar que la interfaz ppp0 tiene asignada una dirección IP. Por lo que se concluye que la RPi tiene acceso a la red celular.

```
pi@raspberrypi:~ $ sudo wvdial
```

Comando 3.3. Activar acceso a internet

```
pi@raspberrypi:~ $ ifconfig
```

Comando 3.4. Muestra las interfaces de red RPi

```
ppp0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST>  mtu 1500
        inet 10.60.247.221  netmask 255.255.255.255  destination 10.64.64.64
        ppp  txqueuelen 3  (Point-to-Point Protocol)
        RX packets 12  bytes 198 (198.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13  bytes 273 (273.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figura 3.3. Asignación de IP a la interfaz ppp0

El funcionamiento del prototipo está comprometido netamente por la conexión a internet, por lo que en este punto se revisó la cobertura que ofrece MOVISTAR, CLARO y CNT (ver ANEXO C), de las cuales se determinó que la operadora que brinda mejor cobertura para el prototipo es CLARO, por lo que en el archivo de configuración wvdial.conf el APN es el correspondiente a esta operadora (ver Figura 3.4).

```
  GNU nano 2.7.4                File: /etc/wvdial.conf

[Dialer Defaults]
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0
Init3 = AT+CGDCONT=1,"IP","internet.claro.com.ec"
Stupid Mode = 1
Modem Type = Analog Modem
Baud =9600
New PPPD = yes
Modem = /dev/ttyUSB2
ISDN=0
Phone =*99#
Username = { }
Password = { }
```

Figura 3.4. Configuración del programa wvdial con operadora CLARO

3.2.2. CODIFICACIÓN HILOS

Para una codificación más eficiente para el prototipo se crearon varios hilos. El script main.py (ver ANEXO D) tiene varios hilos, los cuales fueron codificados para cada módulo que forman parte de la unidad remota, con ello se garantiza que los procesos que realiza cada módulo sean más rápidos e independientes.

En la Tabla 3.1 se presenta un resumen de los hilos usados en el script main.py.

Tabla 3.1. Descripción de hilos

MÓDULO	NOMENCLATURA HILO
GPS	hilo_obtener_ubicación
Huella	hilo_huella, hilo_zona
Control electrónico	hilo_pulsadores

En el Código 3.1 se muestra un ejemplo de la creación e iniciación de un hilo, el cual es ejecutado por un hilo principal, del que se puede observar que el método que ejecuta el *hilo_pulsadores* es *estadoPulsador()*.

```
#Inicio del hilo de los pulsadores
hilo_pulsadores =threading.Thread(target=estadoPulsador)
hilo_pulsadores.start()
```

Código 3.1. Creación e inicio de un hilo

3.2.3. IMPLEMENTACIÓN DEL MÓDULO GPS

En esta sección se presenta la habilitación del puerto serial, la conexión del módulo, la instalación de la librería pynmea2 y la codificación necesaria para que este módulo realice su función.

3.2.3.1 Configuración del puerto UART en la RPi

En esta versión de RPi, el Bluetooth está en los pines 8 y 10 (ttyAMA0), los mismos que son de alto rendimiento y para comunicación serial en los pines 14 y 15 (ttyS0). Debido a que el prototipo no necesita de Bluetooth se ha configurado la comunicación serial en los pines 8 y 10, de esta manera el módulo GPS tendrá una mejor conectividad con la RPi. Para conseguir esto se necesita editar el archivo ***config.txt*** con el Comando 3.5.

```
pi@raspberrypi:~ $ sudo nano /boot/config.txt
```

Comando 3.5. Editar archivo config.txt

Dentro de este archivo se necesita agregar las siguientes líneas (ver Figura 3.5), la primera línea habilita el puerto serial, mientras que la segunda realiza un ajuste en la frecuencia del núcleo del CPU de la RPi para el sistema operativo Stretch-Raspbian. Finalmente, la última línea desactiva el Bluetooth, con ello se garantiza un alto rendimiento en estos pines, ahorro de recursos de CPU y memoria para la RPi.

```
enable_uart=1
core_freq=250
dtoverlay=pi3-disable-bt
```

Figura 3.5. Líneas que se agregan en el archivo

Para que la configuración se realice de manera exitosa, se necesita hacer un reboot a la RPi. Posteriormente se necesita desactivar la consola (ver Comando 3.6).

```
pi@raspberrypi:~ $ sudo systemctl stop serial-getty@ttyAMA0.service
pi@raspberrypi:~ $ sudo systemctl disable serial-getty@ttyAMA0.service
```

Comando 3.6. Desactivar consola

Además, se necesita el editar el archivo **cmdline.txt** (ver Comando 3.7):

```
pi@raspberrypi:~ $ sudo nano /boot/cmdline.txt
```

Comando 3.7. Editar el archivo cmdline.txt

Una vez dentro del archivo, se necesita cambiar la línea de código por la que se presente en la Figura 3.6:

```
dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait
```

Figura 3.6. Línea de código editada

Finalmente se deshabilita el servicio HCIUART con el Comando 3.8:

```
pi@raspberrypi:~ $ sudo systemctl disable hciuart
```

Comando 3.8. Deshabilitar servicio HCIUART

3.2.3.2 Conexión para el módulo GPS

A continuación, se indica la conexión realizada del módulo GPS hacia los pines de la RPi (ver Figura 3.7).

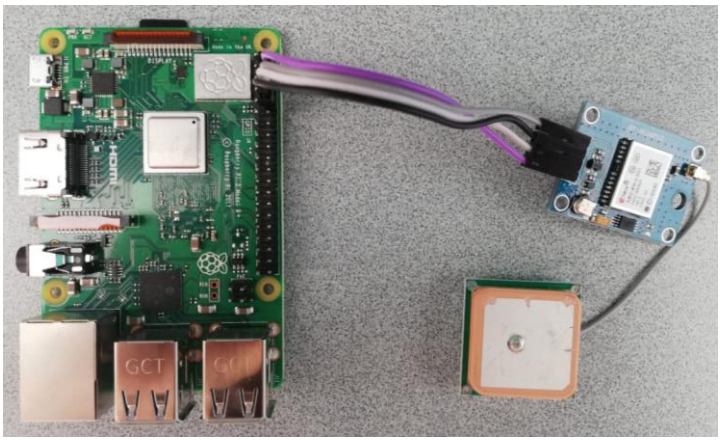


Figura 3.7. Conexión GPS a RPi

En la Tabla 3.2 se detalla la conexión realizada.

Tabla 3.2. Pines de conexión RPi a GPS

Raspberry Pi 3 (Pines)	Ublox Neo 6M (Pines)
2 - 5V Power	5V
9 – GND	GND
8 - UART0_Tx	Rx
10 - UART_Rx	Tx

Luego de conectar el módulo GPS, el siguiente paso es realizar una prueba para comprobar que la configuración fue exitosa. Para ello leemos el puerto serial ttyAMA0 y capturamos su contenido con el Comando 3.9.

```
pi@raspberrypi:~ $ sudo cat /dev/ttyAMA0
```

Comando 3.9. Captura del contenido del puerto serial

Luego de ejecutar el comando, el módulo GPS muestra las tramas en formato NMEA (ver Figura 3.8), por tanto, la configuración fue correcta.

```
$GNGGA,201915.374,,,,,0,00,25.5,,,,,,*74
$GNGLL,,,,,201915.374,V,M*69
$GPGSA,A,1,,,,,,,,,,,,,25.5,25.5,25.5*02
$BDGSA,A,1,,,,,,,,,,,,,25.5,25.5,25.5*13
$GPGSV,1,1,03,12,,,26,25,,,28,29,,,33*7B
$BDGSV,1,1,01,12,,,28*60
$GNRMC,201915.374,V,,,,,,,161019,,,M*50
$GNVTG,,,,,,,,,M*2D
$GNZDA,201915.374,16,10,2019,00,00*4A
$GPTXT,01,01,01,ANTENNA OK*35
```

Figura 3.8. Trama en formato NMEA

3.2.3.3 Codificación para el módulo GPS

Para realizar la codificación de los métodos, primero se deben instalar las librerías pynmea2 y geopy desde la terminal de la RPi con los comandos *pip install pynmea2* y *pip install geopy*, para luego ser añadidas en el espacio del Código 3.2, del script.

```
#Librerias necesarias para el modulo GPS
import pynmea2
from geopy.geocoders import Nominatim
```

Código 3.2. Librerías utilizadas para el módulo GPS

A continuación, se presenta la codificación de los métodos que forman parte del hilo "***hilo_obtener_ubicacion***". El primer método *obtenerUbicacion()* del Código 3.3 lee las líneas de datos que obtiene el módulo GPS, conectado al puerto serial (ttyAMA0). Cada línea de dato en formato NMEA es leída por el método *readline(),* posteriormente son guardadas en la variable "*lineaSerial*". Para el caso del prototipo solo es necesaria la línea GNGGA, una vez ubicada esta línea dentro del código se procede a obtener las coordenadas de latitud y longitud con la ayuda de la librería pynmea2.

```
#Metodo para obtener la ubicacion del modulo GPS
def obtenerUbicacion():
    global latitud,longitud
    while(selector=="on"):
        try:
            #Lee el puerto serial
            lineaSerial =modulo_gps.readline()
                except:
            print("\n-----------------------------------------------\n")
            print("ERROR AL LEER EL SERIAL")
                        print("\n-----------------------------------------------\n")
            #Busca la linea $GNGGA
            while (lineaSerial[0:6] != '$GNGGA'):
                        lineaSerial = modulo_gps.readline()
                        esperar(1)
                        if (selector=="off"):
                                print("\n-----------------------------------------------\n")
                                print("PARANDO UBICACION")
                                print("\n-----------------------------------------------\n")
                                exit(1)
                        pass
            coordenadas = pynmea2.parse(lineaSerial)
            longitud=coordenadas.longitude
            latitud=coordenadas.latitude
```

Código 3.3. Método obtenerUbicación()

En el Código 3.4 se presenta el método *guardarUbicacion()* que permite guardar los datos obtenidos. Dentro de este método primero se debe referenciar la ubicación del nodo *('/BUS/'+ str (mac)),* BUS corresponde al nombre del nodo principal y *str(mac)* permite identificar a que bus le corresponden las coordenadas en el caso de que existan más buses en el nodo, por último se envían los datos a la base de datos con el método *update*, puesto que el bus va a estar en movimiento y estos datos van a ir variando, cada que se genere un cambio.

```
#Metodo para guardar la ubicacion del bus en Firebase
def guardarUbicacion():
    global referenciaUbicacion
    try:
        referenciaUbicacion = db.reference('/BUS/'+str(mac))
        guardar=referenciaUbicacion.update({'latitud':latitud,'longitud':longitud })
    except:
```

Código 3.4. Método guardarUbicación()

3.2.4. IMPLEMENTACIÓN DEL MÓDULO HUELLA

En esta sección se presenta la conexión y codificación correspondiente al módulo huella.

3.2.4.1 Conexión del módulo huella

En la Figura 3.9, se muestra la conexión realizada para el sensor biométrico, debido a que el puerto UART de la RPi ya está siendo utilizado por el módulo GPS, se optó por utilizar un conversor UART-USB para la conexión del módulo con la RPi.

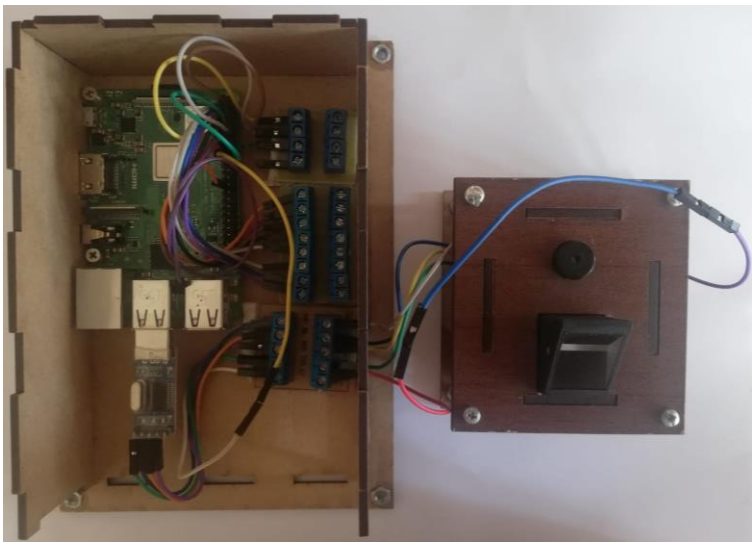


Figura 3.9. Conexión módulo huella

En la Tabla 3.3 se indica los puertos y pines que se utilizaron para conectar el módulo huella hacia el conversor y a la RPi.

Tabla 3.3. Conexión del módulo huella con los diferentes elementos que intervienen

MÓDULO HUELLA			CONVERSOR	RASPBERRY
Estudiante / Chofer (Control de acceso / Asistencia)	Sensor biométrico	Vcc	5V	USB0
		Tx	Rx	
		Rx	Tx	
		GND	GND	
	Bocina		-----	Pin 12

3.2.4.2 Codificación para el módulo huella

A continuación, se presenta el código (ver Código 3.5) necesario para la implementación del módulo huella, el cual provee los métodos:

- readImage(): Evalúa si una huella está colocada en el lente óptico del sensor biométrico y devuelve un valor True o False.
- convertImage(): Captura la imagen de la huella para obtener las características de la misma y luego las almacena en un espacio de memoria.
- uploadCharacteristics(): Carga las características de una huella en un espacio de memoria.

- compareCharacteristics():Compara las características de una huella con otra.

```
from pyfingerprint.pyfingerprint import PyFingerprint
```

Código 3.5. Módulo pyfingerprint

Como se mencionó anteriormente el módulo huella consta de un sensor biométrico, el mismo que es codificado mediante el hilo *hilo_huella* (ver Código 3.6).

```
#Inicio del hilo para el modulo huella
hilo_huella= threading.Thread(target=leer_huella)
hilo_huella.start()
```

Código 3.6. Hilos del módulo huella

Para este módulo se codificaron varios métodos, entre ellos el primero en mencionar es *leerHuella(),* método que permite evaluar las características de la huella que está colocada en el lente óptico del sensor, las cuales se almacenan temporalmente en un espacio de memoria (0x01) junto con las características de cada una de las huellas que se encuentran almacenadas en la base de datos (ver Código 3.7). Por lo que se realiza una consulta para importar todas las características de las huellas y una a una se las carga en un espacio de memoria (0x02) con el fin de compararlas con respecto a la que se mantiene en el espacio de memoria 0x01, si la comparación anteriormente mencionada es acertada se obtendrá la cédula del usuario, valor que será importante para las posteriores codificaciones de los métodos correspondientes a este módulo.

```
#Lee la huella del sensor
modulo_estudiante.convertImage(0x01)
on_off_bocina(0.6,12)
#Obtiene las huellas
referencia = db.reference("/HUELLA")
huellas= referencia.order_by_child("CODIGOHUELLA").get()
for key in huellas:
    ref = db.reference()
    baseHuella=ref.child("HUELLA").child(str(key)).child('CODIGOHUELLA')
    huellaBase=baseHuella.get()
    #Compara las huella de la base con la huella que estuvo en el sensor
    modulo_estudiante.uploadCharacteristics(0x02,huellaBase)
    if ( modulo_estudiante.compareCharacteristics() == 0 ):
        opcionHuella=0
    else:
        #Obtiene la cedula de esa huella
        opcionHuella=1
        cedula=key
        break
#Compara el rol del usuario
if(opcionHuella==1):
    resultado = modulo_estudiante.searchTemplate()
    idHuella = resultado[0]
    fechaP=datetime.now()
    comprobarRol(fechaP)
else:
    print("\n----------------------------------------------\n")
    print("NO ESTA REGISTRADO")
    print("\n----------------------------------------------\n")
```

Código 3.7. Código leerHuella

Además, se codificaron los métodos *comprobarRol(), guardarAsistencia()* y *guardarPasajero(),* los cuales almacenan en la base de datos información de asistencia del chofer y estudiante correspondientemente después de verificar su rol en él sistema.

3.2.5. IMPLEMENTACIÓN DEL MÓDULO CONTROL ELECTRÓNICO

En esta sección se presenta la conexión y codificación involucrada para el funcionamiento del módulo de control electrónico.

3.2.5.1 Conexión módulo de control electrónico

En la Figura 3.10 se presenta la conexión del módulo de control electrónico, se puede observar la integración de todos los componentes electrónicos (leds, pulsadores, que hacen parte de este módulo.

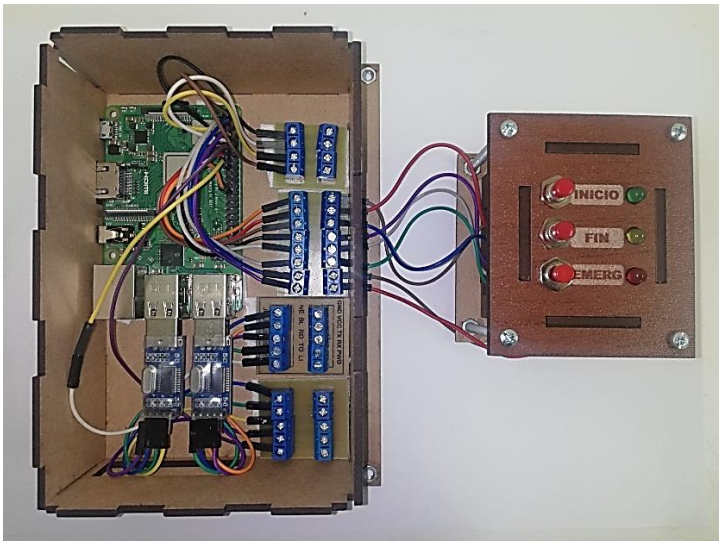


Figura 3.10. Conexión módulo de control electrónico

En la Tabla 3.4 se detalla los pines utilizados para conectar el módulo a la RPi.

Tabla 3.4. Pines utilizados para la conexión del módulo de control electrónico

Módulo de control electrónico.		**Raspberry Pi 3 (Pines)**
VCC		4 – 5V
GND		6 – GND
PULSADORES	Inicio ruta.	3 – GPIO
	Fin ruta.	5 – GPIO
	Emergencia.	7 – GPIO
LEDS	Verde - Inicio ruta.	11 – GPIO
	Amarillo - Fin ruta.	13 – GPIO
	Rojo - Emergencia.	23 – GPIO

3.2.5.2 Codificación del módulo de control electrónico

A continuación, se describirá la codificación más relevante utilizada para este módulo. Antes de empezar a codificar se debe importar la siguiente librería (ver Código 3.8), la cual permitirá agregar operaciones trigonométricas, posteriormente se indicará el fragmento de código donde se utilizó este módulo.

```
import math
```

Código 3.8. Módulo para la codificación del módulo control electrónico

El método *estadoPulsador()*, se ejecuta con el hilo *hilo_pulsadores* (ver Código 3.1) estará constantemente detectando el estado de cada uno de los pulsadores del módulo control electrónico, dependiendo del estado en el que se encuentren se definirán los procesos que se deben seguir.

El Código 3.9 que se encuentra dentro del método antes mencionado permite examinar el estado en el que se encuentran los pulsadores estos son: “si ha sido presionado” o “no ha sido presionado”, también se debe mencionar que los leds o indicadores se deben prender o apagar dependiendo del estado en el que se encuentren, por ello en el Código 3.10 se indica la codificación para encender o apagar un led que esté conectado a un pin de la RPi.

```
#Captura el estado de los pulsadores
estadoIR=gpio.input(3)
estadoFR=gpio.input(5)
estadoE=gpio.input(7)
```

Código 3.9. Captura del estado de los pulsadores

```
#Activa/Desactiva indicador led
gpio.output(11,True)
indicadorIR=gpio.input(11)
gpio.output(13,False)
```

Código 3.10. Prender o apagar un led

A continuación, se detallarán los principales procesos a seguir, dependiendo del estado en el que se encuentren los pulsadores del módulo.

- Cuando pulsador INICIO RUTA y pulsador FIN RUTA *“han sido presionados”*: primero se debe analizar que el bus se encuentre cerca del punto de parada final o parada inicial caso contrario no se podrá activar el INICIO RUTA o FIN RUTA, por

tal razón se definió un método llamado *distancia(),* este método lee los valores de las coordenadas de latitud y longitud almacenadas en la base: de las ramificaciones latitudInicio, longitudInicio, latitudFin y longitudFin, que se encuentran en el nodo PARADA las mismas que son constantes y las coordenadas de longitud y latitud que van actualizando dependiendo del movimiento del bus. Todas las coordenadas antes expuestas se encuentran dentro del nodo principal BUS, cada coordenada tiene diferente referencia como se puede apreciar en el Código 3.11.

```
#Se extrae de la base las coordenadas de paradaInicio
latitudInicioBase=db.reference('/BUS/'+str(mac)+'/PARADA/latitudInicio').get()
longitudInicioBase=db.reference('/BUS/'+str(mac)+'/PARADA/longitudInicio').get()
#Se extrae de la base las coordenadas de paradaFin
latitudFinBase=db.reference('/BUS/'+str(mac)+'/PARADA/latitudFin').get()
longitudFinBase=db.reference('/BUS/'+str(mac)+'/PARADA/longitudFin').get()
#Se extrae de la base las coordenadas actuales
latitudActualBase=db.reference('/BUS/'+str(mac)+'/latitud').get()
longitudActualBase=db.reference('/BUS/'+str(mac)+'/longitud').get()
```

Código 3.11. Referencias a la base de datos de latitudes y longitudes

Una vez obtenidos los valores antes mencionados se procede a aplicar la ecuación de Haversine (3.1), la misma que obtiene la distancia entre dos puntos usando como datos las coordenadas de latitud y longitud, examinando la curvatura de la Tierra [39], esta ecuación trigonométrica se expresa de la siguiente manera:

$$Distancia\ = 2R\ arcsin\left(\sqrt{sin^2\left(\frac{lat2-lat1}{2}\right)\cos(lat1)*\cos(lat2)*sin^2\left(\frac{lon2-lon1}{2}\right)}\right) \quad (3.1)$$

Donde:

R: radio de la Tierra.

lat2, lon2: latitud y longitud del punto 2.

Lat1, lat1: latitud y longitud del punto 1.

En el Código 3.12 se puede observar la codificación de la ecuación de Haversine.

```
#Calculos matemeticos
rad=math.pi/180
R=6372.795477598 #Radio de la tierra
#Diferencia entre paradaInicio-ubicacionActual [km]
dlat1=latitudInicioBase-latitudActualBase
dlong1=longitudInicioBase-longitudActualBase
a1=(math.sin(rad*dlat1/2))**2 +math.cos(rad*latitudActualBase)
    *math.cos(rad*latitudInicioBase)*(math.sin(rad*dlong1/2))**2
diferencia1=2*R*math.asin(math.sqrt(a1))
#Diferencia entre paradaFin-ubicacionActual [Km]
dlat2=latitudActualBase-latitudFinBase
dlong2=longitudActualBase-longitudFinBase
a2=(math.sin(rad*dlat2/2))**2 +math.cos(rad*latitudFinBase)
    *math.cos(rad*latitudActualBase)*(math.sin(rad*dlong2/2))**2
diferencia2=2*R*math.asin(math.sqrt(a2))
```

Código 3.12. Implementación de la ecuación de Haversine

Luego de encontrar la distancia de la ubicación actual con respecto a las coordenadas de parada inicio y parada fin, se verifica que estas distancias estén dentro de un rango de 0 a 200 metros, si es el caso, el pulsador INICIO RUTA o FIN RUTA podrán activarse, posteriormente se guardará los datos a través de los métodos *guardarInicioRuta()* o *guardarFinRuta()* y activará su indicador correspondiente.

- Cuando pulsador EMERGENCIA "ha sido presionado": en el Código 3.13 se aprecia el proceso que se ejecuta para este pulsador, primero se crea un nuevo nodo dentro de la base de datos por cada pulsación realizada cuyo valor para la ramificación estado será igual a "True", este valor permite recibir y activar las notificaciones al aplicativo móvil, posteriormente este valor se debe cambiar a "False" con esto se asegura tener una sola notificación por pulsación.

```
if (estadoE==False):
    try:
        fechaEmergencia=datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        #Nueva emergencia con estado TRUE
        estado="True"
        nuevaEmergencia = db.reference('/EMERGENCIA/'+str(mac)).push({'estado':estado,'fecha':fechaEmergencia})
        gpio.output(23,True)
        #Tiempo de 5 segundos para cambiar el estado de la emergencia
        esperar(5)
        #Busqueda para desactivar emergencia
        emergencias = db.reference('/EMERGENCIA/'+str(mac)).get()
                for key in emergencias:
                    estadoEmergencia=referenciaCE.child('EMERGENCIA/'+str(mac)+'/'+str(key)+'/estado').get()
                    if(estadoEmergencia=="True"):
                    #Actualiza el estado de la emergencia a FALSE
                        estado="False"
                        actualizarEmergencia = db.reference('/EMERGENCIA/'+str(mac)+'/'+str(key)).update({'estado':estado})
                        gpio.output(23,False)
                        esperar(2)
```

Código 3.13. Código para el pulsador de emergencia

3.2.6. IMPLEMENTACIÓN DEL MÓDULO CÁMARA

En esta sección se presenta la instalación y configuración de todos los requerimientos involucrados para la conexión y funcionamiento del módulo cámara con la máquina virtual en AWS.

3.2.6.1 Raspberry

En la RPi viene por defecto deshabilitada la cámara, por lo que a continuación, se presenta la configuración necesaria para habilitar la cámara cuando ésta se encuentre conectada al puerto CSI en la RPi.

3.2.6.1.1. Habilitar cámara

Como primer paso se necesita ingresar al menú principal de configuraciones con el Comando 3.10.

```
pi@raspberrypi:~ $ sudo raspi-config
```

Comando 3.10. Ingreso al menú de configuración de la RPi

Luego de ejecutar el comando se tiene el menú de configuración (Ver Figura 3.11), en el cual se debe ingresar en *Interfacing Options*.

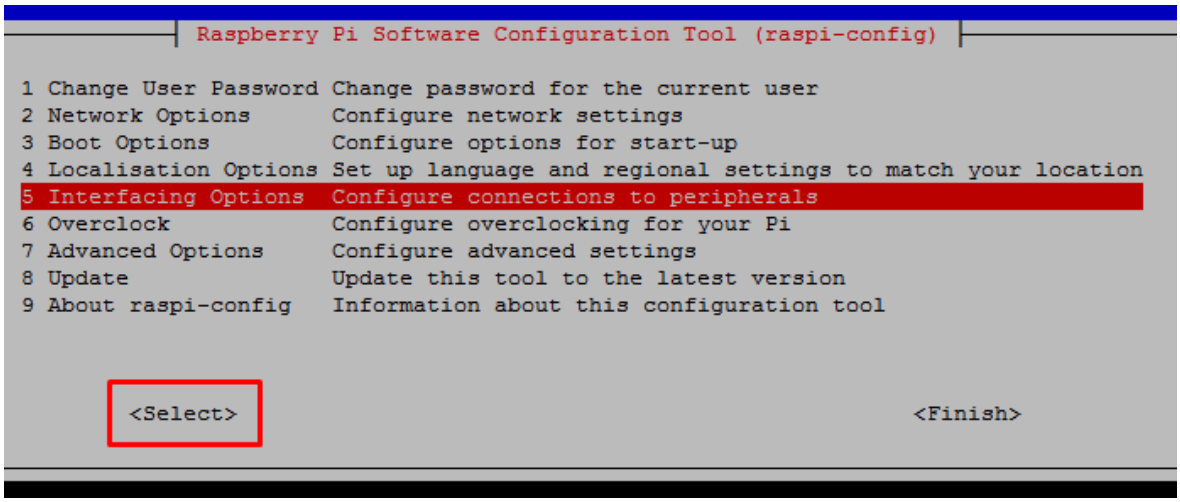


Figura 3.11. Menú raspi-config

Posteriormente dentro del menú Interfacing Options (ver Figura 3.12), se debe seleccionar “Camera” y habilitarla.

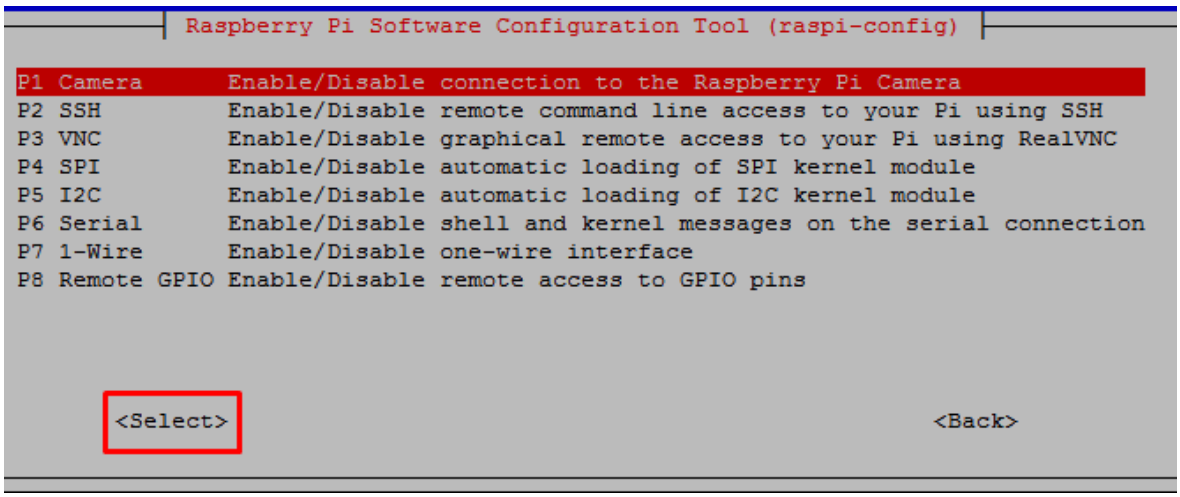


Figura 3.12. Menú interfacing options

Finalmente se necesita reiniciar a la RPi para que los cambios tengan efecto, por lo que es necesario ingresar el Comando 3.11.

```
pi@raspberrypi:~ $ sudo reboot
```

Comando 3.11. Reinicio para la RPi

Configuraciones adicionales

Es necesario realizar una actualización al firmware y el kernel de la RPi , para lo cual desde el terminal se digita el Comando 3.12.

```
pi@raspberrypi:~ $ sudo rpi-update
```

Comando 3.12. Actualización del firmware y KERNEL de la RPi

Posteriormente se necesita obtener el archivo que contiene la key del controlador para luego agregarla a la lista de keys confiables. Esto se consigue ingresando Comando 3.13.

```
pi@raspberrypi:~ $ curl http://www.linux-projects.org/listing/uv4l_repo/lpkey.asc | sudo apt-key add -
```

Comando 3.13. Obtención de la key UV4L

Luego en el archivo *source.list* se debe agregar el repositorio del cual se va a descargar los archivos de instalación del controlador, por lo que es necesario editar este archivo con el Comando 3.14.

```
pi@raspberrypi:~ $ sudo nano /etc/apt/sources.list
```

Comando 3.14. Editar el archivo source.list

Una vez abierto el archivo se agrega la siguiente línea (ver Figura 3.13):

```
deb http://www.linux-projects.org/listing/uv4l_repo/raspbian/stretch stretch main
```

Figura 3.13. Repositorio UV4L

Después se necesita realizar una actualización de los repositorios con el Comando 3.15.

```
pi@raspberrypi:~ $ sudo apt-get update
```

Comando 3.15. Actualización RPi

Posteriormente es necesario descargar e instalar el controlador con el Comando 3.16.

```
pi@raspberrypi:~ $ sudo apt-get install uv4l uv4l-raspicam
```

Comando 3.16. Descarga e instalación de uv4l

Cuando este procedimiento haya concluido, si se desea que el controlador cargue desde el arranque, es necesario instalar el paquete por el medio del Comando 3.17.

```
pi@raspberrypi:~ $ sudo apt-get install uv4l-raspicam-extras
```

Comando 3.17. Activación arranque

Finalmente, para que los cambios tengan efecto es necesario reiniciar el servicio (ver Comando 3.18) del controlador.

```
pi@raspberrypi:~ $ sudo service uv4l_raspicam restart
```

Comando 3.18. Reinicio del controlador

3.2.6.1.2. Instalación del servidor ZoneMinder

Como se mencionó en el marco teórico el software ZoneMinder requiere de Apache, MySQL y PHP a continuación, se presenta la instalación y configuración de cada uno de estos requerimientos.

Para instalar Apache y MySQL se digita en el terminal el Comando 3.19.

```
pi@raspberrypi:~ $ sudo apt-get install apache2 mysql-server
```

Comando 3.19. Instalación Apache y MySQL

Luego es necesario editar la zona horaria, para lo cual se ingresa al archivo *php.ini* del servidor apache con el Comando 3.20.

```
pi@raspberrypi:~ $ sudo nano /etc/php/7.0/apache2/php.ini
```

Comando 3.20. Edición archivo php.ini

Una vez dentro en el archivo se debe buscar y editar el valor de la línea *date.timezone* como se indica en la Figura 3.14.

```
date.timezone ="America/Guayaquil"
```

Figura 3.14. Línea zona horaria

A continuación, en el archivo *source.list* es necesario agregar el repositorio, donde se extraerán los paquetes necesarios para la instalación de ZoneMinder, para ello es necesario ingresar en el terminal el Comando 3.21 y agregar la línea (ver Figura 3.15).

```
pi@raspberrypi:~ $ sudo nano /etc/apt/sources.list
```

Comando 3.21. Editar archivo source.list

```
deb https://zmrepo.zoneminder.com/debian/release stretch/
```

Figura 3.15. Repositorio ZoneMinder

Posteriormente es necesario digitar el Comando 3.22 para descargar el archivo con la key y agregarla a la lista de keys confiables.

```
pi@raspberrypi:~ $ wget -O - https://zmrepo.zoneminder.com/debian/archive-keyrin
g.gpg | sudo apt-key add -
```

Comando 3.22. Obtención de la key ZoneMinder

Para que todos los cambios tengan efecto es necesario realizar una actualización en la RPi, esto se consigue mediante el Comando 3.23.

```
pi@raspberrypi:~ $ sudo apt-get update
```

Comando 3.23. Actualización RPi

Concluido el paso anterior, se instala ZoneMinder (ver Comando 3.24) y posteriormente se habilita el servicio (ver Comando 3.25).

```
pi@raspberrypi:~ $ sudo apt install zoneminder
```

Comando 3.24. Instalación ZoneMinder

```
pi@raspberrypi:~ $ sudo systemctl enable zoneminder.service
```

Comando 3.25. Habilitar servicio de ZoneMinder

Los siguientes comandos permiten configurar el directorio virtual de ZoneMinder (ver Comando 3.26).

```
pi@raspberrypi:~ $ sudo a2enconf zoneminder
pi@raspberrypi:~ $ sudo a2enmod rewrite
pi@raspberrypi:~ $ sudo a2enmod cgi
```

Comando 3.26. Configuración del directorio virtual ZoneMinder

Finalmente se carga la nueva configuración de Apache y se inicia el servicio de ZoneMinder (ver Comando 3.27).

```
pi@raspberrypi:~ $ sudo systemctl reload apache2
pi@raspberrypi:~ $ sudo systemctl start zoneminder
```

Comando 3.27. Configuración de Apache e inicio de ZoneMinder

Para comprobar que se tiene una correcta instalación es necesario ingresar en un navegador web la siguiente dirección: *direccionIP/zm.*

A continuación, en la Figura 3.16 se presenta la página principal de ZoneMinder una vez ya instalado.

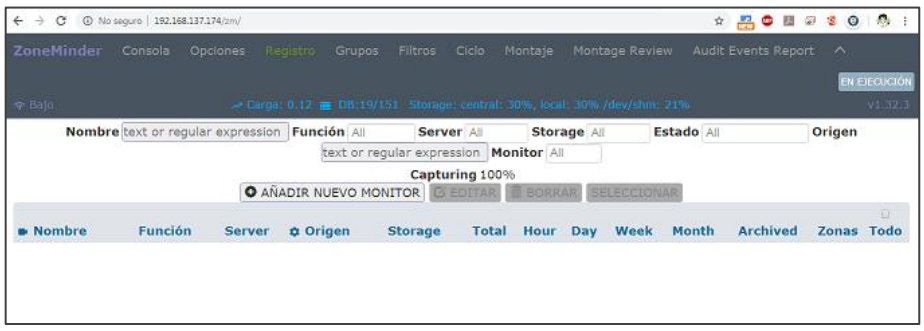


Figura 3.16. Página principal de ZoneMinder

3.2.6.1.3. Agregación de cámara en ZoneMinder

Para agregar la RaspiCam se debe dar clic sobre el botón “AÑADIR NUEVO MONITOR” que se encuentra en la página principal o también lo podemos encontrar dentro de la pestaña “Registro” de ZoneMinder. Seguidamente se mostrará una ventana, lugar donde se configura los parámetros necesarios dependiendo del usuario y tipo de dispositivo de video, a continuación, en la Figura 3.17 se presenta la configuración de la pestaña *General*.

Los parámetros de configuración varían dependiendo del tipo de cámara, para este caso el campo *Nombre* se refiere a la denominación o identificación que tendrá la cámara de video para este caso se la denomino *Video*, el campo *Server* define en cual servidor está la cámara y Storage Area define el servidor donde se almacenarán las grabaciones de los eventos, para fines de pruebas estos dos valores serán “Local”, es decir, en la RPi, el parámetro *Tipo de origen* es *ffmpeg* que permite capturar y codificar imágenes de video en tiempo real desde la RaspiCam, el parámetro *Función* es igual a *Monitor* permite visualizar las imágenes de video que está capturando.

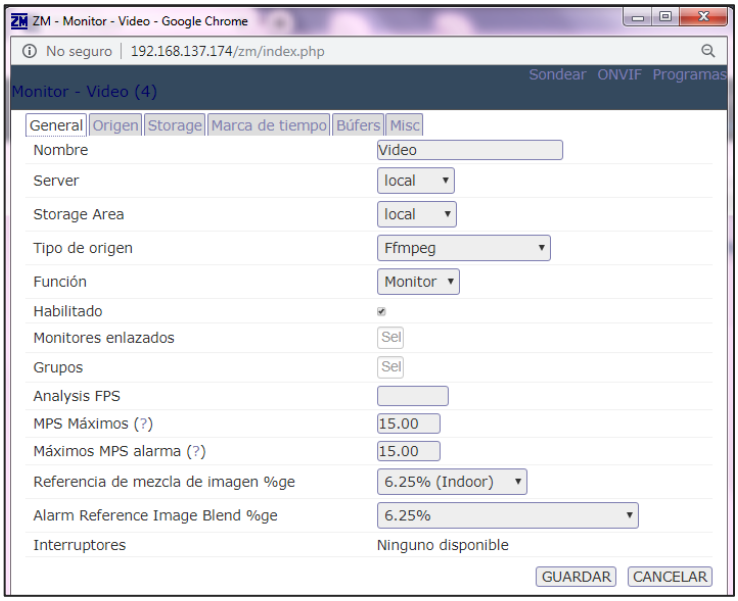


Figura 3.17. Pestaña "General"

La siguiente pestaña que se debe configurar es *Origen* (ver Figura 3.18) en el parámetro Ruta de origen se ubica el path de ubicación de la cámara, en este caso es */dev/video1*, para el parámetro Método remoto se eligió *UDP* debido a las características de este protocolo, en cuanto a los parámetros de *Ancho de captura* y *Altura de captura* se refieren al tamaño que tendrá el video que se generará en el software ZoneMinder.

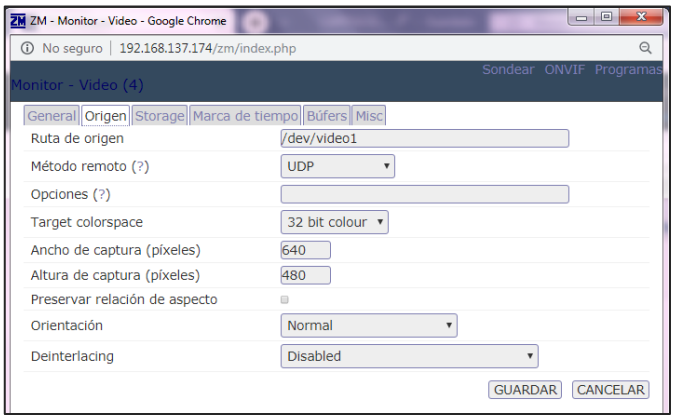


Figura 3.18. Pestaña "Origen"

La pestaña *Storage* (ver Figura 3.19) permite establecer como el monitor debe grabar el video, para este caso el campo *Save JPEGs* esta deshabilitado, debido a que el video en formato JPEG es de gran tamaño y como las grabaciones se van a almacenar en la RPi el espacio debe ser optimizado. Para el campo *Video Writer* se selecciona *H264 Camera Passthrough* ya que la RaspiCam se encuentra enviando grabaciones H264.

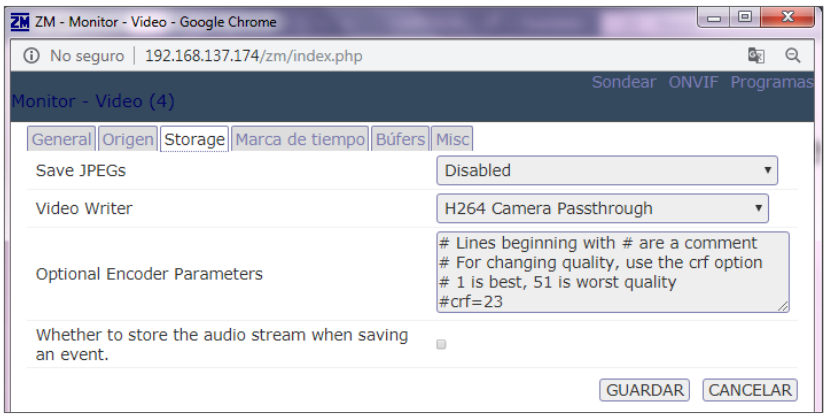


Figura 3.19. Pestaña "Storage"

Finalmente se guarda la configuración, posteriormente se mostrará la Figura 3.20, donde se puede visualizar el monitor creado. Para comprobar que la configuración fue correcta se accede dando clic en el nombre del monitor *Video*, seguidamente se cargara la ventana (ver Figura 3.21), donde se mostrará el video que está capturando en ese momento la RaspiCam.

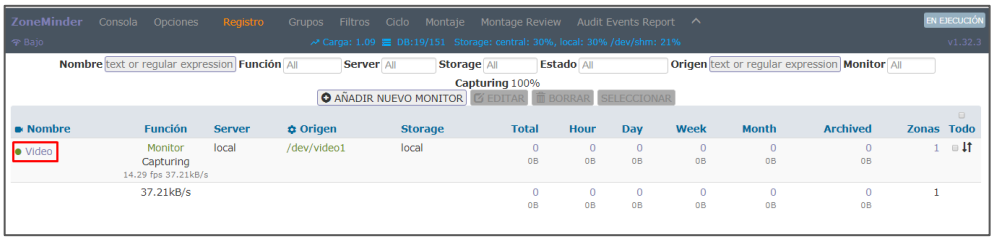


Figura 3.20. Monitor nuevo agregado

Figura 3.21. Visualización del monitor Video (RaspiCam)

3.2.6.1.4. Instalación de Dataplicity

Para poder tener acceso remoto a la cámara instalada en la RPi, se utilizó esta plataforma, su instalación y configuración es muy sencilla y se describe a continuación.

Para hacer uso de esta plataforma es necesario dirigirse a la página oficial https://www.dataplicity.com/ (Ver Figura 3.22) y registrar una dirección de correo de electrónico.

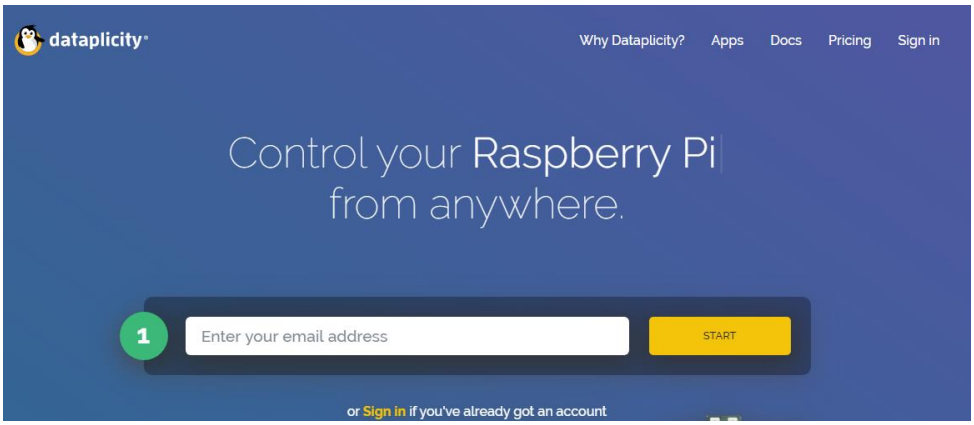


Figura 3.22. Página oficial de Dataplicity

Una vez introducida la dirección de correo, se generará automáticamente un comando (ver Figura 3.23) que permitirá descargar e instalar el cliente Dataplicity en la RPi.

Figura 3.23. Comando para instalar Dataplicity en RPi

Luego de instalar el cliente Dataplicity es necesario dirigirse nuevamente a la página web, donde se puede comprobar que la RPi está conectada y se puede controlarla remotamente. (ver Figura 3.24).

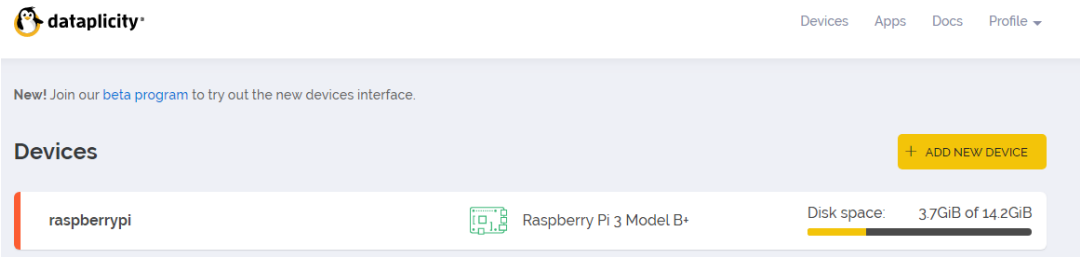


Figura 3.24. Listado de dispositivos registrados en Dataplicity

Cabe mencionar que a cada dispositivo asociado en la plataforma Dataplicity se le otorga un link permanente (ver Figura 3.25), mediante el cual la RPi puede publicar los distintos servicios que puede tener instalados. Este link será usado a futuro en la configuración multiservidor de ZoneMinder.

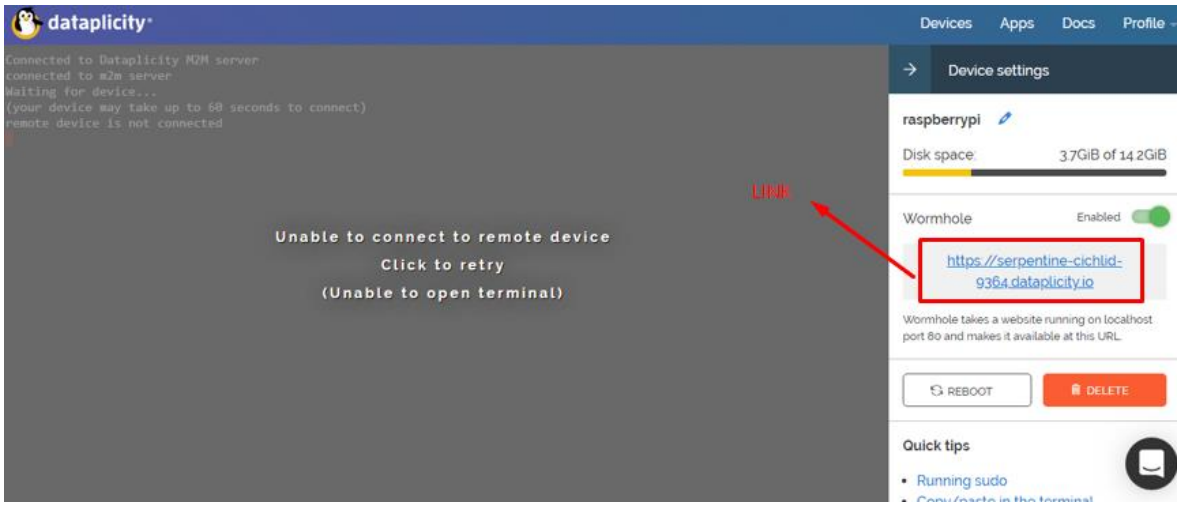


Figura 3.25. Link de RPi en Dataplicity

3.2.6.2 Máquina virtual en AWS

Como se mencionó en la fase de diseño el prototipo tendrá un servidor ZoneMinder centralizado para poder monitorear las cámaras en el caso de haya más de una unidad de

transporte estudiantil, el cual está en la nube de Amazon, por lo que a continuación, se presenta la instalación y configuración de una máquina virtual en la plataforma AWS.

3.2.6.2.1. Instalación y configuración de la máquina virtual

Para crear una máquina en la plataforma AWS es necesario registrarse en la página oficial https://aws.amazon.com/es/ (ver Figura 3.26) mediante un correo, así mismo es necesario registrar un método de pago puesto que el servicio que ofrece AWS no es gratuito.

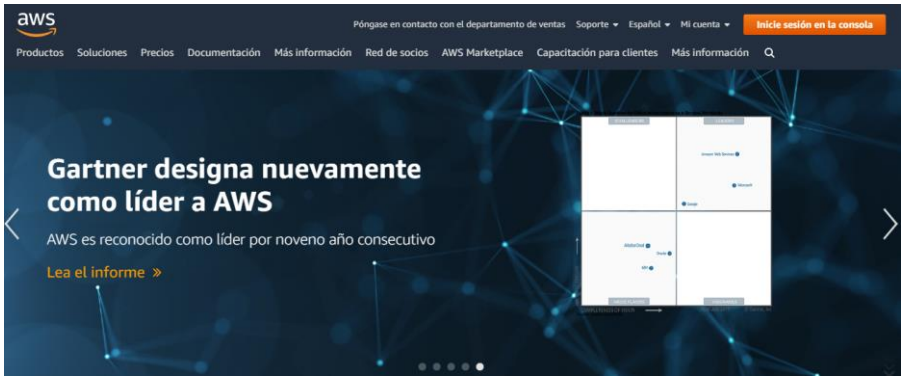


Figura 3.26. Página oficial AWS

Luego de crear la cuenta, se abre la consola de administración de AWS, en la pestaña de *Servicios* se selecciona *EC2* (ver Figura 3.27).

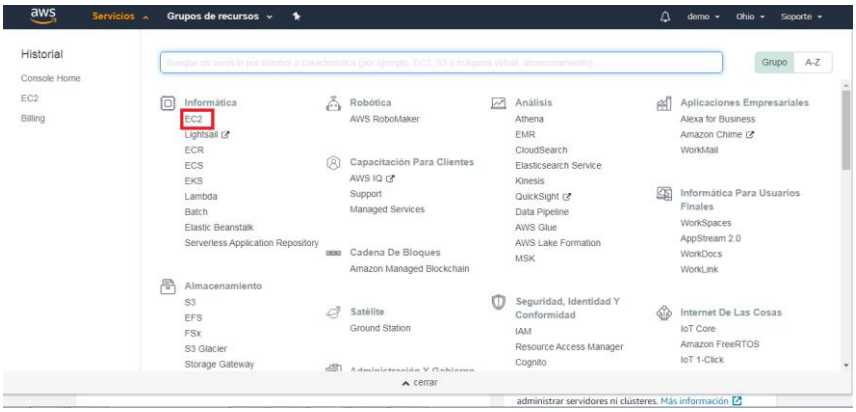


Figura 3.27. Panel de control de AWS

Posteriormente se debe seleccionar *Launch instance* en la sección *Create Instance* (ver Figura 3.28) para crear una máquina virtual.

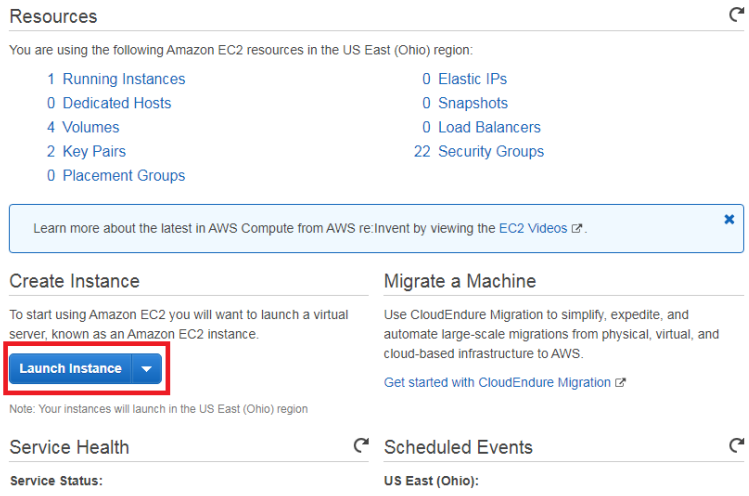


Figura 3.28. Creación de la instancia

Se mostrará la siguiente pantalla (ver Figura 3.29), donde es necesario elegir el sistema operativo que se instalará en la máquina virtual. Para este prototipo se selecciona el sistema operativo *UBUNTU SERVER 18.04 LTS*, puesto que es compatible con ZoneMinder.

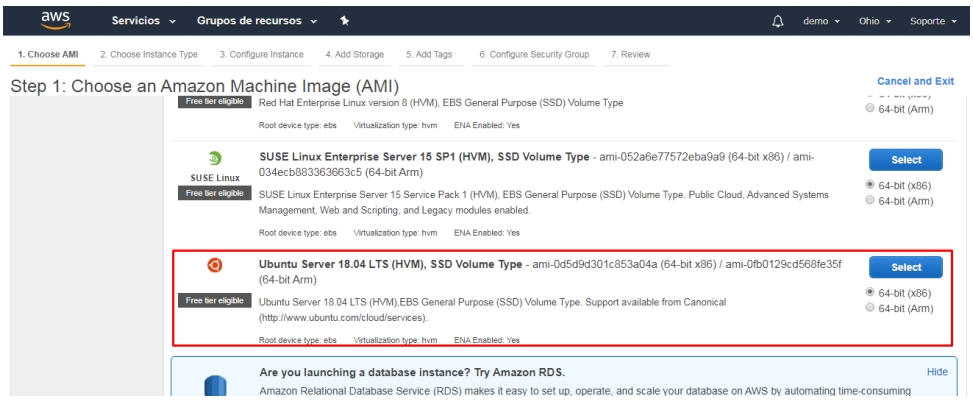


Figura 3.29. Selección del sistema operativo

Posteriormente es necesario escoger el tipo de instancia. Existen varios tipos de instancias (ver Figura 3.30) y se debe elegir dependiendo de las aplicaciones que se van a usar en la máquina virtual. Por defecto se escoge el tipo más bajo (t2.micro), ya que es la instancia más económica, la cual ofrece características suficientes para dar soporte al servicio de monitoreo de este prototipo. Luego de seleccionar la instancia adecuada se selecciona *Review Launch*.

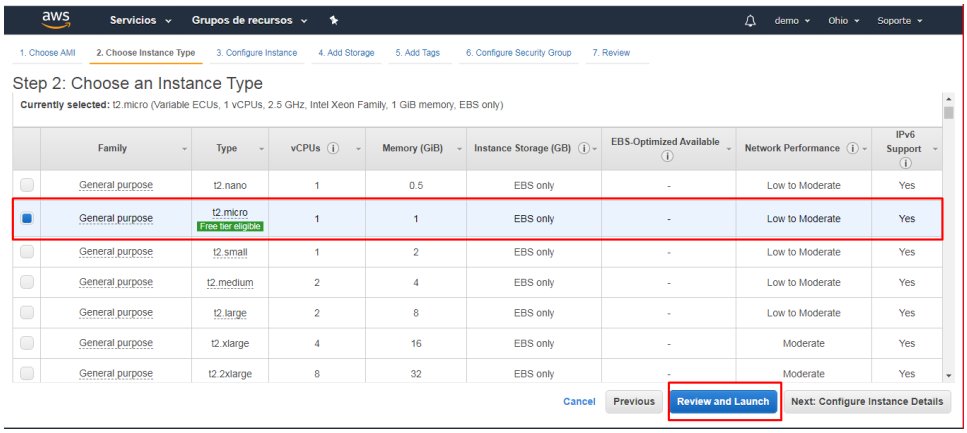


Figura 3.30. Tipos de instancias AWS

Finalmente, AWS muestra un resumen de los ajustes de configuración, memoria, tipo de instancia y seguridad (ver Figura 3.31). Una vez de acuerdo con el resumen es necesario dar clic en *Launch*.

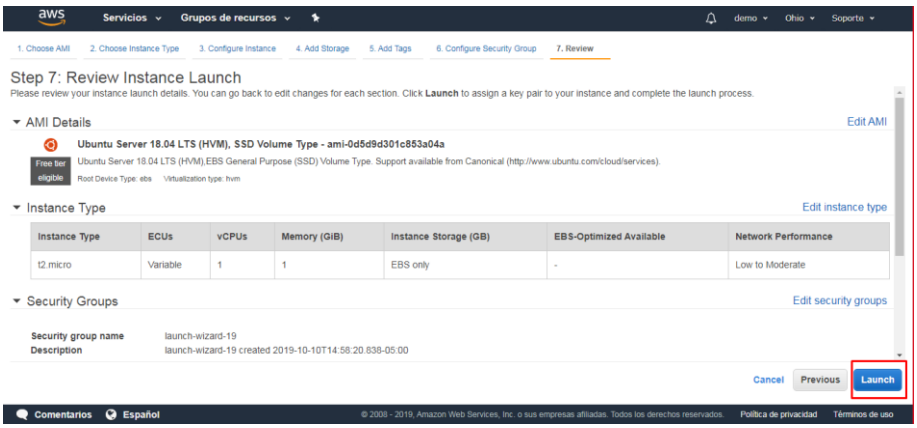


Figura 3.31. Resumen máquina virtual AWS

AWS solicita una última configuración (ver Figura 3.32), en la cual se debe crear una llave pública que permitirá acceder mediante el protocolo SSH a la máquina virtual que se creará. Una vez descargada la llave, se debe dar clic en *Launch* para inicializar la máquina virtual.

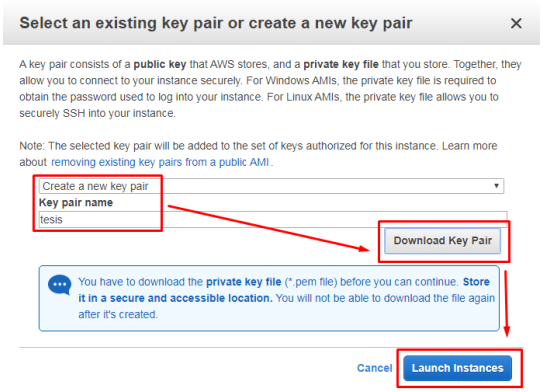


Figura 3.32. Configuración de key AWS

Como resultado en el panel de *EC2* se puede observar las instancias (máquinas virtuales) que estan iniciadas en la cuenta (ver Figura 3.33).

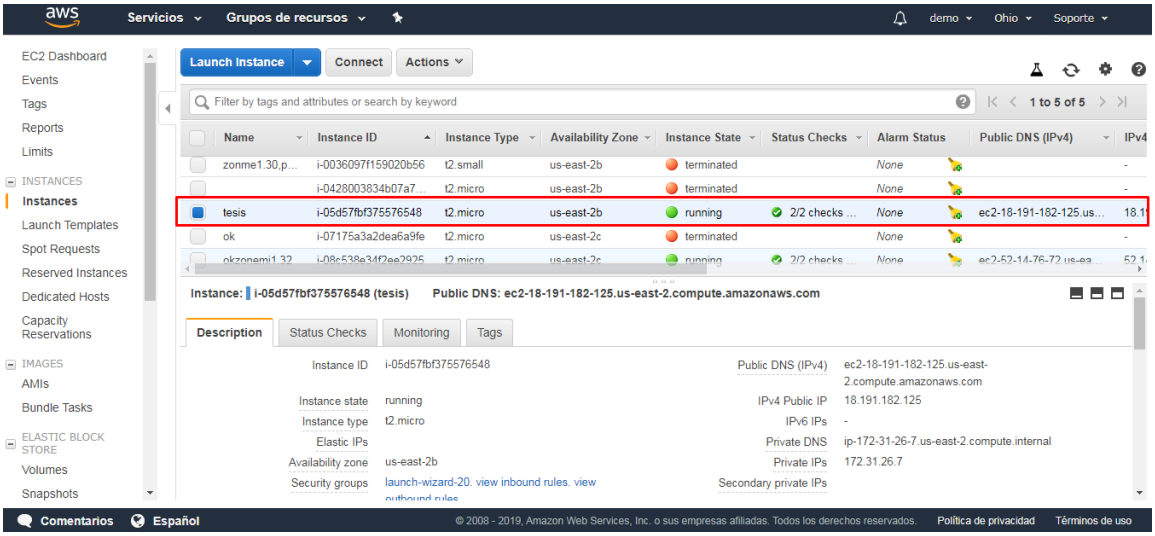


Figura 3.33. Máquinas virtuales de la cuenta

Cabe destacar que para realizar una conexión SSH con la máquina virtual es necesario conocer el DNS y el nombre de la máquina, por lo que en la Figura 3.34 se muestra que mediante clic derecho aparece la opcion de *Connect* teniendo asi como resultado el DNS y el nombre de la máquina (ver Figura 3.35), los mismos que hay que guardarlos para futuras configuraciones.

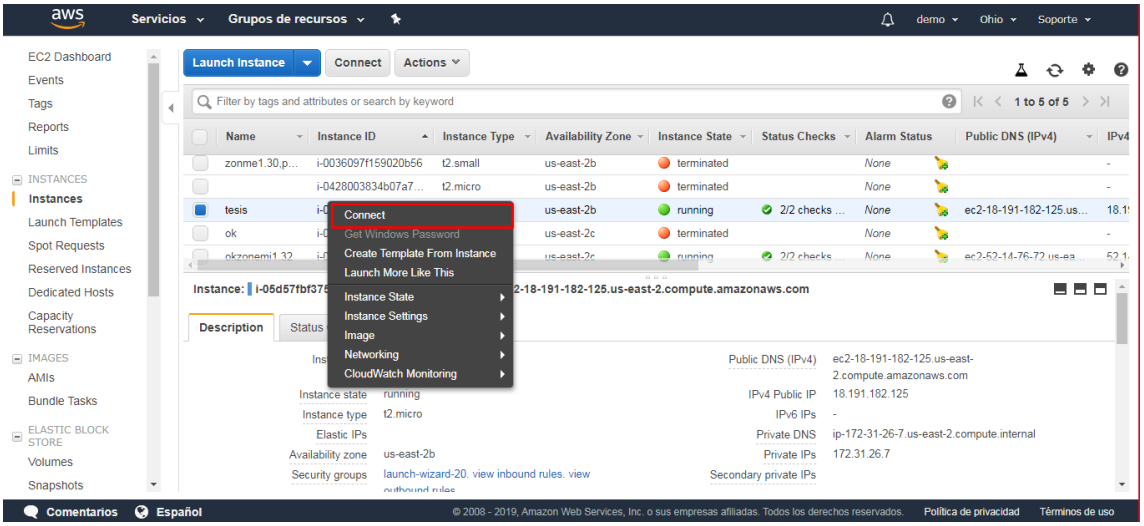


Figura 3.34. Opción Connect

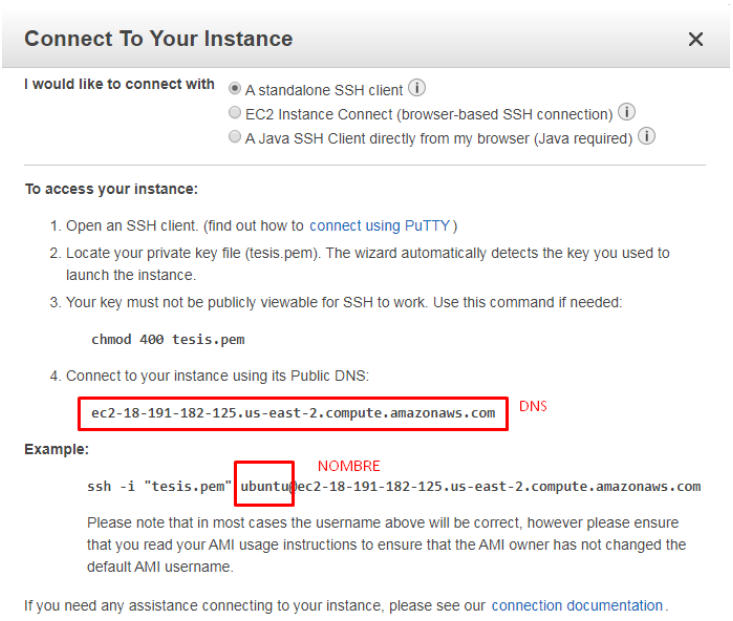


Figura 3.35. Valor de DNS y nombre de la máquina en AWS

3.2.6.2.2. Conexión SSH

Una vez ya finalizada la instalación y configuración del sistema operativo en la máquina virtual en AWS, es necesario generar una llave privada a partir de la llave pública descargada anteriormente por lo que se requiere cualquier programa que soporte conexiones SSH y genere llaves privadas, en este caso se utilizó PuTTYgen (ver Figura 3.36).

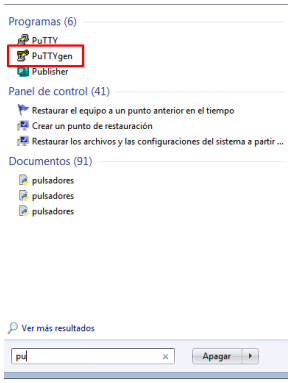


Figura 3.36. Selección PuTTYgen

Para generar la llave privada con PuTTYgen es necesario dar clic en *Load* (ver Figura 3.37) para seleccionar y cargar la llave pública anteriormente descargada.

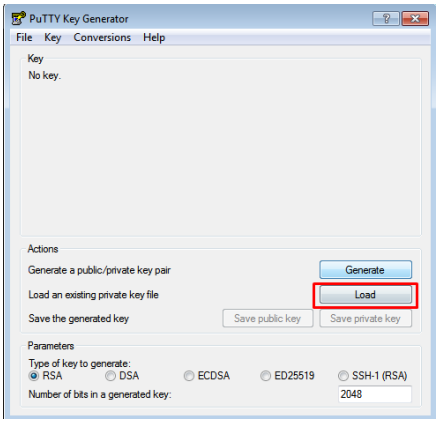


Figura 3.37. PuTTYgen

Luego de cargar el archivo con la llave pública, hay que guardar la llave privada que el programa generó, para lo cual se debe dar clic en *Save private key* (ver Figura 3.38), darle un nombre a esta llave y guardarla.

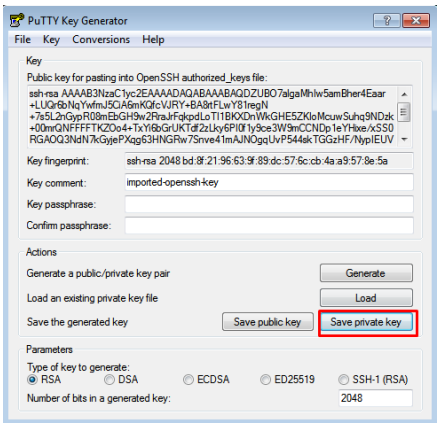


Figura 3.38. Guardar la llave privada

Una vez que se obtiene la llave privada se debe abrir el programa PuTTY y configurarlo para poder establecer una conexión SSH con la máquina virtual en AWS. En la Figura 3.39 se muestra la sección SSH, donde se debe cargar la llave privada.

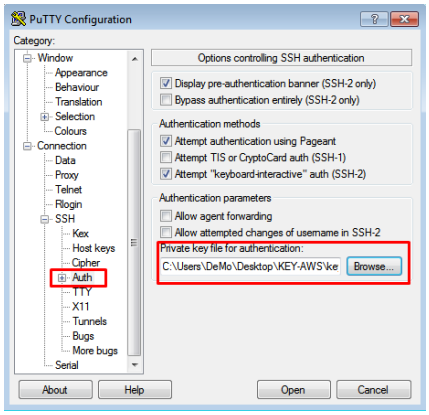


Figura 3.39. Cargar llave privada PuTTY

En la sección *Connection* > *Data*, se debe cargar el nombre de la máquina (ver Figura 3.40), el cual se obtuvo en un paso anterior.

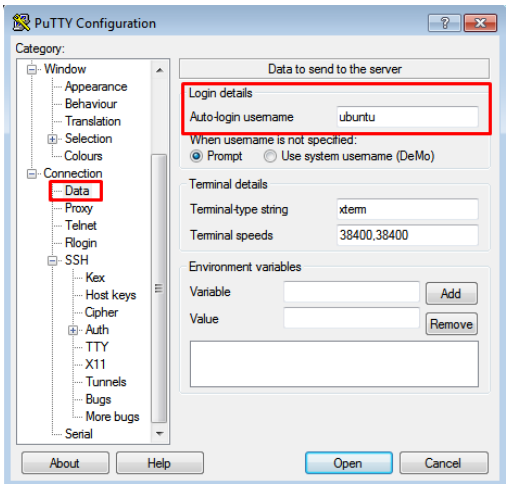


Figura 3.40. Datos para la conexión

Para concluir la configuracion de la conexión SSH, en la sección *Session* > *Host Name* (ver Figura 3.41) se debe colocar el DNS de la máquina que se obtuvo anteriormente, además, hay que darle un nombre a la sesión para que se guarden la configuración y así no volver a realizar todos los pasos anteriores. Acto seguido se debe dar clic en *Open*.

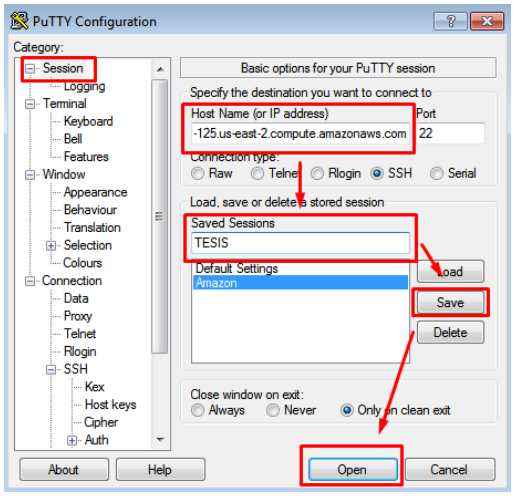


Figura 3.41. Sesión PuTTY

Como resultado de la configuración se tiene la Figura 3.42, una conexión SSH a la máquina virtual en la cual se podrá instalar y configurar los servicios necesarios a futuro.

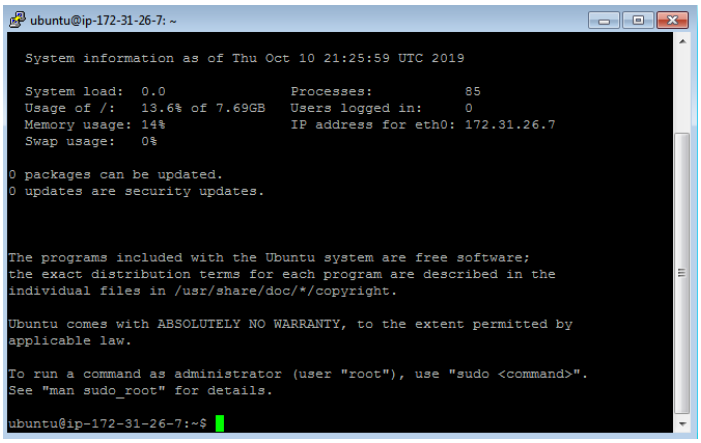


Figura 3.42. Conexión SSH hacia máquina AWS

3.2.6.2.3. Instalación del servidor ZoneMinder centralizado

En cuanto a la instalación de ZoneMinder lo primero que se debe realizar es una actualización de la máquina virtual, para lo cual se usa el Comando 3.28.

```
ubuntu@ip-172-31-26-7:~$ sudo apt update
```

Comando 3.28. Actualización máquina AWS

Al finalizar la actualización, se instala el servidor Apache y el servidor de base de datos mariadb con el Comando 3.29.

```
ubuntu@ip-172-31-26-7:~$ sudo apt install -y apache2 php mariadb-server php-mysql libapache2-mod-php7.2
```

Comando 3.29. Instalación de Apache y MariaDB

Una vez finalizado el proceso de instalación, es necesario editar el archivo *php.ini* con el Comando 3.30, para configurar la zona horaria. En la Figura 3.43 se indica el valor del campo *date.timezone*.

```
ubuntu@ip-172-31-26-7:~$ sudo nano /etc/php/7.2/apache2/php.ini
```

Comando 3.30. Editar archivo php.ini

```
date.timezone =America/Guayaquil
```

Figura 3.43. Línea zona horaria

A continuación, se agrega el repositorio, se actualiza y se configura el sistema para instalar ZoneMinder (ver Comando 3.31).

```
ubuntu@ip-172-31-26-7:~$ sudo apt-get install -y software-properties-common
ubuntu@ip-172-31-26-7:~$ sudo add-apt-repository ppa:iconnor/zoneminder-1.32
ubuntu@ip-172-31-26-7:~$ sudo apt update
```

Comando 3.31. Agregación y configuración del repositorio

Posteriormente se instala ZoneMinder y luego se habilita e inicia el servicio (ver Comando 3.32).

```
ubuntu@ip-172-31-26-7:~$ sudo apt install -y zoneminder
ubuntu@ip-172-31-26-7:~$ sudo systemctl enable zoneminder
ubuntu@ip-172-31-26-7:~$ sudo service zoneminder start
```

Comando 3.32. Instalación, habilitación e inicio de ZoneMinder

Por otra parte los comandos (ver Comando 3.33 y 3.34) son utilizados para agregar un usuario denominado *video* para el grupo *www-data* del servidor Apache. Y se establece como propietario al usuario *www-data* del directorio */usr/share/zoneminder* y a la vez permite que cualquier miembro del grupo *www-data* tenga acceso a los archivos de este directorio.

```
ubuntu@ip-172-31-26-7:~$ sudo adduser www-data video
```

Comando 3.33. Agregar usuario en servidor Apache

```
ubuntu@ip-172-31-26-7:~$ sudo chown -R www-data:www-data /usr/share/zoneminder/
```

Comando 3.34. Establecimiento de propietario y acceso al directorio

Luego se configura el directorio virtual de ZoneMinder y se reinicia el servidor Apache (ver Comando 3.35).

```
ubuntu@ip-172-31-26-7:~$ sudo a2enconf zoneminder
ubuntu@ip-172-31-26-7:~$ sudo a2enmod rewrite
ubuntu@ip-172-31-26-7:~$ sudo service apache2 reload
```

Comando 3.35. Configuración del directorio virtual

Finalmente, para ingresar a la pagina de inicio de ZoneMinder (ver Figura 3.44), se necesita digitar en la barra de direcciones del navegador web el DNS de la máquina virtual seguido el directorio zm (DNS/zm).

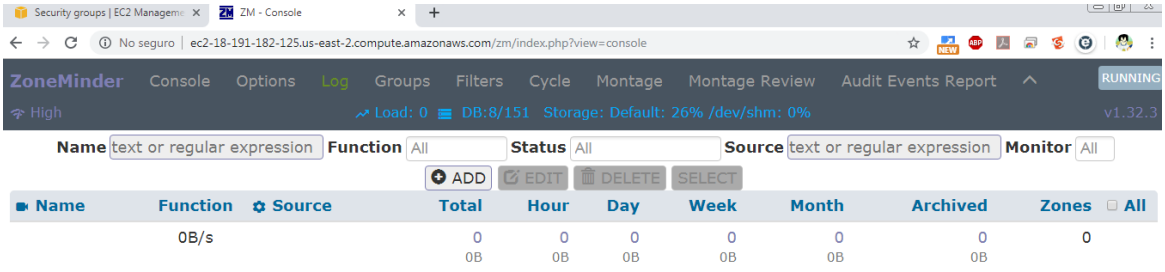


Figura 3.44. Página de inicio ZoneMinder centralizado

3.2.6.3 Configuración Multiservidor

Como se mencionó en la parte de diseño se tendrá un servidor centralizado al que se conectarán todas las unidades remotas. Para este momento ya se tiene instalado el servidor ZoneMinder en la RPi (local) y el servidor ZoneMinder en la máquina virtual de AWS (central). Basándose en la documentación oficial de ZoneMinder para la configuración del modo Multiservidor pasos detallados en [37]. A continuación, se presenta la configuración realizada:

Se edita el archivo *zm.conf* tanto para el servidor local y central con el Comando 3.36.

```
sudo nano /etc/zm/zm.conf
```

Comando 3.36. Editar archivo zm.conf

- **Local:**

En la Figura 3.45 se visualiza la modificación realizada en el archivo, donde:

ZM_DB_HOST: dirección IP del servidor centralizado.

ZM_SERVER_HOST: nombre que identifique al servidor remoto.

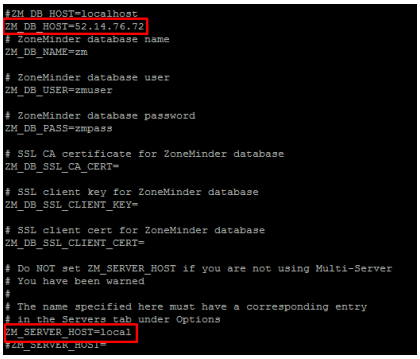


Figura 3.45. Configuración archivo zm.conf (local)

- **Central:**

La Figura 3.46 presenta la modificación realizada para el archivo *zm.conf* en el servidor central.

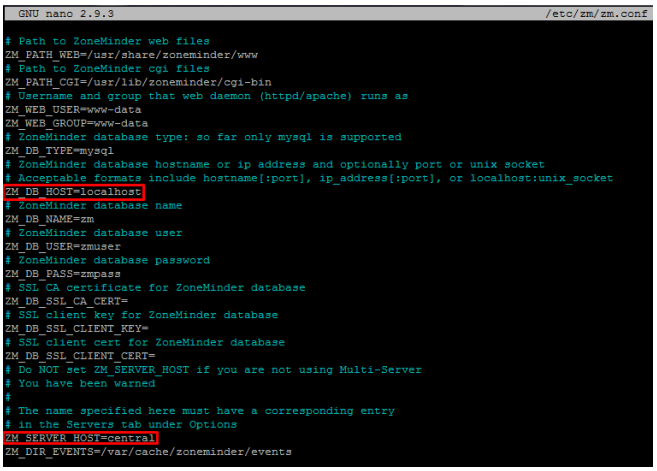


Figura 3.46. Configuración archivo zm.conf (central)

3.2.6.3.1. Configuración de servidores en servidor Centralizado

Para finalizar la configuración de ZoneMinder en multiservidor es necesario agregar el servidor *local* y *central* en la máquina AWS, para ello se accede a la opción *Servers* y finalmente al botón *ADD NEW SERVER* como se indica en la Figura 3.47. Seguidamente se mostrará una ventana donde se configura los parámetros del servidor central (ver Figura 3.48) y local (ver Figura 3.49).

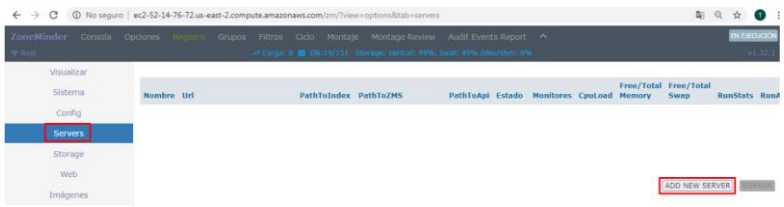


Figura 3.47. Opción Servers

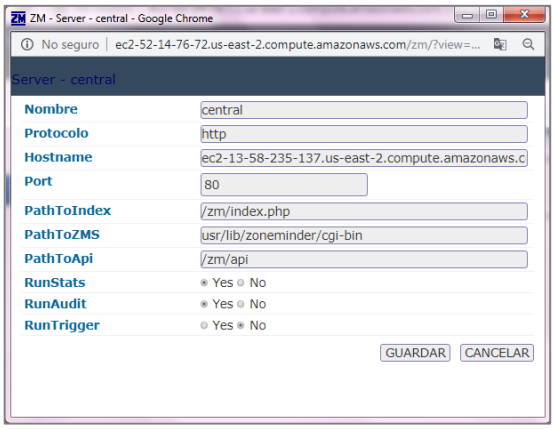


Figura 3.48. Añadir Servidor ZoneMinder AWS

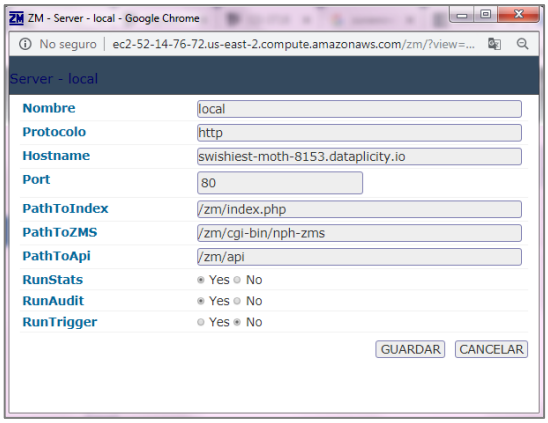


Figura 3.49. Añadir servidor ZoneMinder de RPi

Luego es necesario agregar el monitor remoto en el servidor central. La Figura 3.50 indica los valores añadidos a los parámetros de la pestaña *General*.

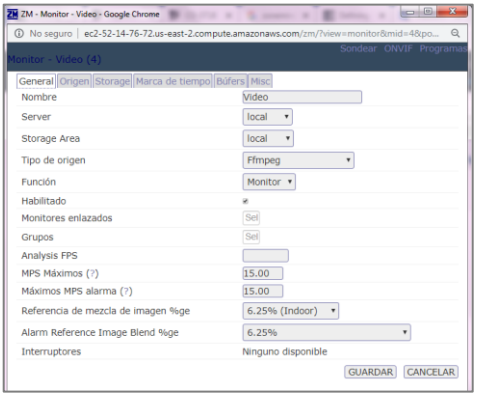


Figura 3.50. Añadir monitor remoto en ZoneMinder central

La Figura 3.51 presenta la configuración para la pestaña *Origen*.

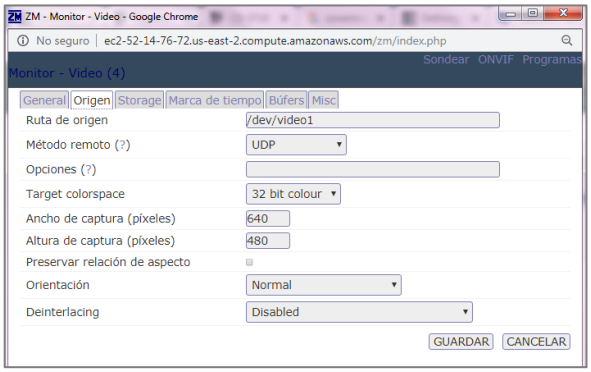


Figura 3.51. Configuración pestaña Origen

Y finalmente la Figura 3.52 indica la configuración para la pestaña Storage.

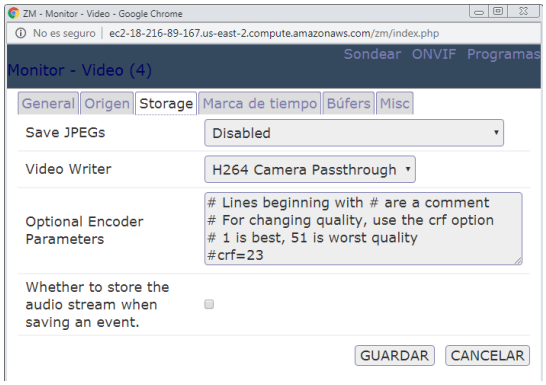


Figura 3.52. Configuración pestaña Storage

3.2.7. CREACIÓN DE SERVICIOS

Se emplearon 2 servicios en la RPi, uno de ellos inicia automáticamente la conexión a internet y el otro inicia el script, además, se puede monitorizar el script principal que es el encargado de obtener, procesar y guardar la información de cada uno de los módulos dentro de la unidad remota en la base de datos.

A continuación, se indica como ejemplo la creación del servicio denominado *mibus.service*, puesto que la creación y configuración de los parámetros es la misma para los dos servicios antes mencionados.

Para crear los servicios es necesario usar el Comando 3.37, el cual crea un archivo en el que se añaden los parámetros de configuración (ver Figura 3.53). Cabe mencionar que para el prototipo cada servicio está ejecutando un script diferente.

```
pi@raspberrypi:~ $ sudo nano /lib/systemd/system/mibus.service
```

Comando 3.37. Creación de un servicio

Figura 3.53. Configuración de parámetros del servicio mibus

Los valores y funciones de los parámetros usados se los detalla a continuación en la Tabla 3.5.

Tabla 3.5. Parámetros, función y valor de un servicio

PARÁMETRO	FUNCIÓN	VALOR
Description	Este parámetro describe el servicio.	*Este servicio monitorea el script del prototipo*
After	Este parámetro indica que el servicio se iniciará después de los servicios que se configuren como valor.	*multi-user.target network.target network-online.target*
StartLimitIntervalSec	Este parámetro reinicia el servicio las veces que sean necesarias.	*0*
Type	Este parámetro configura el inicio del servicio.	*simple*
Restart	Este parámetro reinicia el servicio en el caso de que el mismo se caiga.	*always*
RestartSec	Este parámetro configura el tiempo que debe transcurrir para que se intente reiniciar el servicio.	*2*
User	Este parámetro configura al nombre del usuario de la RPi.	*pi*
ExecStart	Este parámetro indica la ruta completa del archivo a ejecutar.	*/usr/bin/python /home/pi/Documents/miBus/main.py*
StandardOutput	Estos parámetros permiten escribir la salida y errores producidos por el script dentro del log de systemd.	*syslog*
StandardError		
WantedBy	Este parámetro establece el target o los targets necesarios para iniciar el servicio.	*multi-user.target*

Posteriormente con el Comando 3.38, se cargan los servicios creados en systemd.

```
pi@raspberrypi:~ $ sudo systemctl daemon-reload
```

Comando 3.38. Cargar servicios creados en systemd

Para habilitar cada servicio se necesita usar el Comando 3.39.

```
pi@raspberrypi:~ $ sudo systemctl enable mibus.service
```

Comando 3.39. Habilitar un servicio

Finalmente se ejecuta o inicia el servicio (ver Comando 3.40).

```
pi@raspberrypi:~ $ sudo systemctl start mibus.service
```

Comando 3.40. Inicializar un servicio

Con la inicialización de los servicios se puede monitorizar (ver Comando 3.42), para conocer el estado del programa codificado en el script (ver Figura 3.54).

```
pi@raspberrypi:~ $ sudo systemctl status mibus.service
```

Comando 3.41. Monitorea el estado del servicio

```
● mibus.service - Este servicio monitorea el script del prototipo.
   Loaded: loaded (/lib/systemd/system/mibus.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2019-12-09 19:49:09 -05; 26min ago
 Main PID: 795 (python)
   CGroup: /system.slice/mibus.service
           └─795 /usr/bin/python /home/pi/Documents/miBus/main.py

Dec 09 20:15:03 raspberrypi python[795]: -0.227008333333
Dec 09 20:15:03 raspberrypi python[795]: -78.4906283333
Dec 09 20:15:03 raspberrypi python[795]: -0.226978333333
Dec 09 20:15:03 raspberrypi python[795]: -78.490565
Dec 09 20:15:03 raspberrypi python[795]: -0.226948333333
Dec 09 20:15:03 raspberrypi python[795]: -78.4904983333
Dec 09 20:15:03 raspberrypi python[795]: -0.226913333333
Dec 09 20:15:03 raspberrypi python[795]: -78.4904266667
Dec 09 20:15:03 raspberrypi python[795]: -0.226873333333
Dec 09 20:15:03 raspberrypi python[795]: -78.4903516667
```

Figura 3.54. Monitoreo del script

3.3. IMPLEMENTACIÓN DEL SISTEMA DE COMUNICACIÓN

En esta sección de describe la implementación desarrollada para cada uno de los elementos que forman parte del sistema de comunicación: los servicios otorgados por la

plataforma Firebase y los diferentes aplicativos propuestos para el prototipo y a su vez se presenta la codificación relevante de los aplicativos.

3.3.1. IMPLEMENTACIÓN SERVICIOS FIREBASE

3.3.1.1 FIREBASE REALTIME DATABASE

Para empezar con la implementación del prototipo, se debe configurar la base de datos en la plataforma de Firebase, se accede al siguiente link https://console.firebase.google.com/ ,se loguea con una cuenta de Gmail, una vez ahí se puede acceder a la consola de Firebase (ver Figura 3.55), que muestra la opción de crear un nuevo proyecto.

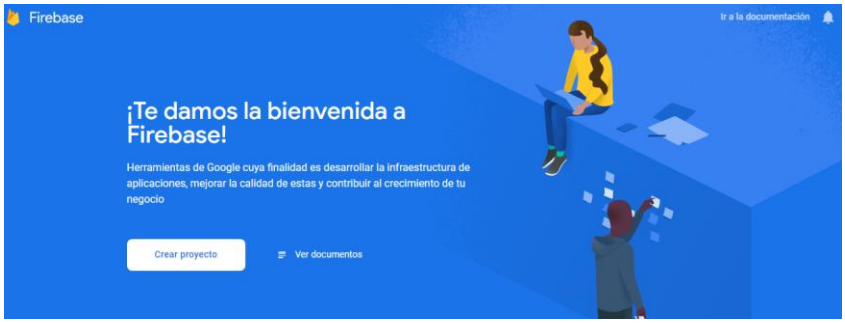


Figura 3.55. Consola de Firebase

Se crea el proyecto denominado *TesisBus*, se selecciona el país, se aceptan los términos de protección que ofrece Google, al finalizar la configuración se muestra la siguiente Figura 3.56 y finalmente se podrán incorporar los servicios que se deseen añadir al proyecto creado.

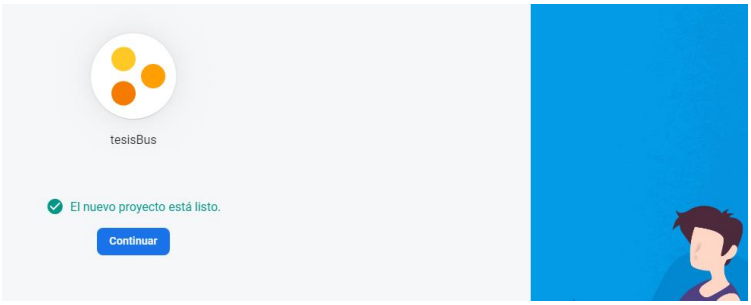


Figura 3.56. Mensaje de confirmación proyecto Firebase creado

Dentro del panel del proyecto creado, se añade el servicio *Database* y se escoge la opción de *Realtime Database* (ver Figura 3.57).

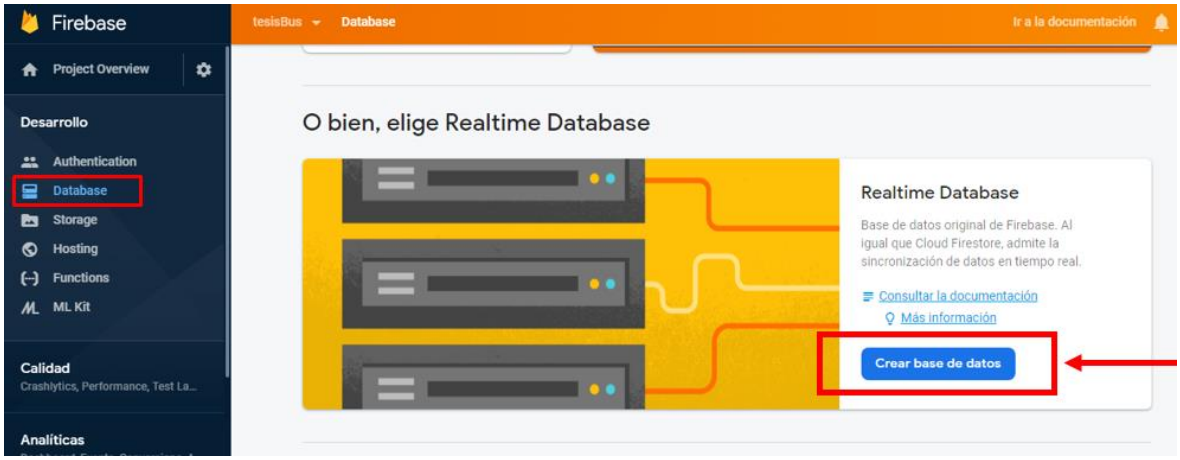


Figura 3.57. Opción base de datos Realtime Database

Al concluir esta configuración dentro del menú se accede a *Database* para crear los nodos principales definidos anteriormente en la fase de diseño (ver Figura 3.58).

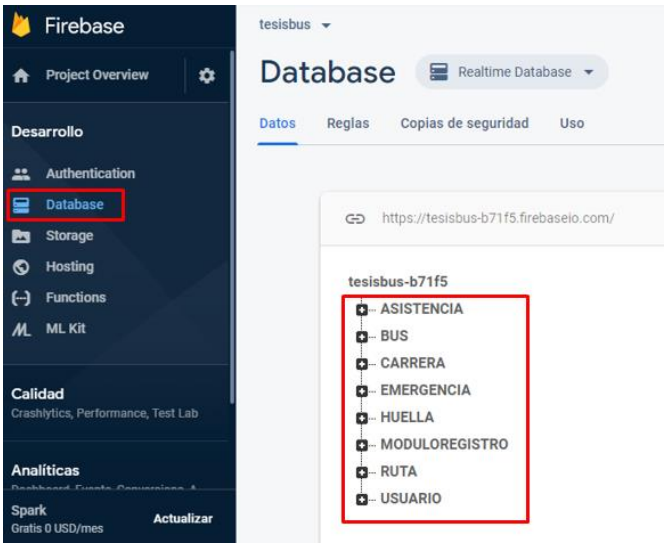


Figura 3.58. Nodos principales de base de datos

3.3.1.2 FIREBASE AUTHENTICATION

Para usar este servicio se debe configurarlo antes, para ello se debe dirigir a la consola de Firebase, luego seleccionar el proyecto creado anteriormente, después a la sección de *Authentication* (ver Figura 3.59), como se puede visualizar en la parte derecha existe una barra de navegación horizontal comprendida por: *Usuarios, Método de inicio de sesión, plantillas y uso*.

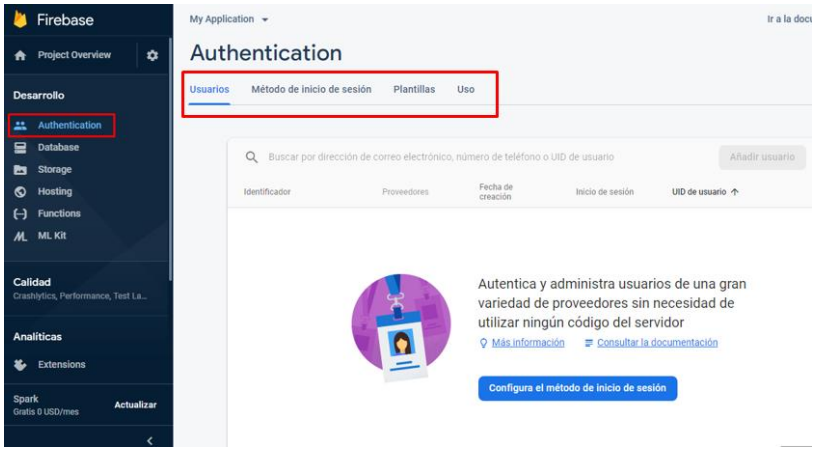


Figura 3.59. Selección del servicio de Autenticación

Para comenzar se debe elegir el tipo de *Método de inicio de sesión* y como se ha mencionado en la fase de diseño para el caso del prototipo se ha elegido el método de *correo y contraseña* (ver Figura 3.60).

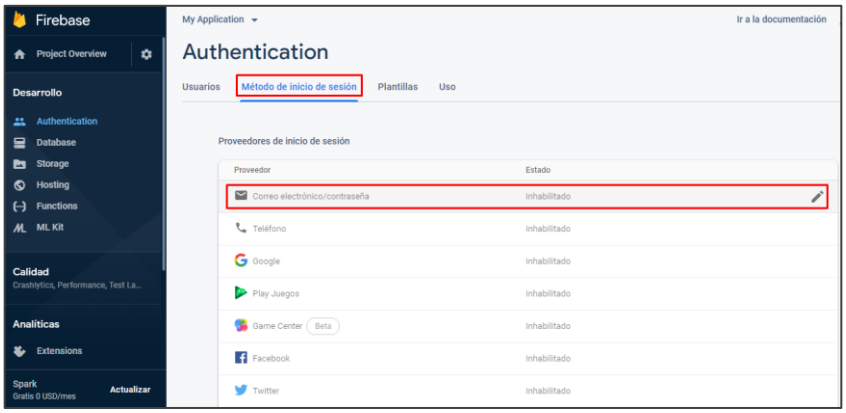


Figura 3.60. Selección del método de inicio de sesión

Seguidamente se presentará una ventana emergente la cual permitirá habilitar el uso del método de inicio de sesión elegido y se procede a guardar los cambios (ver Figura 3.61).

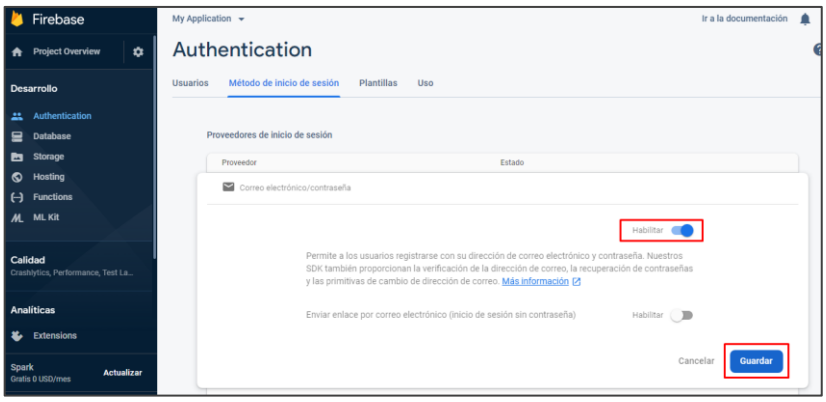


Figura 3.61. Habilitación del método de inicio de sesión

Como se mencionó en los requerimientos no funcionales se debe crear un usuario por defecto cuyo rol es *Administrador*, por lo tanto, en la barra de navegación se selecciona la pestaña *Usuarios*, se pulsa el botón *añadir usuario*, seguidamente se presentará una ventana en la que se ingresaran los datos que se soliciten (ver Figura 3.62), para el caso del prototipo el administrador tendrá como correo *tesisbus@gmail.com*.

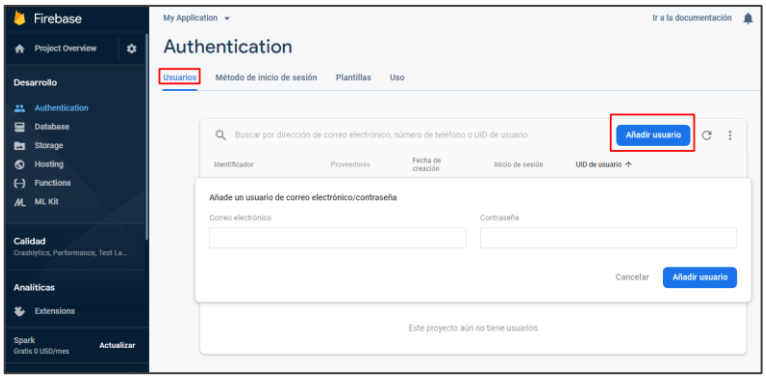


Figura 3.62. Creación de usuario predeterminado

Una vez finalizado el paso anterior se comprueba que el usuario añadido se presenta dentro de la pestaña *Usuarios*, en esta pestaña se enlistarán todos los usuarios que se registraran para usar los diferentes aplicativos (ver Figura 3.63). Como se puede visualizar cada usuario tiene varios atributos como:

- Identificador: es el correo con el que se registro al usuario.
- Proveedores: significa el tipo de autenticación elegido.
- Fecha de creación: fecha del registro del usuario.
- Inicio de sesion: indica el estado de inicio de sesion del usuario.
- UID de usuario: es un identificador unico otorgado por el servicio Firebase para diferenciar a los usuarios registrados.

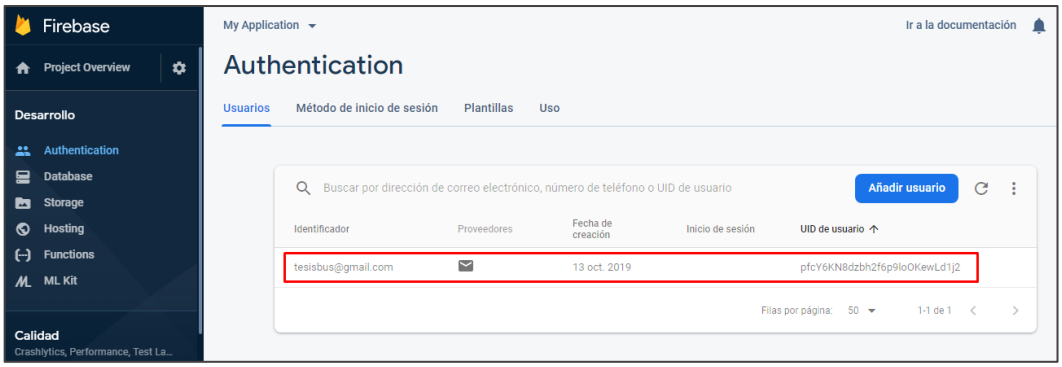
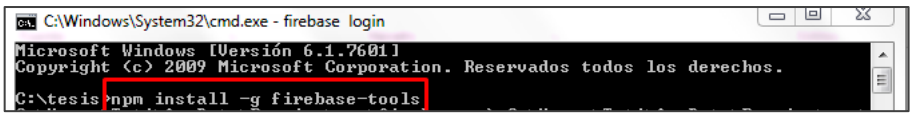


Figura 3.63. Usuario predeterminado

3.3.1.3 FIREBASE HOSTING

Para implementar el servicio de hosting es necesario descargar *Node.js* en la versión recomendada por Firebase. Una vez instalado, se debe crea una carpeta con el nombre del proyecto, para este caso es *tesis*, dentro de esta carpeta se ejecuta la consola de Windows, por medio de esta consola se instalan las herramientas de Firebase CLI con el Comando 3.42.



Comando 3.42. Instalación de herramientas Firebase

Una vez finalizada la instalación, con la ayuda del Comando 3.43, que permitirá enlazar el hosting con cuenta de Google, cuenta vinculada a la plataforma Firebase. A continuación, preguntará si se dan permisos, a lo cual se digitará *yes*.

Comando 3.43. Autenticación Firebase

Inmediatamente se abrirá un navegador con la información que se presenta en Figura 3.64, donde se debe autenticar. Para el caso del prototipo anteriormente se ha creado una cuenta de Gmail *tesisbus@gmail.com*.

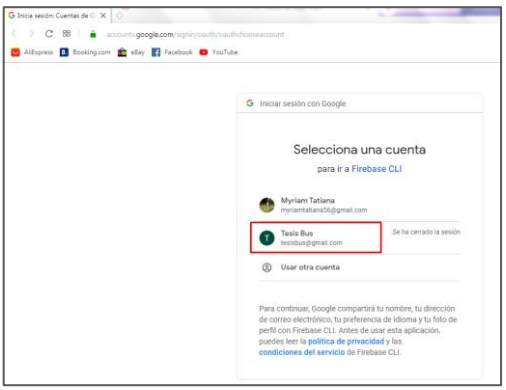
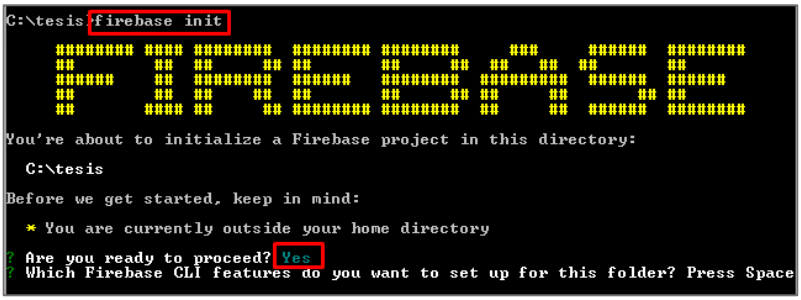


Figura 3.64. Autenticación Firebase

Enseguida se muestra el mensaje que se indica en la Figura 3.65, dentro de la consola de Windows.

Figura 3.65. Mensaje de autenticación Firebase

Luego se digita el Comando 3.44 que permitirá crear una carpeta *public* dentro de la carpeta que antes se creó, la misma que contendrá todos los archivos del aplicativo web.



Comando 3.44. Inicialización de Firebase

Después se muestra un menú, se selecciona el servicio que se desea incorporar, para este caso se seleccionara la opción que se indica en la Figura 3.66.

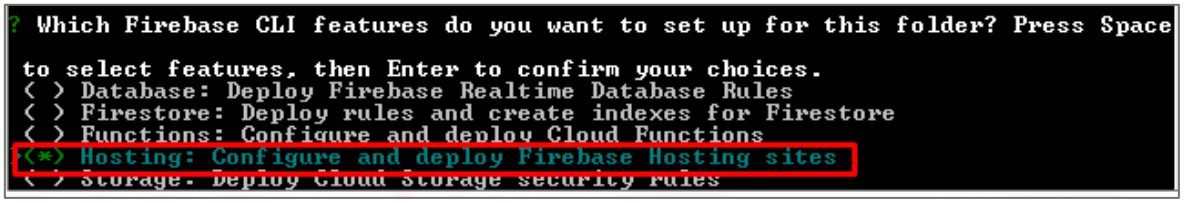


Figura 3.66. Selección de servicio de Hosting

Finalmente se solicita el proyecto en el cual se va a trabajar, para este caso se elige el proyecto *tesisBus* que fue creado anteriormente en la sección de implementación de Firebase Realtime Database. Para comprobar que Firebase Hosting está inicializado, en la carpeta *tesis* se podrá observar que se crearon varios archivos y además, la carpeta *public* (ver Figura 3.67).

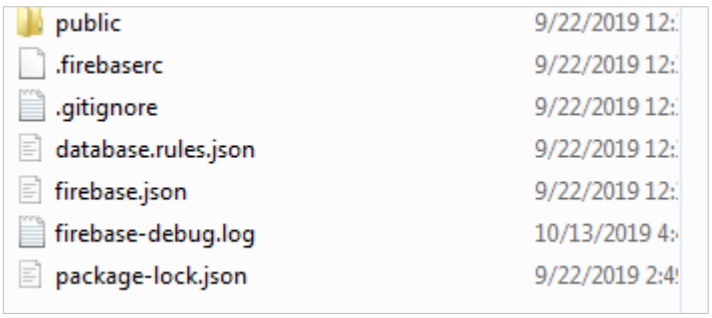


Figura 3.67. Archivos Firebase Hosting

Se debe destacar que una vez que se creen los archivos HTML o .js dentro de la carpeta *public*, para publicarlos en el hosting de Firebase se debe ingresar el comando *firebase deploy* desde la ubicación de la carpeta tesis.

3.3.2. IMPLEMENTACIÓN DEL APLICATIVO WEB

En esta sección se presenta el proceso de implementación del aplicativo web y módulo registro poniendo énfasis en las principales funciones del aplicativo.

La codificación de los archivos JavaScript y el diseño en HTML se encuentran en el Anexo E.

3.3.2.1 Servicio de Firebase

Al proyecto Firebase se le puede agregar uno o varios tipos de aplicaciones, para este caso se agregará una aplicación web, por ello se debe dirigir a la página principal del proyecto *Proyect Overview*, una vez ahí se escoge la aplicación cuyo icono a seleccionar se indica en la Figura 3.68.

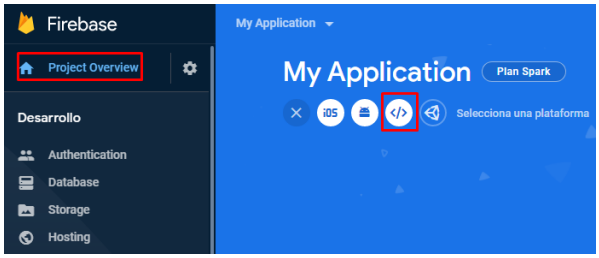


Figura 3.68. Selección de la aplicación web

Y se mostrará el siguiente fragmento código en *JavaScript* (ver Código 3.14), el mismo que permitirá a la aplicación web conectarse con el proyecto creado en Firebase y utilizar los servicios que se desee añadir.

```
<!-- The core Firebase JS SDK is always required and must be listed
<script src="https://www.gstatic.com/firebasejs/7.2.0/firebase-app.j

<!-- TODO: Add SDKs for Firebase products that you want to use
     https://firebase.google.com/docs/web/setup#available-libraries

<script>
  // Your web app's Firebase configuration
  var firebaseConfig = {
    apiKey: "AIzaSyAxIVi6K5W6P6he21ksx1ronVHcJxBfABA",
    authDomain: "tesisbus-b71f5.firebaseapp.com",
    databaseURL: "https://tesisbus-b71f5.firebaseio.com",
    projectId: "tesisbus-b71f5",
    storageBucket: "tesisbus-b71f5.appspot.com",
    messagingSenderId: "116828864684",
    appId: "1:116828864684:web:def6a1c8fcaaf3dd"
  };
  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
</script>
```

Código 3.14. Conexión al proyecto de Firebase

3.3.2.2 Módulo Registro

El módulo registro será utilizado únicamente por el administrador y trabajará conjuntamente con el aplicativo web para enrolar estudiantes y choferes.

3.3.2.2.1. Conexión

En la Figura 3.69 se puede apreciar el módulo registro que está conformado por los siguientes elementos electrónicos: una RPI, un sensor biométrico, un conversor serial-USB y una bocina.

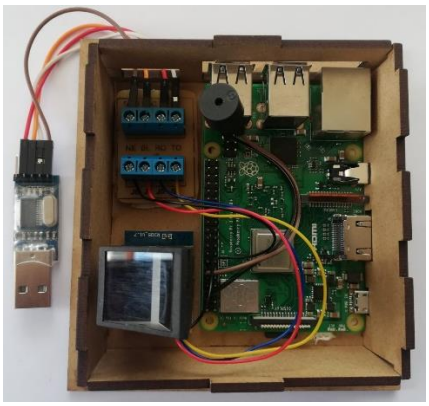


Figura 3.69. Módulo registro

La Tabla 3.6 muestra la conexión de los elementos que forman parte del módulo.

Tabla 3.6. Conexión elementos módulo registro

MÓDULO REGISTRO		CONVERSOR	RASPBERRY
Sensor biométrico	Vcc	5V	USB0
	Tx	Rx	
	Rx	Tx	
	GND	GND	
Bocina		-----	Pin 12

3.3.2.2.2. Codificación

El módulo registro tiene un hilo, el cual constantemente está verificando si existe un cambio en el nodo MODULOREGISTRO, este módulo cuenta con varios métodos pero el más relevante es el método *ordenes()* (ver Código 3.15) encargado de recibir órdenes desde el aplicativo web tales como:

- Activar el módulo registro solo cuando sea necesario.
- Enrolar estudiantes y choferes.

```
def ordenes():
    global opcionEnrolar,cedula,huella,idHuella
    while(selector=="on"):
        #Obtiene la orden y la cedula
        datos=db.reference('/MODULOREGISTRO/opcionEnrolar').get()
        separar = (datos).split(",")
        opcionEnrolar = separar[0]
        cedula =separar[1]
        if(opcionEnrolar=="0"):
            pass
        else:
            if(selector=="off"):
                pass
            try:
                #Lee por primera vez la huella
                modulo_huella.convertImage(0x01)
                #Enciente la bocina
                on_off_bocina(1)
                    while(modulo_huella.readImage()==False):
                    ]    pass
                #Lee por segunda vez la huella
                        modulo_huella.convertImage(0x02)
                        on_off_bocina(1)
                #Evalua si coincide o no la huella 01 y 02
                if ( modulo_huella.compareCharacteristics() == 0 ):
                    reset(0)
                else:
                    #Crea el template de 01 y 02
                    idHuella= modulo_huella.storeTemplate()
                    #Descarga la huella
                    huella = modulo_huella.downloadCharacteristics(0x01)
                    #Guarda la huella
                    guardarHuella()
                    opcionEnrolar="0"
                    reset(1)
                    #Borra la huella del sensor
                    modulo_huella.deleteTemplate(idHuella)
```

Código 3.15. Método ordenes()

3.3.2.3 Codificación de Formularios

Los formularios de administración básicamente utilizan los mismo métodos y propiedades otorgadas por Realtime Database Firebase para crear, eliminar o modificar cada uno de los objetos que se van a gestionar dentro del prototipo.

El Código 3.16 presenta la codificación realizada para crear un nuevo usuario en este caso un objeto estudiante, tanto en la base de datos como en Firebase Authentication.

```
function guardarEstudiante() {
  btn_huella.style.display = 'none';
  nombres = document.getElementById("txt_nombresE").value;
  apellidos = document.getElementById("txt_apellidosE").value;
  correo = document.getElementById("txt_correoE").value;
  clave = document.getElementById("txt_claveE").value;
  carrera = document.getElementById("txt_carreraE").value;
  linea = document.getElementById("txt_lineaE").value;
  estudianteNuevo = new Usuario(cedula, nombres, apellidos, carrera, correo, clave, linea, rol);
  firebase.auth().createUserWithEmailAndPassword(estudianteNuevo.correo, estudianteNuevo.clave)
    .then(function (userRecord) {
      baseDatosF.ref('USUARIO/' + estudianteNuevo.cedula).set({
        nombres: estudianteNuevo.nombres,
        apellidos: estudianteNuevo.apellidos,
        carrera: estudianteNuevo.carrera,
        correo: estudianteNuevo.correo,
        linea: estudianteNuevo.linea,
        rol: estudianteNuevo.rol
      });
```

Código 3.16. Creación usuario

El Código 3.17 muestra la codificación realizada para eliminar un usuario dentro de la base de datos.

```
function borrar() {
  var cedulaTabla = this.getAttribute("borrar");
  var referenciaborrar = baseDatosF.ref('USUARIO/').child(cedulaTabla);
  referenciaborrar.remove();
  var borrarHuella = baseDatosF.ref('HUELLA/').child(cedulaTabla);
  borrarHuella.remove();
}
```

Código 3.17. Eliminar usuario

El Código 3.18 muestra al método *update()* de Firebase que permite editar un objeto que se encuentra en la base de datos.

```
function editar() {
  linea = document.getElementById("txt_linea").value;
  baseDatosF.ref('BUS/' + linea).update({
    linea: document.getElementById("txt_linea").value,
    capacidad: parseInt(document.getElementById("txt_capacidad").value),
    descripcion: document.getElementById("txt_descripcion").value
  });
```

Código 3.18. Editar usuario

Para los formularios de registros principalmente se utilizan los métodos *traerEstudiantelinea(), traerRutasTerminadas()*, *traerPR() y pdf(),* por cada formulario los nombres de estos métodos varían pero realizan el mismo proceso.

El método *traerEstudianteLinea()* consiste en buscar dentro de la base de datos a todos los estudiantes que pertenecen a la misma unidad de transporte, unidad que se selecciona dentro de la interfaz de los formularios definidos. La codificación de este método se lo muestra en el Código 3.19.

```
function traerEstudianteLinea() {
    listaUsuarios = [];
    referenciaUsuario.orderByChild('rol').equalTo("Estudiante").once('value', function (snapshot) {
        var usuariosBDF = snapshot.val();
        for (var key in usuariosBDF) {
            usuario = new Usuario(key, usuariosBDF[key].nombres, usuariosBDF[key].apellidos,
                usuariosBDF[key].carrera, usuariosBDF[key].correo, 0, usuariosBDF[key].linea,
                usuariosBDF[key].rol);
            if (usuario.linea == linea) {
                listaUsuarios.push(usuario);
            }
        }
    });
}
```

Código 3.19. Método traerEstudianteLinea()

El Código 3.20 presenta la codificación realizada por el método *traerRutasTerminadas(),* este método busca basándose en el número de línea de la unidad de transporte solo a aquellas rutas que se hayan concluido.

```
function traerRutasTerminadas() {
    listaRutas = [];
    referenciaRuta = firebase.database().ref('RUTA/' + mac)
    referenciaRuta.once('value', function (snapshot) {
        var rutasBDF = snapshot.val();
        for (key in rutasBDF) {
            trt = new Ruta(key, rutasBDF[key].fechaInicioRuta, rutasBDF[key].fechaFinRuta);
            if (trt.fechaInicioRuta != trt.fechaFinRuta) {

                listaRutas.push(trt);
                traerPR(trt.id);
            }
```

Código 3.20. Método traerRutasTerminadas()

El Código 3.21 muestra la codificación del método *traerPR()*, este método ingresa a la base de datos para identificar a los estudiantes que han accedido al bus en una ruta especifica.

```
function traerPR(keyRuta) {
    var idRuta = keyRuta
    referenciaRuta = firebase.database().ref('RUTA/' + mac + "/" + idRuta + "/PASAJERO")
    referenciaRuta.once('value', function (snapshot) {
        var pasajeroRutaBDF = snapshot.val();
        for (cedula in pasajeroRutaBDF) {
            pasajeroRuta = new auxiliarPasajeros(idRuta, cedula,
                pasajeroRutaBDF[cedula].direccionSubida, pasajeroRutaBDF[cedula].direccionBajada);
            listaPasajeros.push(pasajeroRuta);
        }
    });
```

Código 3.21. Método traerPR()

Y finalmente, el método *pdfS()* que se encarga de generar un documento en formato .pdf del registro generado, su codificación se aprecia en el Código 3.22.

```
function pdfS() {
    var doc = new jsPDF('L', 'mm', 'A4');
    var res = doc.autoTableHtmlToJson(document.getElementById("tabla_semanal"));
    var resT = doc.autoTableHtmlToJson(document.getElementById("tabla_total"));
    var header = function (data) {
        /***setFontSize tamaño de letra, (N1,N2,'text') N1:espaciobordeizquierdo, N2:EspacioArriba*/
        doc.setFontSize(16);
        doc.text(100, 20, 'ESCUELA POLITÉCNICA NACIONAL')
        doc.setFontSize(12);
        doc.text(105, 28, 'CONTROL DE ASISTENCIA ESTUDIANTE')
        doc.setFontSize(12);
        doc.text(114, 34, 'REGISTRO SEMANAL "PoliBus"')
        doc.setFontSize(10);
        doc.text(15, 42, 'Línea: ' + linea + '      Descripción: ' + descripcion + '        Capacidad:

    };
    doc.autoTable(resT.columns, resT.data, { margin: { top: 50 }, beforePageContent: header });
    doc.autoTable(res.columns, res.data, { margin: { top: 60 }, beforePageContent: header });
    filename = "RegistroEstudianteSemanal"+fechaSeleccionada + '.pdf';
    doc.save(filename);
```

Código 3.22. Método pdf()

3.3.3. IMPLEMENTACIÓN DEL APLICATIVO ANDROID

En esta sección se presenta el proceso para la implementación del aplicativo móvil, poniendo énfasis en las principales funciones que esta aplicación tiene tanto con Firebase como con sus propias funcionalidades.

La codificación en Java y el diseño en XML para cada pantalla (activity) de este aplicativo se encuentra en el ANEXO F.

3.3.3.1 Servicios Firebase

El aplicativo usa los servicios de Firebase Realtime Database y Firebase Authentication, los mismos que pueden ser agregados de dos maneras al proyecto en Android Studio: *manual*, consiste en seguir la documentación que está disponible en la página oficial de Firebase y el asistente de *Firebase* que viene instalado en Android Studio. Para esta implementación se usó la forma más rápida de agregar las dependencias de cada servicio Firebase, por lo que a continuación, se detalla cómo se agregaron estos servicios. En la Figura 3.70 se muestra como vincular una cuenta de Gmail a Android Studio.

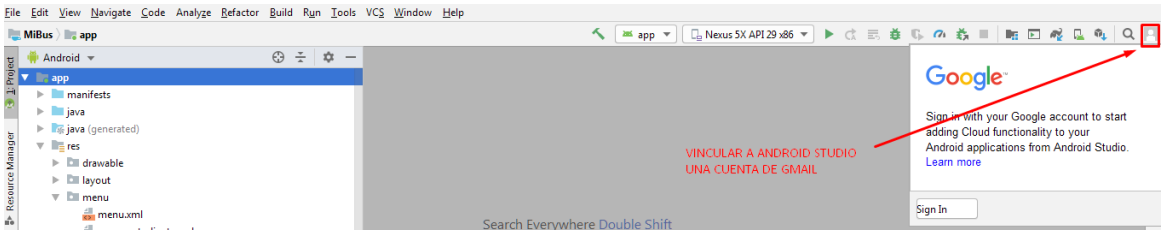


Figura 3.70. Vinculación de una cuenta Gmail a Android Studio

Una vez finalizado el proceso de vinculación, se usa el asistente de Firebase para añadir las dependencias de Firebase Realtime Database (ver Figura 3.71) y de las dependencias de Firebase Authentication (ver Figura 3.72).

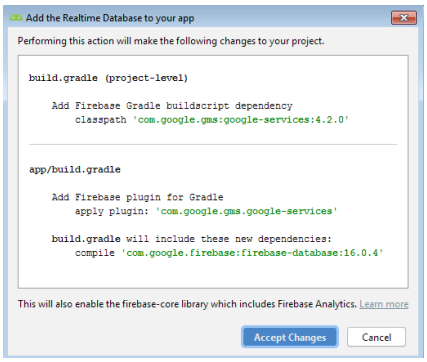


Figura 3.71. Dependencias Firebase Realtime Database

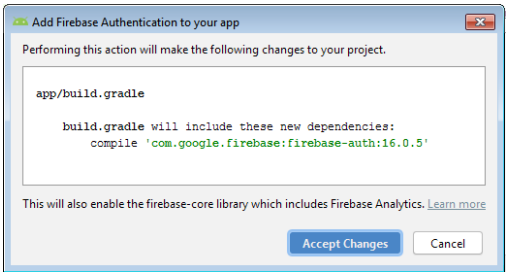


Figura 3.72. Dependencias Firebase Authentication

3.3.3.2 Pantallas

La *pantalla inicio* es la que se lanza inicialmente cuando el aplicativo se instalada por primera vez en el dispositivo móvil o bien cuando el usuario lo inicia en el día a día, por lo que en esta pantalla se necesita comprobar si el usuario este logueado para luego redirigirlo a la pantalla adecuada dependiendo de su rol (administrador o estudiante). Para conseguir este proceso se codificó el método *onAuthStateChanged()* (ver Código 3.23), en el cual se evalúa el inicio de sesión del usuario por medio de Firebase Authentication, de haber iniciado sesión se obtiene la dirección de correo electrónico y no haberlo hecho se lanza la *pantalla login*.

```
mAuthStateListener = new FirebaseAuth.AuthStateListener() {
    @Override
    public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {

        FirebaseUser user = firebaseAuth.getCurrentUser();
        if(user!=null){
            CargarUsuarioActual(user.getEmail());
        }else{
            Intent abrirLogin = new Intent ( packageContext: Inicio.this, Login.class);
            startActivity(abrirLogin);
        }
    }
};
```

Código 3.23. Método onAuthStateChanged()

El Código 3.24 presenta la codificación realizada para el caso *restablecer contraseña*, primero verifica que un correo haya sido ingresado en el *EditText* perteneciente al correo, luego se codifica el método *sendPasswordResetEmail()*, el cual es el responsable de enviar un email al correo especificado para restablecer la contraseña, el método *setLanguageCode()* que se encarga de enviar el email en un idioma específico.

```
label_olvide_clave.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        String correo = txt_correo.getText().toString().trim();
        String clave = txt_clave.getText().toString().trim();

        if(!correo.isEmpty()){
            mDialog.setMessage("Espere un momento...");
            mDialog.setCanceledOnTouchOutside(false);
            mDialog.show();

            mFirebaseAuth.setLanguageCode("es");
            mFirebaseAuth.sendPasswordResetEmail(correo).addOnCompleteListener(new OnCompleteListener<Void>() {
                @Override
                public void onComplete(@NonNull Task<Void> task) {

                    if(task.isSuccessful()){
                        Toast.makeText(Login.this, "Se ha enviado restablecer contraseña", Toast.LENGTH_SHORT).show();
                    }else{
                        Toast.makeText(Login.this, "No se logro enviar el correo", Toast.LENGTH_SHORT).show();
                    }
                    mDialog.dismiss();
                }
            });

        }else{
            Toast.makeText(Login.this, "Ingrese el correo electrónico", Toast.LENGTH_SHORT).show();
        }
    }
```

Código 3.24. Restablecer contraseña

Para la *pantalla login*, se codifica del método *signInWithEmailAndPassword()* que recibe como parámetros el correo y clave (ver Código 3.25), valores capturados de los *EditText* correspondientes, este método otorgado por el servicio Firebase Authentication permite verificar si las credenciales del usuario están registradas en el servicio, en el caso de encontrarse registrado se realizada una consulta a la base de datos a partir del correo, con el fin de obtener los datos del usuario logueado para almacenarlos dentro de un objeto del tipo usuario, luego se obtiene el atributo rol del objeto ya que dependiendo del rol, se mostrará su pantalla correspondiente y finalmente se enviará el objeto en el *Intent* mediante *putExtras().*

```
mFirebaseAuth.signInWithEmailAndPassword(correo, clave)
        .addOnCompleteListener(Login.this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    Toast.makeText(Login.this, "Bienvenido", Toast.LENGTH_SHORT).show();
                    Query consulta= bbdd_usuarios.orderByChild(getString(R.string.campo_correo)).equalTo(correo);
                    consulta.addValueEventListener(new ValueEventListener() {
                        @Override
                        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                            for (DataSnapshot snapshot:dataSnapshot.getChildren())
                            {
                                USUARIO user= snapshot.getValue(USUARIO.class);
                                if(user.getRol().equals("Estudiante")){
                                    Intent abrirBus = new Intent(Login.this, Informacion.class);
                                    Bundle bundle = new Bundle();
                                    bundle.putSerializable("usuario",user);
                                    abrirBus.putExtras(bundle);
                                    startActivity(abrirBus);
                                }else{

                                    Intent abrirControl = new Intent(Login.this, Control.class);
                                    Bundle bundle = new Bundle();
                                    bundle.putSerializable("usuario",user);
                                    abrirControl.putExtras(bundle);
                                    startActivity(abrirControl);
                        }}}
```

Código 3.25. Método signInWithEmailAndPassword()

Para la *pantalla información* los métodos principales son *cargarUsuarioLogeado()*, *consultarBus()* y *obtenerUbicacionBus().* El Código 3.26 permite recibir el objeto enviado desde la *pantalla login*, guarda en una variable la línea de bus a la cual pertenece el estudiante para posteriormente realizar las diferentes consultas en la base de datos y también inicia el servicio *IntentServicio,* el cual se encarga de escuchar constantemente el nodo *EMERGENCIA* para recibir notificaciones en el dispositivo móvil.

```
private void CargarUsuarioLogueado() {
    Bundle objetoEnviado = getIntent().getExtras();
    USUARIO usuario=null;
    if(objetoEnviado!=null) {
        usuario = (USUARIO) objetoEnviado.getSerializable("usuario");
        linea = usuario.getLinea();
        startService(intentServicio);
    }
```

Código 3.26. Método cargarUsuarioLogeado()

El Código 3.27 presenta una parte de la codificación del servicio de notificaciones, este servicio crea una notificación cuando se genera un nodo secundario en la base de datos

dentro del nodo principal *EMERGENCIA*, para el caso de administrador recibirá las notificaciones generadas por todos los buses que se encuentren registrados pero para los estudiantes solo se recibirá las notificaciones del bus al cual se haya registrado.

```
private void Notificacion(String linea,String descripcion,String fecha) {
    Intent notificacion = new Intent(thisContext, Emergencias.class);
    BUS bus = new BUS(linea,descripcion);
    Bundle bundle = new Bundle();
    bundle.putString("fecha",fecha);
    bundle.putInt("selector",1);
    bundle.putSerializable("bus",bus);
    notificacion.putExtras(bundle);
    notificacion.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
    PendingIntent pendingIntent = PendingIntent.getActivity(thisContext,0,notificacion,
            PendingIntent.FLAG_ONE_SHOT);
    NotificationManager notificationManager=
            (NotificationManager)getSystemService(Context.NOTIFICATION_SERVICE);
    Uri defaulSoundUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
    Notification.Builder notificationBuilder = new Notification.Builder(thisContext).
            setSmallIcon(R.drawable.ic_stat_name).
            setContentTitle("EMERGENCIA EN SU BUS: "+bus.getLinea()+". "+bus.getDescripcion()).
            setContentText("Tome las precauciones del caso").
            setAutoCancel(true).
            setSound(defaulSoundUri).
            setContentIntent(pendingIntent);
    if (Build.VERSION.SDK_INT >=Build.VERSION_CODES.O) {
        String channelId = getString(R.string.normal_channel_Id);
        String channelnombre = getString(R.string.normal_channel_nombre);
        NotificationChannel channel = new NotificationChannel(channelId,channelnombre,
                NotificationManager.IMPORTANCE_DEFAULT);
        channel.enableVibration(true);
        channel.setVibrationPattern(new long[]{100,200,200,50});
        if(notificationManager !=null) {
            notificationManager.createNotificationChannel(channel);
        }
        notificationBuilder.setChannelId(channelId);
    }
    if(notificationManager !=null){
        notificationManager.notify("",0,notificationBuilder.build());
    }
```

Código 3.27. Servicio de notificaciones

El fragmento de Código 3.28 pertenece al método *consultarBus()*, este método muestra en pantalla la ruta que se está realizando en ese instante en el caso de existir, el número de pasajeros que se encuentran actualmente en el bus y el número de asientos disponibles

```
bbdd_ruta.child(keyBus).addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
        for(DataSnapshot datosRuta:dataSnapshot.getChildren()){
            RUTA ruta=new RUTA();
            ruta = datosRuta.getValue(RUTA.class);
            keyRuta=datosRuta.getKey();
            String fechaInicio = ruta.getFechaInicioRuta().toString();
            String fechaFin=ruta.getFechaFinRuta().toString();
            if(!fechaInicio.equals(fechaFin) )
            {
                label_fecha.setText(getString(R.string.bus_no_ruta));
                label_asientos.setText(getString(R.string.bus_no_ruta));
            }else{
                label_fecha.setText(ruta.getFechaInicioRuta());
                bbdd_pasajero = FirebaseDatabase.getInstance().getReference(PATH_RUTA)
                        .child(keyBus).child(keyRuta).child(PATH_PASAJERO);
                bbdd_pasajero.addValueEventListener(new ValueEventListener() {
                    @Override
                    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                        numsub=0;
                        disponibles=0;
                        for(DataSnapshot datosPasajero:dataSnapshot.getChildren() ){
                            PASAJERO pasajero = new PASAJERO();
                            pasajero=datosPasajero.getValue(PASAJERO.class);
                            String estadoPasajero=pasajero.getEstado();
                            if(estadoPasajero.equals("True")){
                                numsub=numsub+1;
                            }}
                        disponibles=capacidad-numsub;
                        label_asientos.setText(" "+disponibles );
                    }
```

Código 3.28. Método consultarBus()

El Código 3.29 presenta al método *ObtenerUbicacionBus()*, método que se encarga de mostrar en el mapa de Google la ubicación del bus al que pertenece el estudiante basándose en las coordenadas de latitud y longitud, además, dibuja un marcador en esa ubicación cada vez que el bus cambie de coordenadas.

```
private void ObtenerUbicacionBus(GoogleMap googleMap) {
    mMap = googleMap;
    LatLng posiBus = new LatLng(-0.3446, -78.48);
    mMap.addMarker(new MarkerOptions().position(posiBus).title("Su bus"));
    mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(posiBus,16));
    Query consultaUbicacion = bbdd_bus.orderByChild(getString(R.string.campo_linea)).equalTo(linea);
    consultaUbicacion.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            for(Marker marker:realTimeMarker){
                marker.remove();
            }
            for (final DataSnapshot snapshot:dataSnapshot.getChildren())
            {
                BUS bus = snapshot.getValue(BUS.class);
                latitud = bus.getLatitud();
                longitud = bus.getLongitud();
                MarkerOptions markerOptions = new MarkerOptions();
                LatLng posiBus = new LatLng(latitud, longitud);
                markerOptions.position(posiBus).title("PoliBus: "+bus.getLinea()+". "+bus.getDescripcion());
                mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(posiBus,16));
                tmpRealTimeMarker.add(mMap.addMarker(markerOptions));
            }
            realTimeMarker.clear();
            realTimeMarker.addAll(tmpRealTimeMarker);
        }
```

Código 3.29. Método ObtenerUbicacioBus()

Tanto el administrador como estudiante tendrán una *pantalla emergencia*, donde podrán visualizar las notificaciones recibidas, pero solo el administrador podrá eliminar estas desde la *pantalla control*. En el Código 3.30 se presenta lo más relevante perteneciente a la pantalla mencionada.

```
lst_emergencias.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(final AdapterView<?> adapterView, View view, int i, long l) {
        if (auxSelector.equals(2)) {
            final String fecha_Seleccionada = adapterView.getItemAtPosition(i).toString();
            Query consultaBus = bbdd_bus.orderByChild(getString(R.string.campo_linea)).equalTo(linea);
            consultaBus.addListenerForSingleValueEvent(new ValueEventListener() {
                @Override
                public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                    for (DataSnapshot datosBus:dataSnapshot.getChildren()){
                        keyBus=datosBus.getKey();
                        bbdd_emergencia.child(keyBus).orderByChild(PATH_FECHA).addValueEventListener(new ValueEventListener() {
                            @Override
                            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                                listadoEmergencia.clear();
                                for(DataSnapshot datosEmergencia:dataSnapshot.getChildren()){
                                    EMERGENCIA emergencia= datosEmergencia.getValue(EMERGENCIA.class);
                                if (emergencia.getFecha().equals(fecha_Seleccionada))
                                {
                                    keyEmergencia = datosEmergencia.getKey();
                                    AlertDialog.Builder dialogo =new AlertDialog.Builder(Emergencias.this);
                                    dialogo.setMessage("¿Desea eliminar la emergencia: "+fecha_Seleccionada+" ?")
                                            .setCancelable(false)
                                            .setPositiveButton("SI", new DialogInterface.OnClickListener() {
                                                @Override
                                                public void onClick(DialogInterface dialogInterface, int i) {
                                                    bbdd_emergencia.child(keyBus).child(keyEmergencia).removeValue();
                                                    Toast.makeText(Emergencias.this, "Emergencia eliminada...", Toast.LENGTH_SHORT).show();
                                                    Intent abrirInicio = new Intent (Emergencias.this, Control.class);
                                                    startActivity(abrirInicio);
                                                }
                                            })
                                            .setNegativeButton("NO", new DialogInterface.OnClickListener() {
                                                @Override
                                                public void onClick(DialogInterface dialogInterface, int i) {
                                                    Toast.makeText(Emergencias.this, "Ningún cambio", Toast.LENGTH_SHORT).show();
                                                    Intent abrirInicio = new Intent (Emergencias.this, Control.class);
                                                    startActivity(abrirInicio);
                                                }
```

Código 3.30. Notificaciones de emergencias

4. RESULTADOS Y DISCUSIÓN

En esta sección se presentan las pruebas de cada uno de los aplicativos y el funcionamiento en conjunto del prototipo, cabe mencionar que las pruebas fueron evaluadas acorde a los criterios de aceptación que se presentaron en cada historia de usuario. Además, se muestran las correcciones a los errores identificados en las pruebas y los ajustes que se realizaron para que funcione de manera óptima.

También, en esta sección se presentan las encuestas de satisfacción realizadas a los usuarios involucrados en el desarrollo de este prototipo (ver ANEXO G) y finalmente se realizó un breve análisis de tráfico.

4.1. ACTUALIZACIÓN DEL TRABLERO KANBAN

En la Figura 4.1 se puede apreciar que al momento no existen tareas por realizar, puesto que toda la implementación ya fue realizada exitosamente y en esta sección se necesita realizar finalmente las pruebas de funcionamiento para todos los sistemas del prototipo.

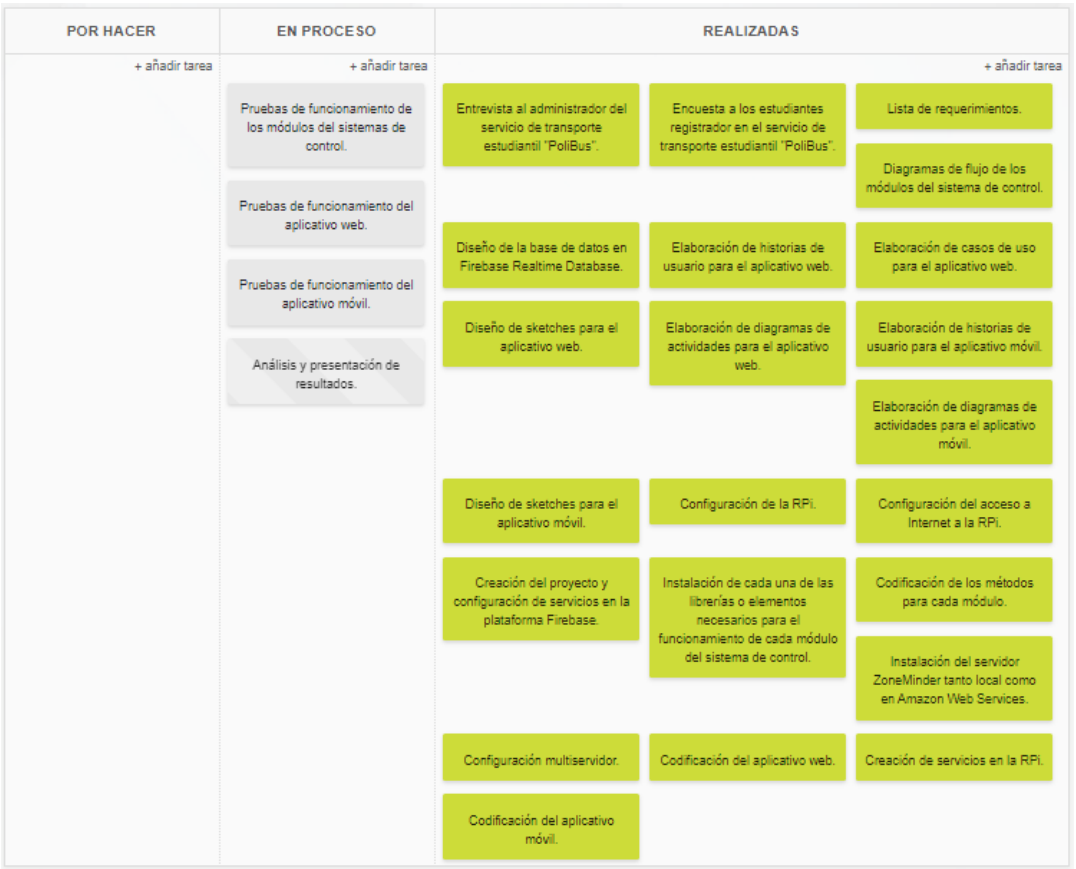


Figura 4.1. Tablero Kanban para la fase de resultados y discusión

4.2. PRUEBAS DE LA FUNCIONALIDAD DEL SISTEMA DE CONTROL

Para llevar a cabo las pruebas del funcionamiento de cada módulo que forma parte del sistema de control, primero se solicitó por medio del Ph.D. Fernando Carrera los permisos respectivos ante las autoridades de la Escuela Politécnica Nacional para instalar en una unidad de transporte estudiantil el sistema de control. Teniendo una respuesta favorable se asignó al presente proyecto la línea 2 correspondiente al sector de Guamaní.

A continuación, se detallan las pruebas locales y remotas junto con los resultados obtenidos e instalación de cada uno de los módulos que forman parte del sistema de control en la unidad de transporte.

4.2.1. PRUEBAS DE FUNCIONAMIENTO DEL MÓDULO GPS

Cuando el módulo está funcionando correctamente obtiene las coordenadas de ubicación y a continuación las envía a su nodo correspondiente. En la Tabla 4.1 se indica la prueba realizada para el módulo, en la Figura 4.2 se muestra las coordenadas obtenidas por el módulo en la unidad remota y en la Figura 4.3 la información anteriormente obtenida se almacena en la base.

Tabla 4.1. Prueba módulo GPS

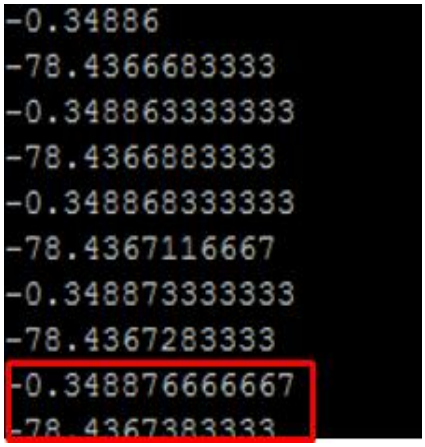


Figura 4.2. Coordenadas obtenidas en la RPI.

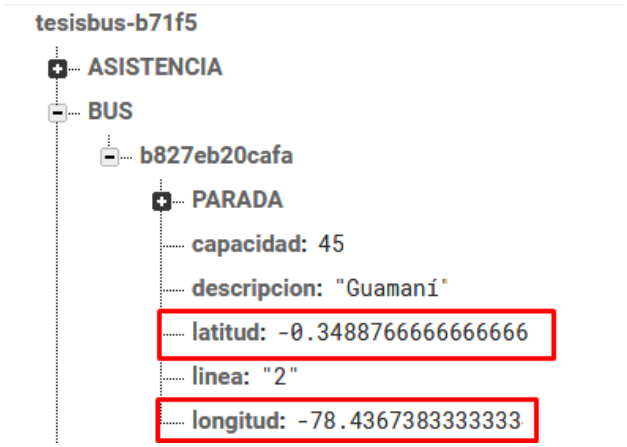


Figura 4.3. Coordenadas obtenidas almacenadas en la base de datos.

En el caso de que el módulo se encuentre sin cobertura se obtiene el mensaje de la Figura 4.3.

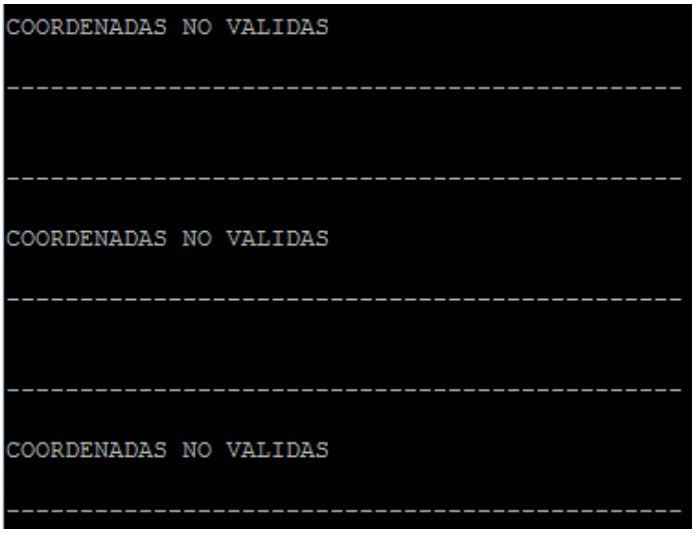


Figura 4.4. Módulo GPS sin cobertura

Para la ubicación de este módulo dentro de la unidad de transporte se verificó que en el lugar (ver Figura 4.5) se puedan obtener coordenadas válidas.

Figura 4.5. Ubicación de módulo GPS dentro de la unidad de transporte

Al realizar las pruebas conjuntamente con los aplicativos se observó que no existió problema alguno.

4.2.2. PRUEBAS DE FUNCIONAMIENTO DEL MÓDULO HUELLA

Para verificar el funcionamiento de este módulo, se trabajó conjuntamente con el aplicativo web ya que previamente se necesita registrar una huella en el sistema.

Cuando ingresa un pasajero registrado, éste coloca su huella dactilar en el lector óptico del sensor biométrico, dependiendo del caso (1,2 o 3) se tendrá como resultado los siguientes mensajes (ver Figura 4.6). Para el caso 1, el usuario coloca su huella dactilar y no está registrado en el sistema o tuvo una lectura errónea de su huella, para el caso 2 el usuario si está registrado en el sistema y posteriormente se añade el pasajero dentro del nodo de

la ruta a la que correspondiente, a la vez en la base de datos para la ramificación *estado* su valor será igual a *True* (ver Figura 4.7). Para el caso 3, el usuario registra que se bajó de la unidad de transporte por lo que seguidamente se modifica el valor de la ramificación *estado* a *False* (ver Figura 4.7). Este módulo también permitirá llevar a cabo el control de asistencia del chofer de la unidad de transporte.

Figura 4.6. Mensaje en RPi de los casos en módulo huella

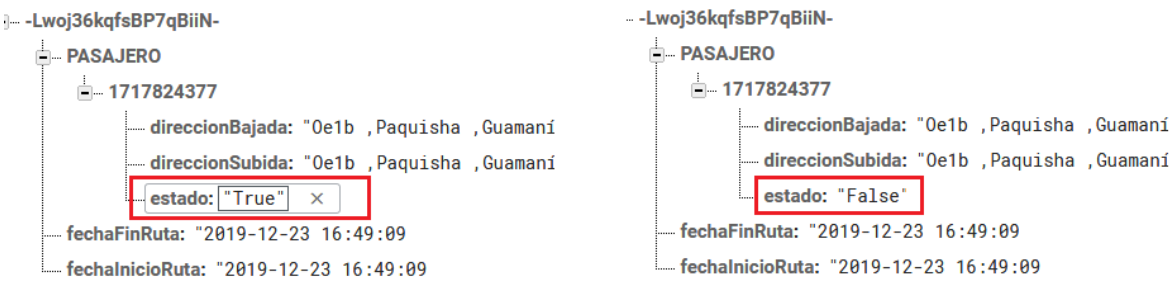


Figura 4.7. Resultado en Firebase de los casos en módulo huella

Para la instalación de este módulo se tomó en consideración la ubicación de la puerta de ingreso a la unidad de transporte, por tal razón se instaló cerca de la misma (ver Figura 4.8).

Figura 4.8. Ubicación del módulo huella

Tomando en cuenta la ubicación del módulo se observó que ya en la práctica se generará una incomodidad para el chofer cuando éste tenga que registrar su asistencia pues no es práctico detener la unidad y acercarse al módulo ubicado a el ingreso de la unidad, por tal razón se decidió integrar otro módulo cerca del chofer y así pueda fácilmente realizar su control de asistencia. Por lo que en este punto se contará con dos módulos uno dedicado al control de acceso para los estudiantes y el otro para el control de asistencia del chofer.

En cuanto al código referente de este módulo el cambio que se realizo fue:

a. Crear dos hilos, uno para estudiantes y el otro para el chofer (ver Código 4.1).
b. Eliminar el método de comparación del rol, debido a que se separa las funciones de cada módulo huella.

```
#Inicio del hilo para el modulo huella Estudiante
hilo_huella_estudiante= threading.Thread(target=leer_huellaE)
hilo_huella_estudiante.start()
#Inicio del hilo para el modulo huella Chofer
hilo_huella_chofer= threading.Thread(target=leer_huellaCh)
hilo_huella_chofer.start()
```

Código 4.1. Corrección hilos

En la Tabla 4.2 se presenta la conexión de cada módulo huella después de la adición de un sensor biométrico a la unidad remota.

Tabla 4.2. Conexión de los módulos huella con los diferentes elementos que intervienen

MÓDULO HUELLA			CONVERSOR	RASPBERRY
Estudiante (Control de acceso)	**Sensor biométrico**	Vcc	5V	USB1
		Tx	Rx	
		Rx	Tx	
		GND	GND	
	Bocina		-----	Pin 12
Chofer (Control de asistencia)	**Sensor biométrico**	Vcc	5V	USB0
		Tx	Rx	
		Rx	Tx	
		GND	GND	
	Bocina		-----	Pin 15

A continuación, se presentan los cambios realizados para el diagrama de flujo:

a. Diagrama de flujo del módulo huella para el estudiante (ver Figura 4.9).

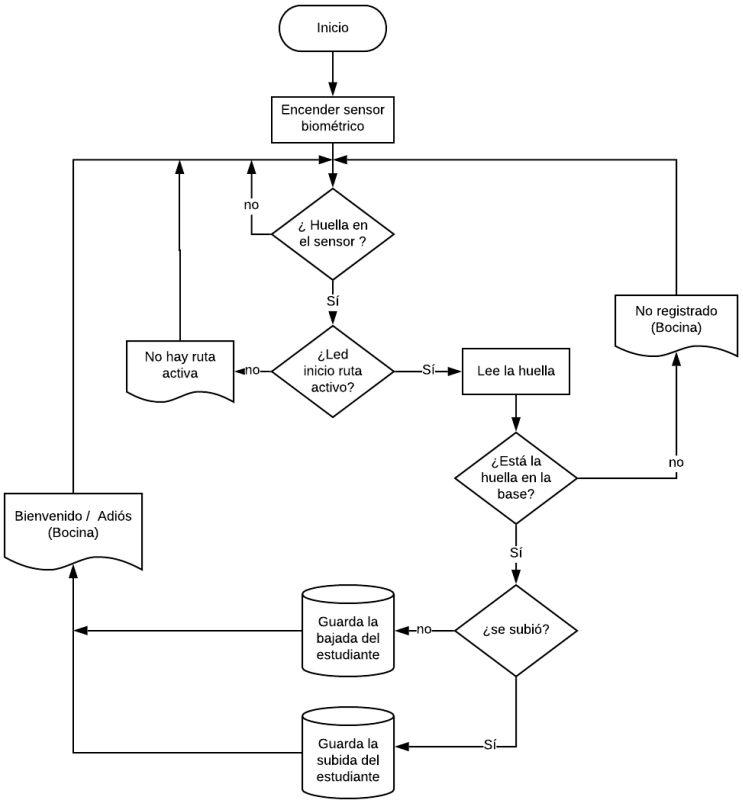


Figura 4.9. Modificación diagrama de flujo módulo huella para el estudiante

b. Diagrama de flujo del módulo huella para el chofer (ver Figura 4.10).

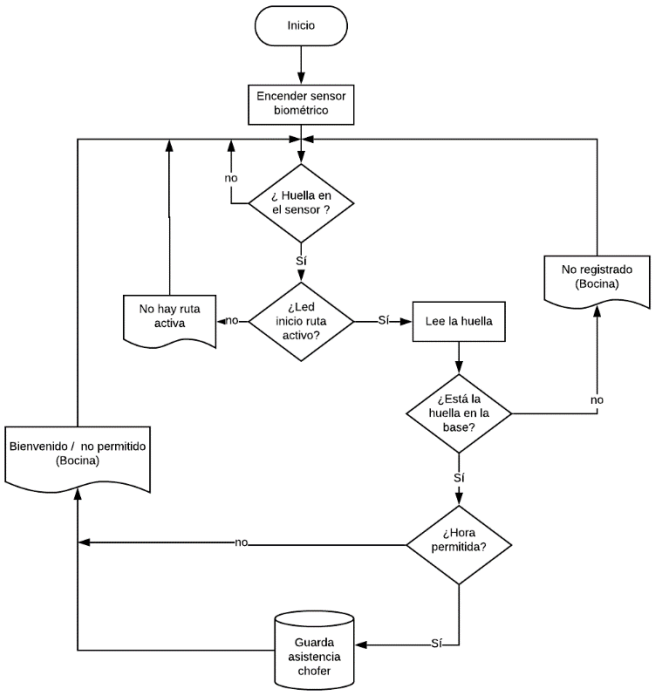


Figura 4.10. Modificación diagrama de flujo módulo huella para el chofer

En la Figura 4.11 se muestra la ubicación de los dos módulos con los cambios realizados. Luego de la corrección realizada se logró observar que el chofer ya puede sin dificultad registrar su asistencia separado del control de acceso de los estudiantes.

Figura 4.11. Módulos huella separados

4.2.3. PRUEBAS DE FUNCIONAMIENTO DEL MÓDULO CONTROL ELECTRÓNICO

Para determinar la funcionalidad del módulo se llevaron a cabo a las siguientes pruebas locales:

- Para iniciar una ruta se presiona el pulsador de INICIO teniendo como resultado: la creación de una nueva ruta, se encenderá el indicador (led verde), en la RPi se muestra el siguiente mensaje (ver Figura 4.12) y en la base de datos se creará el nodo con los datos correspondientes (ver Figura 4.13).

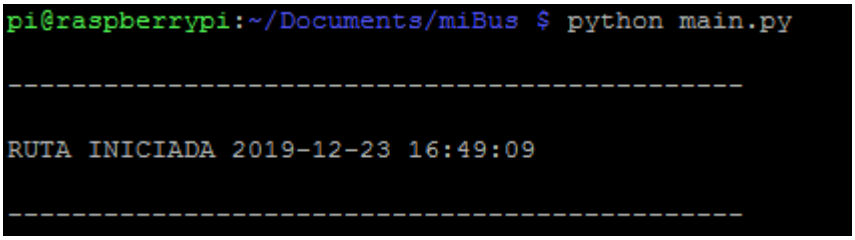


Figura 4.12. Mensaje RPi en inicio de ruta

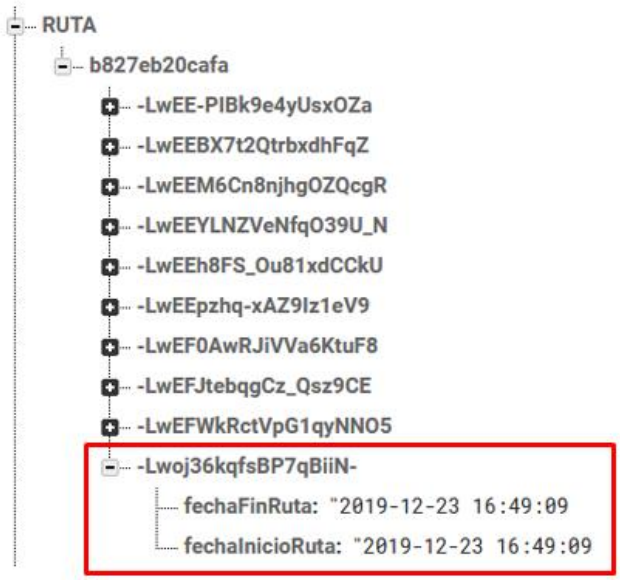


Figura 4.13. Resultado en Firebase de inicio de ruta

- Para finalizar una ruta se presiona el pulsador de FIN teniendo como resultado: la finalización de la ruta creada, se encenderá el indicador (led amarillo), en la RPi se muestra el siguiente mensaje (ver Figura 4.14) y en la base de datos se modificará el nodo correspondiente a esta ruta (ver Figura 4.15).

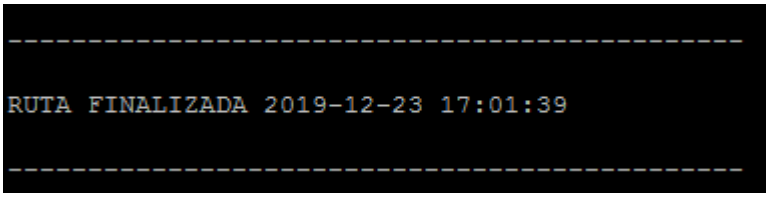


Figura 4.14. Mensaje RPi en fin de ruta

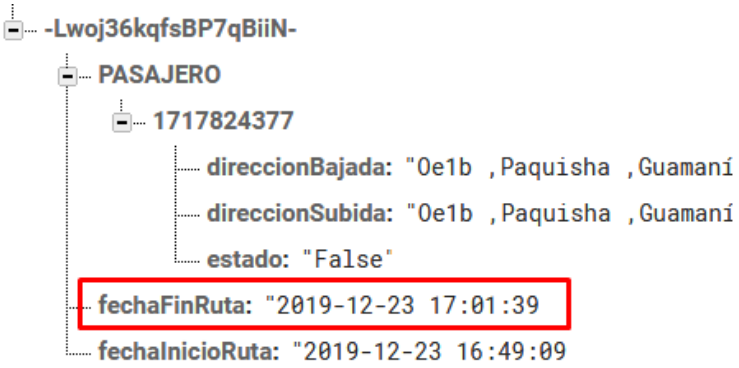


Figura 4.15. Resultado en Firebase de fin de ruta

- Para la opción de emergencia se presiona el pulsador de EMERGENCIA teniendo como resultado: se encenderá el indicador (led rojo), en la RPi se muestra el siguiente mensaje (ver Figura 4.16) y con el fin de notificar una emergencia, en la base de datos se creará el nodo correspondiente, este nodo tiene la ramificación *estado*, la cual es modificada una vez se hayan cumplido 5 segundos codificados en el hilo correspondiente en la RPi (ver Figura 4.17).

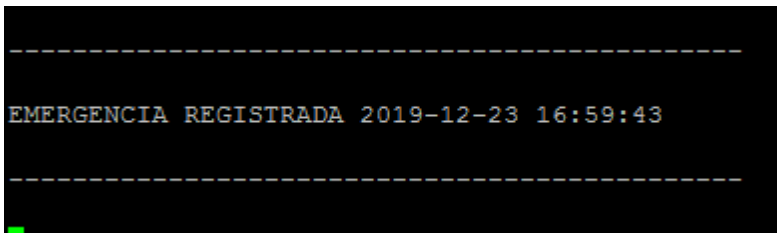


Figura 4.16. Mensaje RPi en emergencia

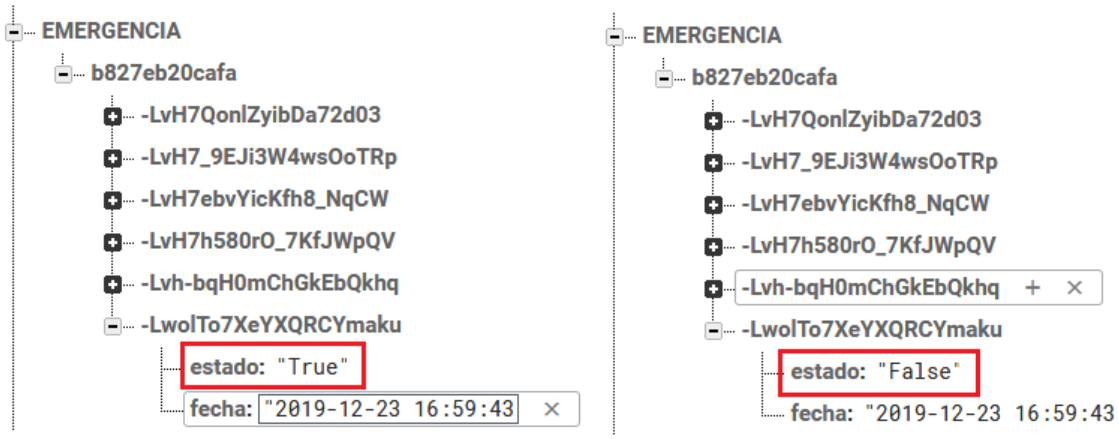


Figura 4.17. Resultado en Firebase en emergencia

Este módulo fue ubicado en la ventana más cercana (izquierda) del asiento del chofer (ver Figura 4.18) debido a que en esta ubicación es más factible la interacción con el módulo.

Figura 4.18. Ubicación del módulo control electrónico

Una vez instalado el módulo se procedió a realizar las pruebas remotas (ver Figura 4.19) de las cuales se pudieron concluir que no tuvieron diferencias a comparación de los resultados de las pruebas locales, por lo que el módulo cumple con la funcionalidad planteada.

Figura 4.19. Prueba remota módulo de control electrónico

4.2.4. UBICACIÓN DEL MÓDULO CÁMARA

Para la ubicación de este módulo se consideró que la captura de imagen de la cámara abarque la mayor cantidad de asientos de la unidad de transporte (ver Figura 4.20).

Figura 4.20. Ubicación módulo cámara

4.3. PRUEBAS DE LA FUNCIONALIDAD DEL APLICATIVO WEB

Se mostrarán los resultados de las pruebas realizadas para cada uno de los formularios que comprenden el aplicativo web. Cabe recalcar que estas pruebas se realizaron de forma remota, puesto que se necesita información en la base de datos, es decir cuando la unidad de transporte está en movimiento y con usuarios.

4.3.1. PRUEBAS DE FUNCIONAMIENTO PARA LOGIN

Para realizar la prueba del funcionamiento del formulario definido para el inicio de sesión previamente se ingresó un usuario administrador por defecto.

- Inicio de sesión del usuario

En la Figura 4.21 se muestra la interfaz del formulario login, este formulario acepta como datos el correo y contraseña de un administrador.

Figura 4.21. Interfaz login

Si el usuario ingresa los datos correctos se ingresará al aplicativo web, caso contrario mostrará los siguientes mensajes:

Caso ingreso de correo no registrado en el sistema (ver Figura 4.22).

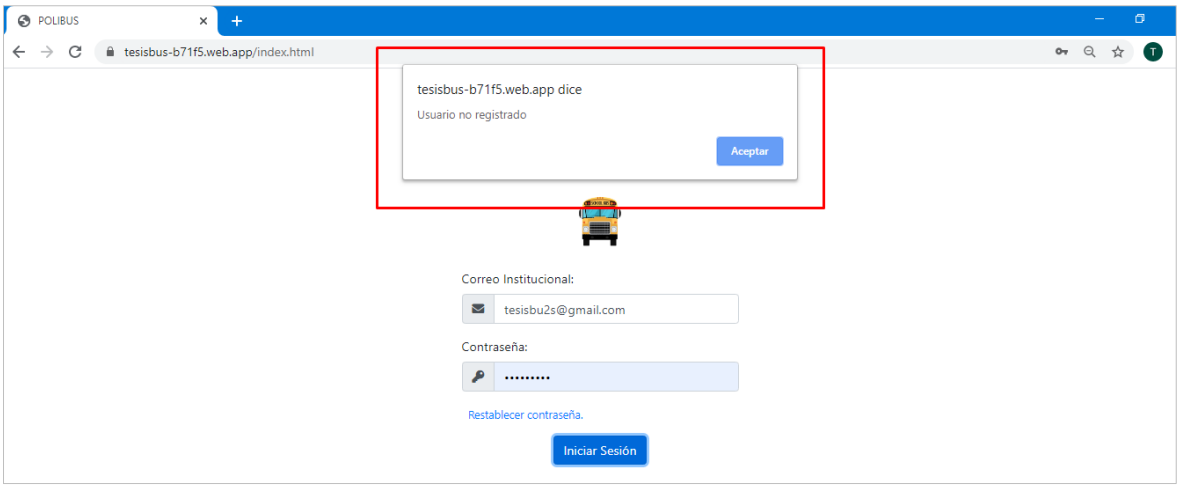


Figura 4.22. Mensaje correo no registrado

Caso ingreso de contraseña incorrecta (ver Figura 4.23).

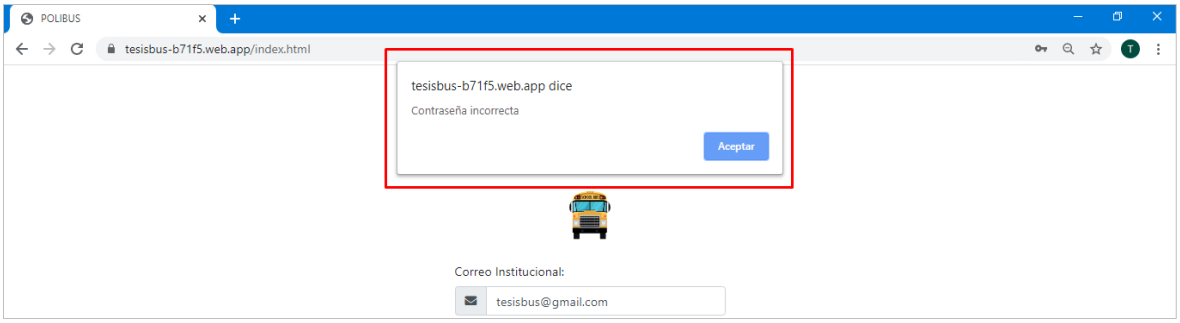


Figura 4.23. Mensaje contraseña incorrecta

- Restablecimiento de contraseña.

Para esta opción se debe ingresar un correo válido en la página de login, seguidamente se enviará un mail al correo (ver Figura 4.24).

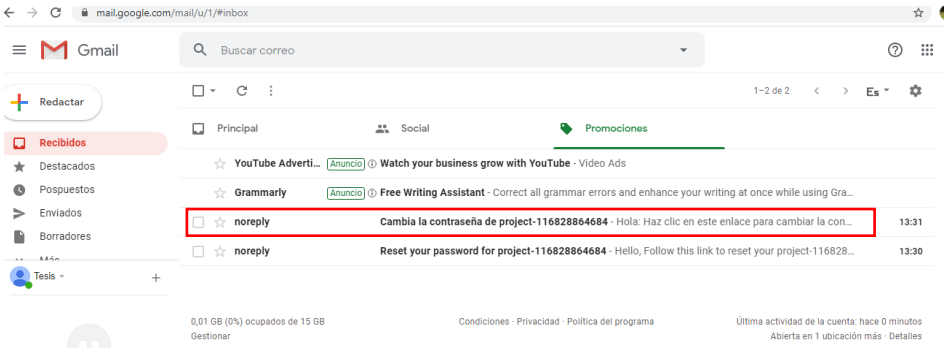


Figura 4.24. Recepción del correo para restablecer contraseña

Una vez ingresado en el link se cargará el siguiente formulario (ver Figura 4.25) que permitirá restablecer la contraseña.

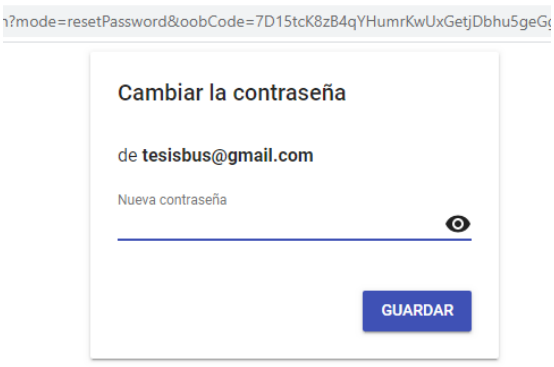


Figura 4.25. Formulario para restablecer contraseña

4.3.2. PRUEBAS DE FUNCIONAMIENTO PARA PRINCIPAL

Este formulario muestra la información solo si en ese preciso momento se está realizando una ruta caso contrario no.

En este formulario se debe elegir el número de unidad de transporte para llenar la información para las herramientas visuales. En la Figura 4.26 se puede apreciar el resultado, se observa la hora de inicio de la ruta, el número total de estudiantes recogidos, el número de estudiantes que se encuentran actualmente en el bus, el número de asientos disponibles y también se puede observar la posición actual del bus en el mapa de Google esta posición ira cambian de acuerdo a las coordenadas que se genere el módulo GPS.

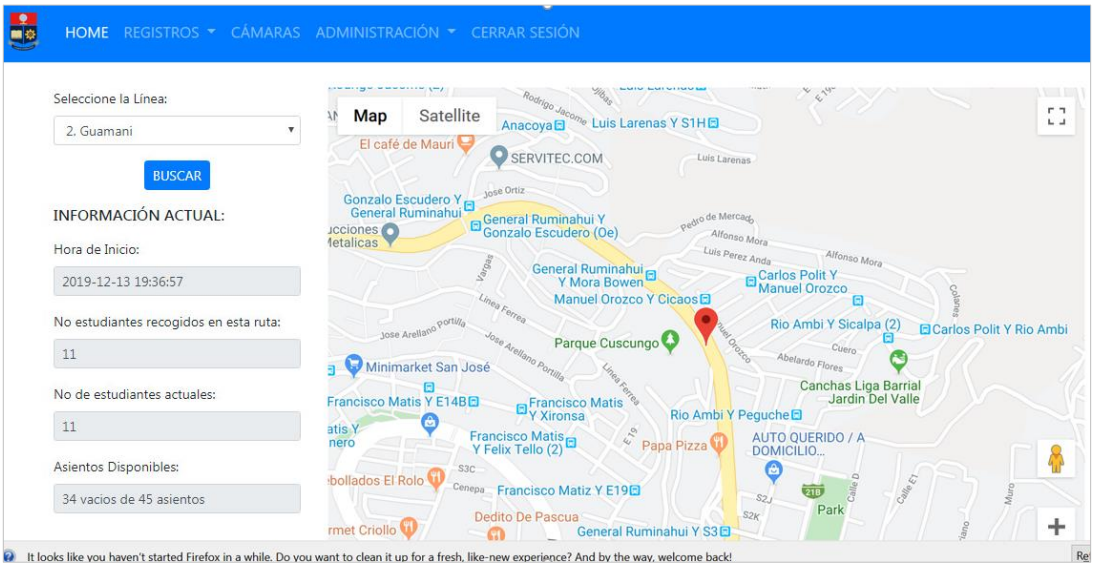


Figura 4.26. Resultado página principal

Cabe mencionar que para usar el API de Google Maps de forma gratuita se tiene un periodo de prueba de un año, el mismo que si caduca mostrará la interfaz del mapa (ver Figura 4.27). Para solucionar este inconveniente se generó una cuenta nueva en Google Maps.

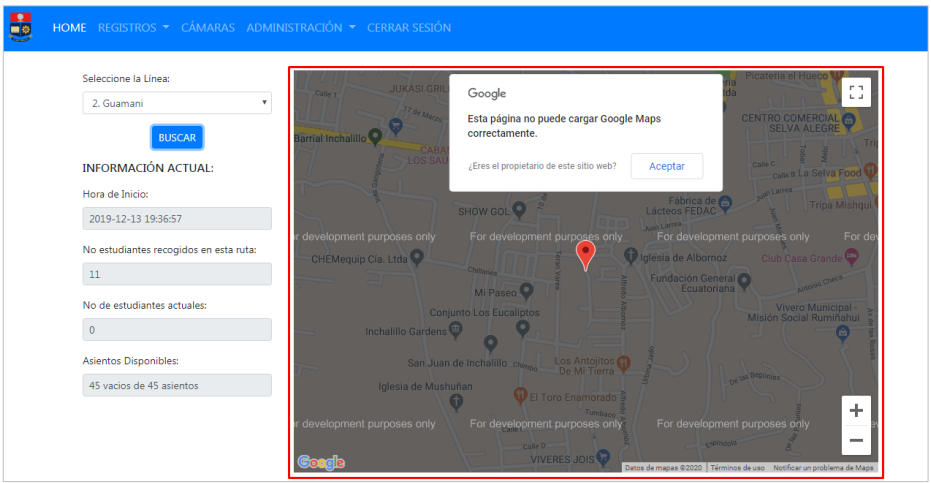


Figura 4.27. Key de API de Google Maps caducada

4.3.3. PRUEBAS DE FUNCIONAMIENTO PARA ADMINISTRACIÓN

Al ingresar a la opción de administración de la barra de navegación dependiendo de la gestión que se desee realizar se puede administrar a: usuarios, buses y carreras.

a. Administración de usuarios.

En esta opción se podrá visualizar la Figura 4.28, donde se apreciará la lista de los usuarios que se han registrado dentro del sistema.

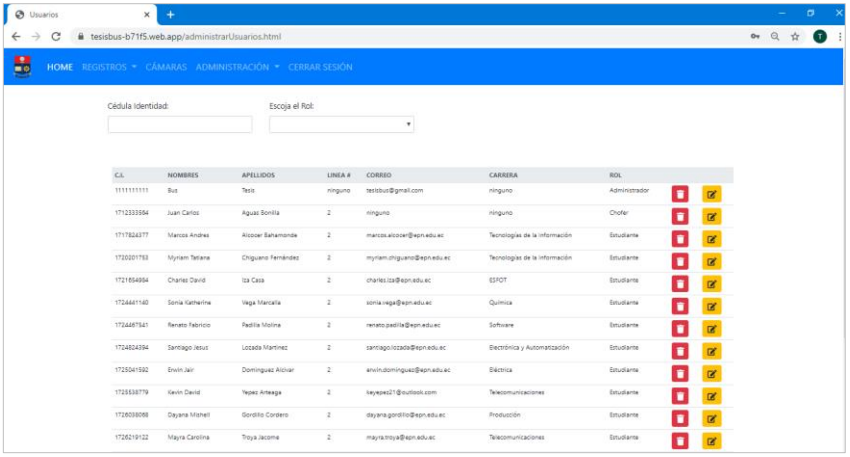


Figura 4.28. Formulario Administración de usuarios

Para crear un usuario del tipo estudiante o chofer primero se ingresa una cédula de identidad, luego se elige el rol seguidamente se activa el botón de huella, el cual permite prender al módulo registro para enrolar al usuario (ver Figura 4.29).

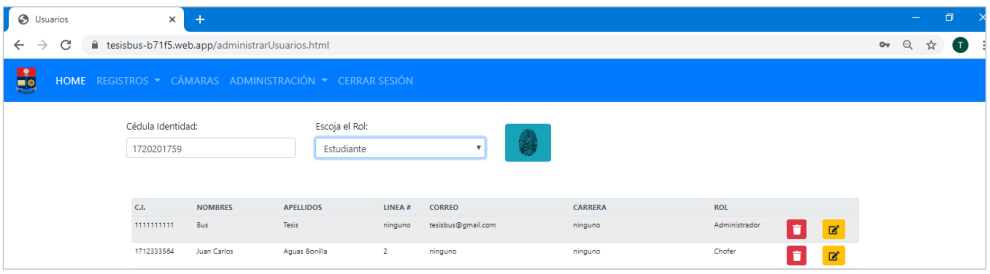


Figura 4.29. Creación de usuario estudiante o chofer

Si la lectura de la huella se realiza correctamente por medio del módulo registro, seguido se abrirá la ventana (modal) de la Figura 4.30, lugar donde se ingresan los datos personales del usuario a registrar para finalmente guardarlos en la base de datos.

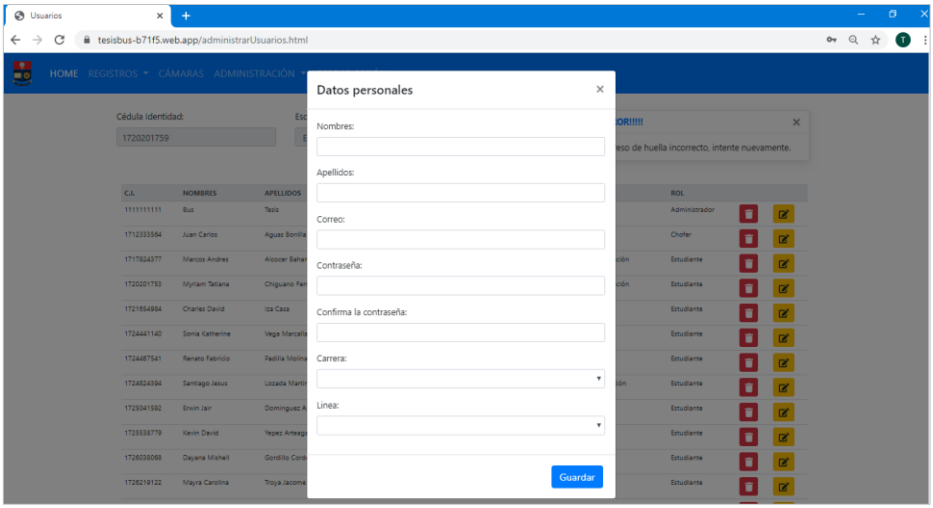


Figura 4.30. Modal datos estudiante o chofer

Si la lectura de la huella no fue correcta se mostrará un mensaje de error (ver Figura 4.31).

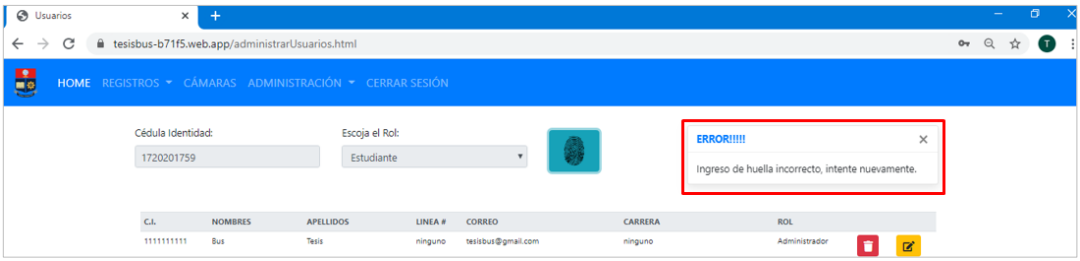


Figura 4.31. Error al leer la huella

Para crear un usuario administrador se ingresa de igual manera una cédula de identidad válida seguido del rol, la diferencia a los demás roles se debe a que no se solicita el registro de huella dactilar, simplemente activará el botón Siguiente (Ver Figura 4.32), para luego mostrar una ventana donde se ingresan los datos pertenecientes al usuario (ver Figura 4.33) para finalmente guardarlos en la base de datos.

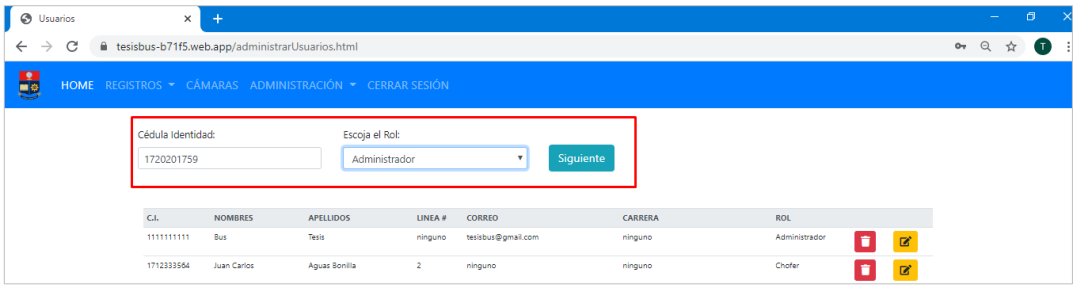


Figura 4.32. Creación usuario administrador

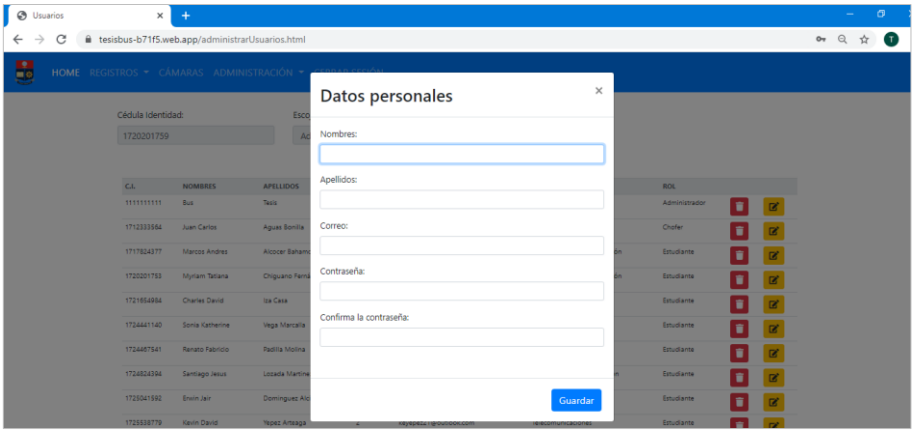


Figura 4.33. Modal datos administrador

Para la eliminación o modificación de un usuario se cuenta que sus botones correspondientes, se elige el usuario a eliminar o modificar de la tabla (ver Figura 4.34) se procede a realizar la acción y seguidamente se mostrará una ventana de confirmación (ver Figura 4.35).

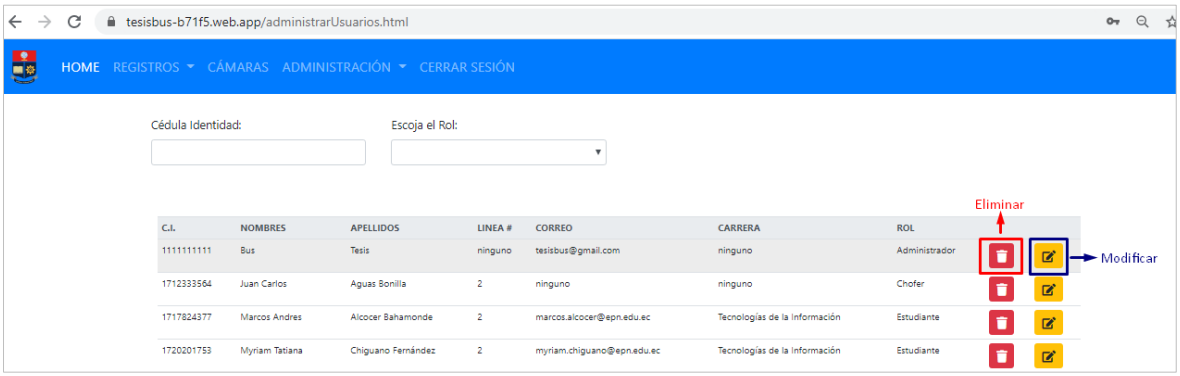


Figura 4.34. Eliminación y modificación de usuarios

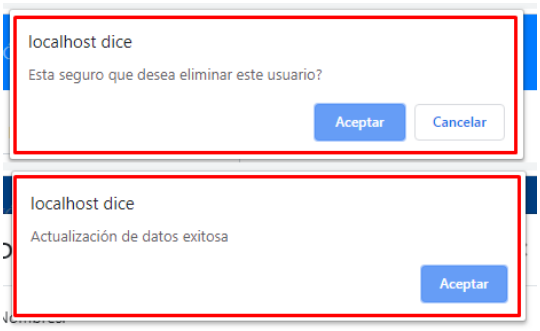


Figura 4.35. Mensajes de confirmación de eliminar o editar

4.3.4. PRUEBAS DE FUNCIONAMIENTO DEL REGISTRO

A continuación, se muestran las pruebas realizadas para la opción Registro de la barra de navegación.

a. Registro chofer.

En la Figura 4.36 se muestra el resultado del registro generado del control de asistencia del chofer, para obtener el resultado previamente se elige el día que se desea realizar la consulta y finalmente el aplicativo muestra el registro. Se observa las abreviaturas de los días de la semana seguido de un número 1 o 2, el 1 corresponde a la asistencia de la mañana y el 2 a la noche del mismo día.

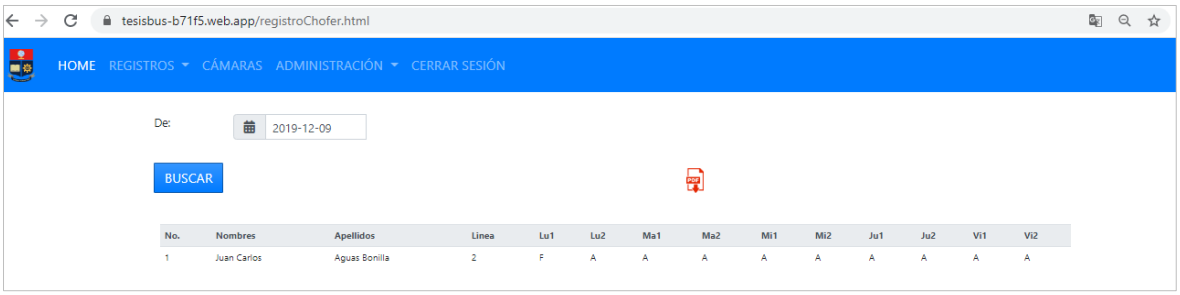


Figura 4.36. Registro de asistencia del chofer

En cuanto a la generación del archivo PDF previamente se debe generar un registro para luego con el botón exportar (ver Figura 4.37) obtener el archivo, este archivo se almacena en la carpeta de descargas del ordenador. En la Figura 4.38 se observa el archivo .pdf generado.

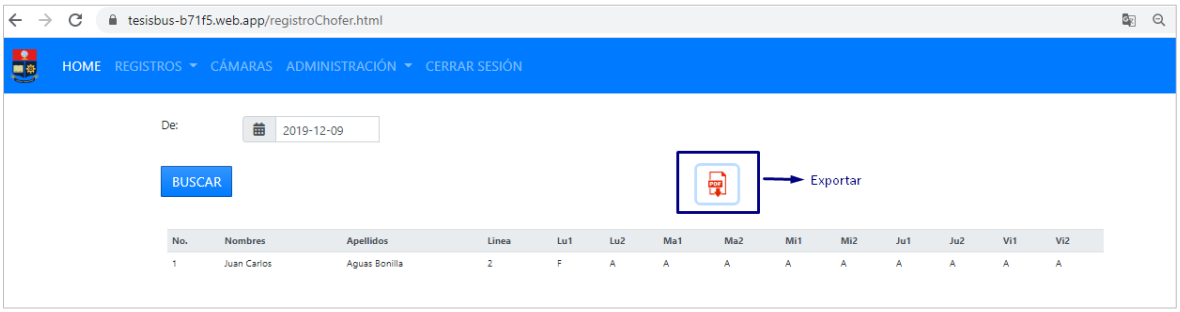


Figura 4.37. Botón exportar archivo .pdf

Figura 4.38. Archivo .pdf generado

b. Registro estudiante.

La Figura 4.39 muestra el resultado generado para obtener un registro de asistencia diario del estudiante, para generar esta lista se debe previamente elegir el número de la línea y el día que se desee generar el registro, también se puede observar que se indica la dirección del lugar donde se subió y donde se bajó el estudiante en la ruta realizada.

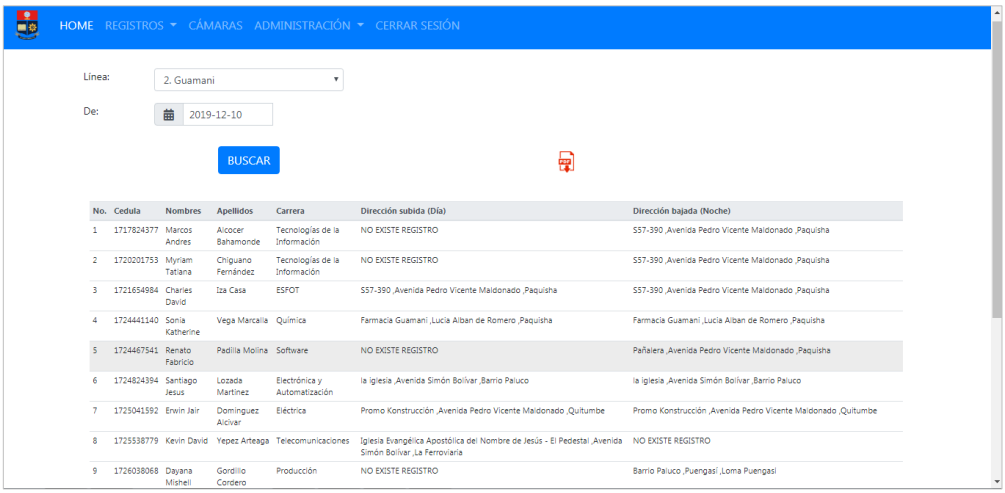
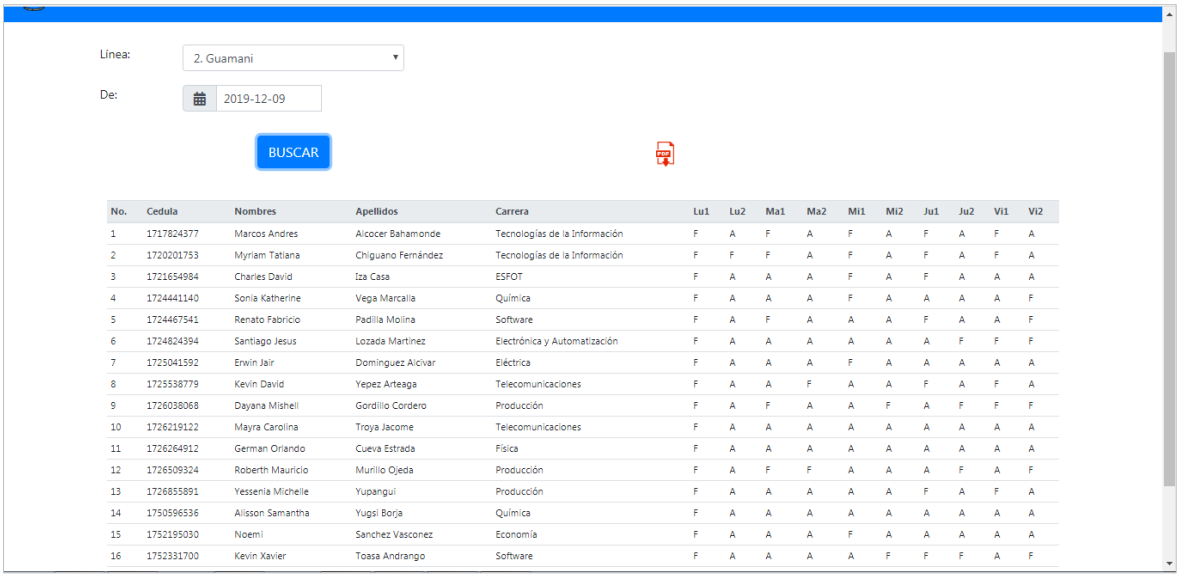


Figura 4.39. Registro diario

La Figura 4.40 muestra el resultado generado para obtener un registro de asistencia semanal del estudiante, para generar esta lista se debe elegir el número de línea de transporte estudiantil y el primer día de la semana del calendario. El registro muestra la información del estudiante, así como la semana indicando con una A que asistió y con F

que faltó, las abreviaturas de los días de la semana están seguidos por un número 1 o 2, donde1 significa el recorrido de la mañana y 2 el recorrido de la noche del mismo día.



No.	Cedula	Nombres	Apellidos	Carrera	Lu1	Lu2	Ma1	Ma2	Mi1	Mi2	Ju1	Ju2	Vi1	Vi2
1	1717824377	Marcos Andres	Alcocer Bahamonde	Tecnologías de la Información	F	A	F	A	F	A	F	A	F	A
2	1720201753	Myriam Tatiana	Chiguano Fernández	Tecnologías de la Información	F	F	F	A	F	A	F	A	F	A
3	1721654984	Charles David	Iza Casa	ESFOT	F	A	A	A	F	A	F	A	A	A
4	1724441140	Sonia Katherine	Vega Marcalla	Química	F	A	A	A	F	A	A	A	A	F
5	1724467541	Renato Fabricio	Padilla Molina	Software	F	A	F	A	A	A	F	A	A	F
6	1724824394	Santiago Jesus	Lozada Martinez	Electrónica y Automatización	F	A	A	A	A	A	A	F	F	F
7	1725041592	Erwin Jair	Dominguez Alcivar	Eléctrica	F	A	A	A	F	A	A	A	A	A
8	1725538779	Kevin David	Yepez Arteaga	Telecomunicaciones	F	A	A	F	A	A	F	A	F	A
9	1726038068	Dayana Mishell	Gordillo Cordero	Producción	F	A	F	A	A	F	A	F	F	F
10	1726219122	Mayra Carolina	Troya Jacome	Telecomunicaciones	F	A	A	A	A	A	A	A	A	A
11	1726264912	German Orlando	Cueva Estrada	Física	F	A	A	A	A	A	A	A	A	A
12	1726509324	Roberth Mauricio	Murillo Ojeda	Producción	F	A	F	F	A	A	A	F	A	F
13	1726855891	Yessenia Michelle	Yupangui	Producción	F	A	A	A	A	A	F	A	F	A
14	1750596536	Alisson Samantha	Yugsi Borja	Química	F	A	A	A	A	A	A	A	A	A
15	1752195030	Noemi	Sanchez Vasconez	Economía	F	A	A	A	F	A	A	A	A	A
16	1752331700	Kevin Xavier	Toasa Andrango	Software	F	A	A	A	A	F	F	F	A	F

Figura 4.40. Registro semanal

Para la exportación de archivo .pdf la opción semanal como diaria cuentan con un botón exportar que genera el archivo .pdf. En la Figura 4.41 se indica el resultado al exportar el archivo .pdf para un registro semanal.

ESCUELA POLITÉCNICA NACIONAL

CONTROL DE ASISTENCIA ESTUDIANTE

REGISTRO SEMANAL "PoliBus"

Línea: 2 **Descripción: Guamaní** **Capacidad: 45** **Fecha: 2019-12-09**

Número de estudiantes registrados	Lu1	Lu2	Ma1	Ma2	Mi1	Mi2	Ju1	Ju2	Vi1	Vi2
17	0	16	12	15	10	15	10	13	10	11

No.	Cedula	Nombres	Apellidos	Carrera	Lu1	Lu2	Ma1	Ma2	Mi1	Mi2	Ju1	Ju2	Vi1	Vi2
1	1717824377	Marcos Andres	Alcocer Bahamonde	Tecnologías de la Información	F	A	F	A	F	A	F	A	F	A
2	1720201753	Myriam Tatiana	Chiguano Fernández	Tecnologías de la Información	F	F	F	A	F	A	F	A	F	A
3	1721654984	Charles David	Iza Casa	ESFOT	F	A	A	A	F	A	F	A	A	A
4	1724441140	Sonia Katherine	Vega Marcalla	Química	F	A	A	A	F	A	A	A	A	F
5	1724467541	Renato Fabricio	Padilla Molina	Software	F	A	F	A	A	A	F	A	A	F
6	1724824394	Santiago Jesus	Lozada Martinez	Electrónica y Automatización	F	A	A	A	A	A	A	F	F	F
7	1725041592	Erwin Jair	Dominguez Alcivar	Eléctrica	F	A	A	A	F	A	A	A	A	A
8	1725538779	Kevin David	Yepez Arteaga	Telecomunicaciones	F	A	A	F	A	A	F	A	F	A
9	1726038068	Dayana Mishell	Gordillo Cordero	Producción	F	A	F	A	A	F	A	F	F	F
10	1726219122	Mayra Carolina	Troya Jacome	Telecomunicaciones	F	A	A	A	A	A	A	A	A	A
11	1726264912	German Orlando	Cueva Estrada	Física	F	A	A	A	A	A	A	A	A	A
12	1726509324	Roberth Mauricio	Murillo Ojeda	Producción	F	A	F	F	A	A	A	F	A	F
13	1726855891	Yessenia Michelle	Yupangui	Producción	F	A	A	A	A	A	F	A	F	A
14	1750596536	Alisson Samantha	Yugsi Borja	Química	F	A	A	A	A	A	A	A	A	A
15	1752195030	Noemi	Sanchez Vasconez	Economía	F	A	A	A	F	A	A	A	A	A

Figura 4.41. Registro semanal de control de asistencia de los estudiantes

4.3.5. PRUEBAS DE FUNCIONAMIENTO PARA CÁMARA

En la Figura 4.42 se indica el resultado obtenido al ingresar a la opción de cámara, se puede observar que se está capturando las imágenes generadas en el transporte estudiantil, así como también se puede observar en la Figura 4.43 que se están generado eventos que son almacenados en la RPi y se muestran en la interfaz de ZoneMinder.

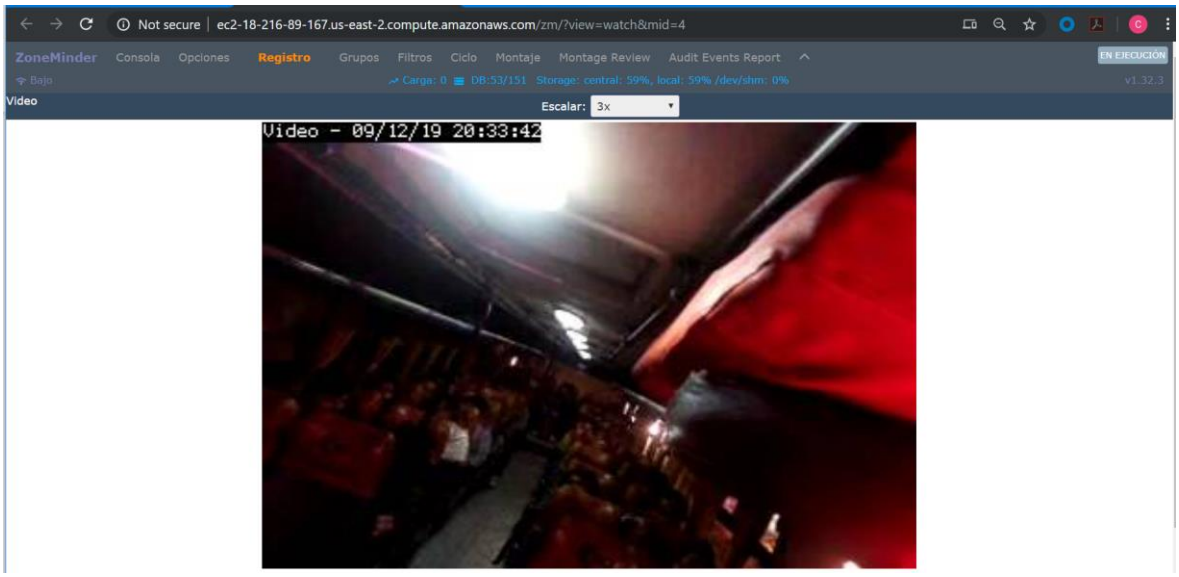


Figura 4.42. Resultado de captura de la cámara

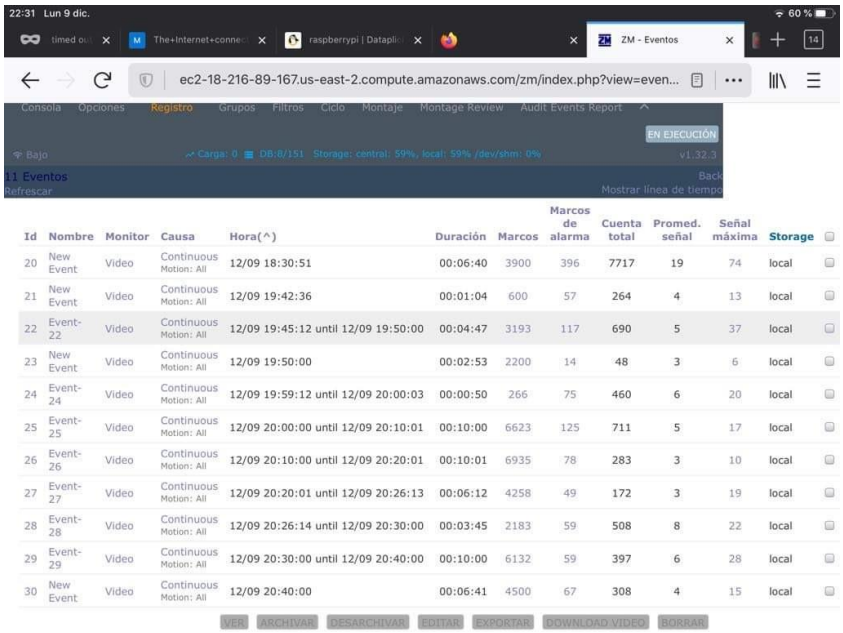


Figura 4.43. Eventos generados por la cámara

4.4. PRUEBAS DE LA FUNCIONALIDAD DEL APLICATIVO MÓVIL

En esta sección se presentan los resultados de las pruebas realizadas para cada pantalla correspondiente al aplicativo móvil de la misma forma que en el aplicativo web, es decir que la unidad de transporte genere información.

4.4.1. PRUEBAS DE FUNCIONAMIENTO PARA ROL ADMINISTRADOR

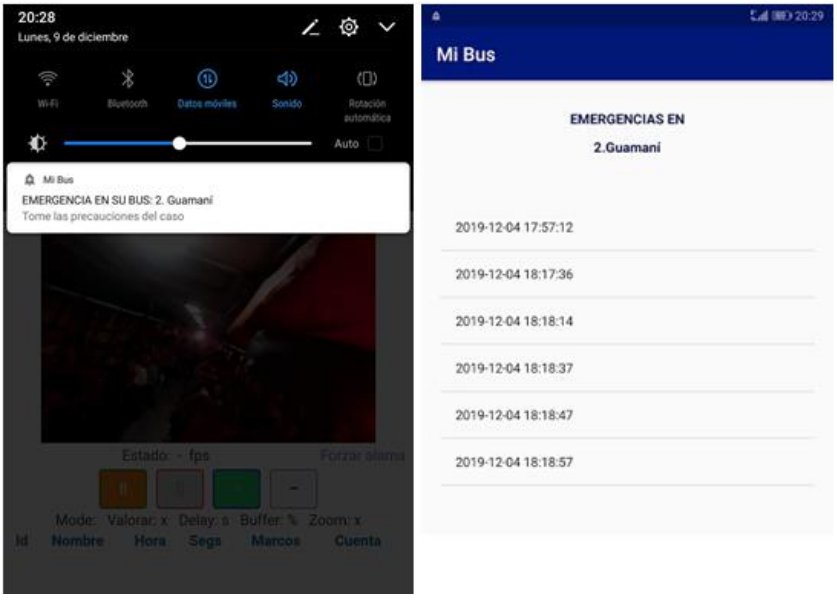
La Figura 4.44 muestra el resultado obtenido de la prueba realizada para la pantalla login, luego de ingresar el correo registrado junto con la contraseña, si estos son correctos de mostrará un mensaje *Bienvenido*.

Figura 4.44. Resultado de pantalla login

Una vez se haya ingresado a la aplicación se observa la pantalla control (ver Figura 4.45), en la que se muestra información de inicio de ruta, asientos libres y ubicación del transporte estudiantil en el mapa de Google, la cual va cambiando conforme al movimiento del transporte.

Figura 4.45. Resultado de pantalla control

En cuanto a módulo de control electrónico si se activa una emergencia por parte del chofer, el resultado se muestra en la Figura 4.46, parte a) la notificación que se genera en el aplicativo móvil, en el caso de presionar la notificación automáticamente se redirige a la ventana que se indica en la parte b) donde se observa un registro de emergencias de la unidad.



a. b.

Figura 4.46. Resultado de la recepción de notificaciones

La Figura 4.47 muestra el resultado obtenido para la opción de cámaras, en esta interfaz se enlistan las cámaras o monitores registrados. Para este caso se tiene una cámara, puesto que solo existe una unidad de transporte en el sistema.

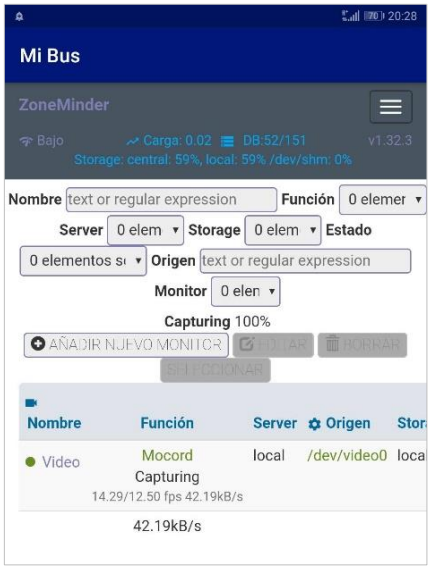


Figura 4.47. Resultado de interfaz de monitores

Las imágenes captadas por la cámara se indican la Figura 4.48 concluyendo que la aplicación móvil junto con la cámara ubicada en la unidad de transporte estudiantil funciona con un retardo de 10 segundos y sin generan ningún inconveniente.

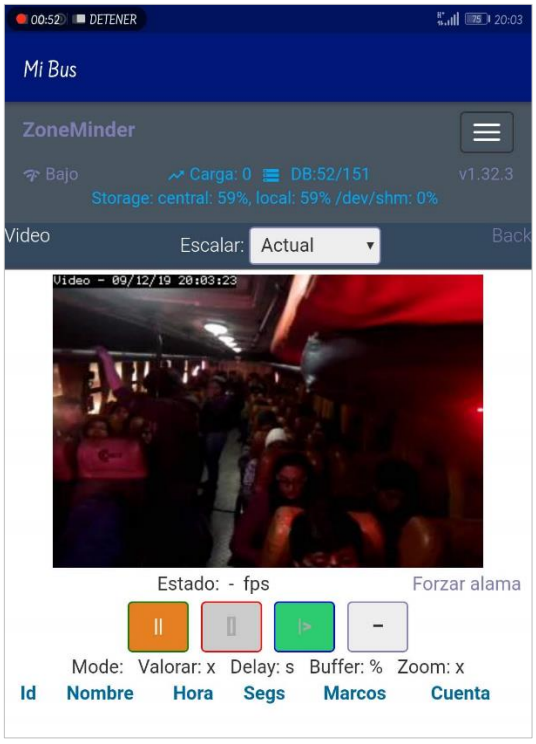


Figura 4.48. Resultado cámara

4.4.2. PRUEBAS DE FUNCIONAMIENTO PARA ROL ESTUDIANTE

Luego de realizar el logueo para el caso de un usuario tipo estudiante la interfaz que se visualiza se muestra en la Figura 4.49 indicando la ubicación del transporte, asientos libres y la fecha y hora de inicio de la ruta.

Figura 4.49. Resultado de pantalla información

4.5. RESULTADOS DE ENCUESTAS DE SATISFACCIÓN

La encuesta y entrevistas realizadas a los usuarios involucrados con el prototipo se encuentran en el ANEXO G.

4.5.1. ENTREVISTA AL ADMINISTRADOR

De la entrevista que se realizó a él ahora exadministrador Ph.D. Fernando Carrera del servicio de transporte estudiantil de la EPN, luego de que se realizaron las pruebas del sistema de comunicación que comprende el aplicativo web y móvil. Se concluye que el prototipo cumple con los requerimientos planteados en la fase de diseño.

4.5.2. ENTREVISTA AL CHOFER

Se realizó una entrevista al chofer Sr. Juan Carlos Aguas de la unidad de transporte línea 2 sector Guamaní, el funcionamiento e interacción de la unidad remota no causo ningún inconveniente para su persona.

4.5.3. ENCUESTAS ESTUDIANTES

Se realizaron 15 encuestas de satisfacción debido a que ese fue el número de estudiantes que se registraron en el sistema.

A continuación, en la Tabla 4.3 se muestra un resumen de los resultados obtenidos después de realizar las respectivas pruebas del prototipo tomando como ejes principales a los estudiantes y al aplicativo móvil.

Tabla 4.3. Resultados encuestas de satisfacción a los estudiantes

No.	PREGUNTA	RESPUESTA (%)
1.	¿Cree usted que la información que muestra el aplicativo móvil es de importancia?	
	Si	93.3
	No	0
	Tal vez	6.7
2.	¿La aplicación móvil funcionó correctamente?	
	Si	100
	No	0
3.	¿Qué calificación le daría usted al diseño de la aplicación móvil?	
	Bueno	80
	Regular	20
	Malo	0
4.	¿La aplicación móvil le resultó fácil de utilizar?	
	Sí	100
	No	0
5.	¿Tuvo algún inconveniente en la interacción con el sensor biométrico?	
	Si	20
	No.	80
6.	¿Cuál fue el inconveniente que tuvo?	
	A veces no se leía mi huella.	
	En ciertas ocasiones tenía que ingresar la huella varias veces y era un poco incómodo.	
	Tenía que poner la huella más de una vez.	

De las encuestas realizadas a los estudiantes se puede destacar que el mayor inconveniente que se produjo fue al ingresar la huella al sensor pues se generaban problemas debido a que muchos de los estudiantes dejaban marcas de grasa en el lente o este a su vez estaba sucio por el polvo o algún otro factor que impida que el lente este limpio.

4.6. ANÁLISIS BREVE DE TRÁFICO

Para determinar el ancho de banda necesario para que el prototipo lleve a cabo su funcionalidad se realizó una monitorización del tráfico generado por la interfaz *ppp0* usando el software *iftop* (ver Figura 4.50*),* tomando en consideración una ruta de la noche desde que empieza a tomar pasajeros hasta que la última parada del recorrido que realiza la unidad de transporte.

El Comando 4.1 realiza la monitorización antes mencionada.

```
pi@raspberrypi:~ $ sudo iftop -i ppp0
```

Comando 4.1. Visualizar tráfico en interfaz ppp0

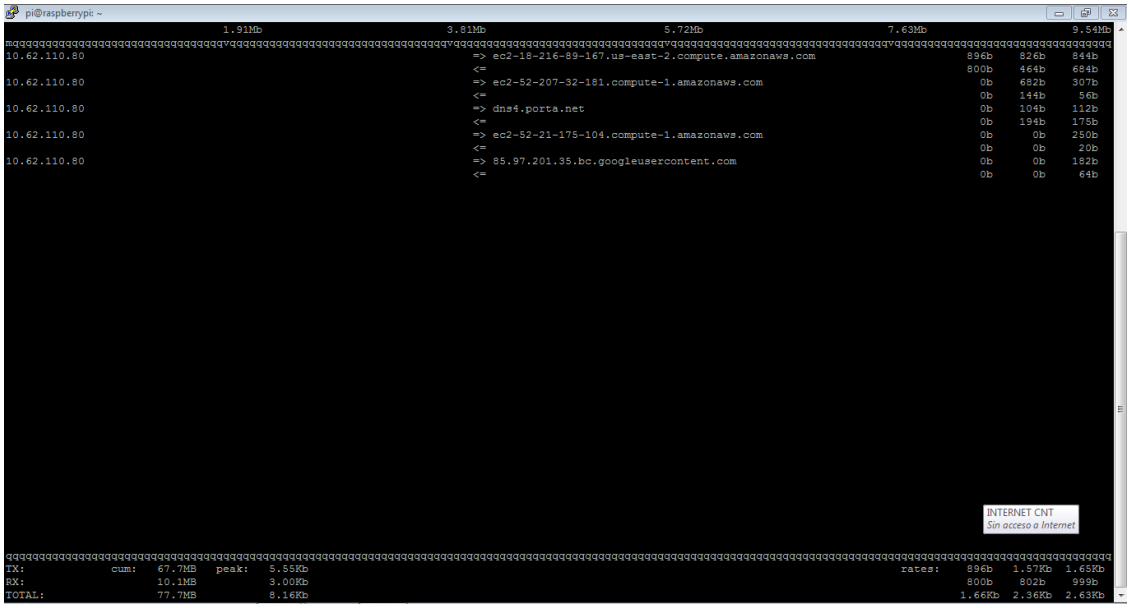


Figura 4.50. Resultado de trafico generado por la interfaz ppp0

De este resultado se puede concluir que el prototipo consume 67.7 [MBps] aproximadamente por cada ruta que realice. Por lo que se sugiere que el prototipo tenga un plan de datos de 3 [GBps], pero debido a que comercialmente la operadora CLARO no oferta un plan que se ajuste a las necesidades se sugiere el Plan Conexión Sin Limite 25.

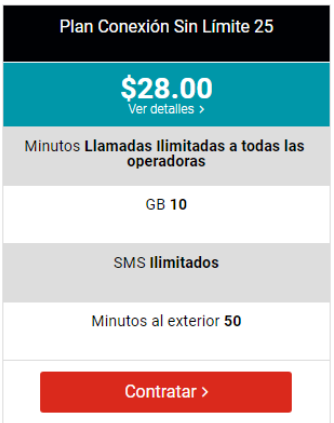


Figura 4.51. Plan sugerido [40]

4.7. ACTUALIZACIÓN FINAL DEL TABLERO KANBAN

En la Figura 4.52 se muestra que las tareas realizadas para el proyecto de titulación han sido culminadas.

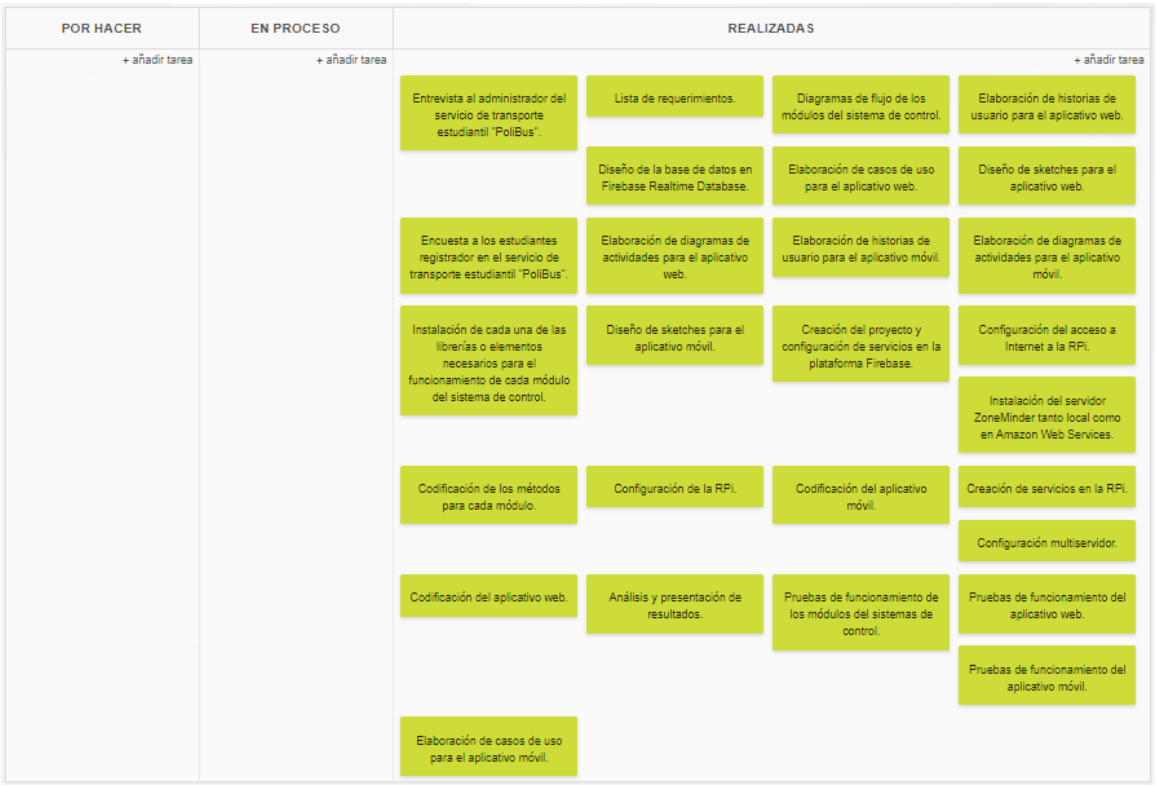


Figura 4.52. Tablero Kanban final

5. CONCLUSIONES Y RECOMENDACIONES

A continuación, se enlistan las conclusiones y recomendaciones obtenidas durante el desarrollo del prototipo.

5.1. CONCLUSIONES

- En el caso de querer migrar de base de datos a una SQL por ejemplo MySQL, esto es imposible, ya que Realtime Database utiliza otro tipo de métodos para conectar los aplicativos o para realizar consultas.
- Para habilitar el internet en la RPI en la unidad remota a través del USB modem 3G se tuvo dos opciones: UMTSkeeper junto con Sakis3g y wvdial. La primera opción demoraba en conectarse automáticamente llegando a tardar hasta 20 minutos, lo cual es perjudicial en el caso de que exista un corte de energía justo en el momento que se encuentre usando el prototipo, mientras la segunda opción es más sencilla de configurar y el tiempo de espera es mínimo (segundos).
- Para que el video Streaming funcione correctamente se necesita de una buena conexión a internet, por lo que la cobertura que ofrece la operadora es primordial para el funcionamiento de este servicio.
- El uso del lenguaje de programación Python para realizar las consultas a los nodos y ramificaciones en la base de datos Firebase es muy intuitivo puesto que solo se necesita tener la referencia para extraer una determinada información.
- Para que la monitorización mediante video funcione correctamente en el prototipo es indispensable el uso de Dataplicity, puesto que no basta solo con la configuración multiservidor de ZoneMinder, sino también que el servidor local (RPi con cámara) y el servidor centralizado estén en la misma red, en este caso el internet y eso se logra hacerlo con la publicación en la nube del servidor local mediante Dataplicity.
- El uso de la placa Raspberry Pi en este proyecto garantiza tener un control y configuración total de los módulos incorporados a los puertos USB y pines GPIO a un bajo costo. Así mismo brinda una escalabilidad en cuanto a servicios y módulos adicionales se necesite ofrecer a futuro.
- Debido a que en el lenguaje de programación Python no se puede implementar directamente un código que detecte el cambio en los nodos o ramificaciones en Firebase es mandatorio la codificación de scripts que consulten y comprueben estos

cambios de forma permanente y con ello posteriormente lograr accionar un determinado módulo en la RPi.

- El uso de las plataformas Firebase y Google Maps en el prototipo brinda una alta disponibilidad en todos los servicios que estas ofrecen. Así mismo, se tiene una transferencia de información segura debido a que siempre se usa un certificado para realizar la conexión y transferencia de información entre RPi y Firebase.
- El uso del asistente de Android Studio para agregar automáticamente las dependencias de la plataforma Firebase al proyecto es de gran utilidad puesto que si se modifica erróneamente los archivos del proyecto la aplicación tendrá errores en la compilación, así mismo, la versión de las dependencias debe ser la más actualizada para usar todas las funciones que los servicios de Firebase ofrecen en la actualidad.
- Con las pruebas realizadas se demuestra que el prototipo funciona correctamente y cumple con todos los Requerimientos Funcionales, así como con los Requerimientos No Funcionales.

5.2. RECOMENDACIONES

- Se debe leer las especificaciones técnicas de cada uno de elemento electrónicos que se van a integrar a la RPi para evitar que estos sufran daños.
- En cuanto a la base de datos proporcionadas por la plataforma Firebase también se puede usar Cloud Firestore en lugar de Realtime Database, lo que proporcionaría un mayor número de operadores que permitirán realizar consultas más específicas o complejas.
- Para evitar lecturas erróneas de la huella dactilar enrolada se debe verificar que el lente del sensor biométrico y dedo a enrolar estén limpios ya que si se encuentran mojados o con grasa dificulta su lectura.
- Para la opción de multiservidor, en la pestaña de configuración de servidores en el ZoneMinder centralizado, se recomienda verificar que los path cargados por defecto existan, pues dependiendo de la versión de ZoneMinder que se esté usando estos path son diferentes.
- Se recomienda incorporar al prototipo una fuente de energía de respaldo debido a que en el entorno de funcionamiento pueden existir apagones ocasionados por maniobras del chofer.

- Se recomienda la integración de una base local de las huellas dactilares para el caso de no existir una conectividad a internet y así continuar brindando un control de acceso a los estudiantes.
- Se recomienda verificar las actualizaciones de la Plataforma Firebase debido a que constantemente realizan mejoras de las funcionalidades de cada servicio.
- Si se desea tener una mejor calidad de video se recomienda mejorar el hardware lo que implica también un ajuste en las configuraciones de ZoneMinder, así como también analizar el ancho de banda necesario.
- Para la configuración de multiservidor de ZoneMinder se recomienda primero verificar la conexión a la base de datos del servidor centralizado.
- Acorde a lo dialogado con el Administrador del servicio de transporte estudiantil de la EPN se recomienda incentivar la realización e implementación de 3 funcionalidades adicionales al presente proyecto las cuales se detallan a continuación: Que el prototipo tenga de una base de datos local para el control de acceso de los estudiantes, la misma que se sincronice cada que tenga conexión a internet con la base de datos en la nube. Además de un respaldo automático de las grabaciones de video dentro de la unidad de transporte y por último un envío automático vía email de los reportes semanales generados por el prototipo.

6. REFERENCIAS BIBLIOGRÁFICAS

[1] F. M. Strauch, «ETSETB,» [En línea]. Available: https://upcommons.upc.edu/bitstream/handle/2099/9969/Article006.pdf?sequence=1&isAllowed=y. [Último acceso: 06 06 2019].

[2] D. L. Hora, «La Hora,» 19 06 2013. [En línea]. Available: https://lahora.com.ec/noticia/1101523240/buses-tendrc3a1n-un-registro-real-de-sus-pasajeros--. [Último acceso: 06 06 2019].

[3] «ECU911,» [En línea]. Available: http://www.ecu911.gob.ec/servicio-integrado-de-seguridad-ecu-911/. [Último acceso: 30 05 2019].

[4] A. M. Carvajal, «El Comercio,» 31 08 2018. [En línea]. Available: https://www.elcomercio.com/actualidad/transporte-escolar-exigencias-quito-control.html. [Último acceso: 1 08 2019].

[5] EducacionIT, «Arduino y Raspberry Pi: qué son y cuáles son sus similitudes y diferencias,» 27 08 2018. [En línea]. Available: https://blog.educacionit.com/2018/08/27/arduino-y-raspberry-pi-que-son-y-cuales-son-sus-similitudes-y-diferencias/. [Último acceso: 2019 08 08].

[6] I. PE, «Comparativa y análisis: Raspberry Pi vs competencia.,» [En línea]. Available: https://comohacer.eu/comparativa-y-analisis-raspberry-pi-vs-competencia/. [Último acceso: 18 03 2020].

[7] Raspbian Pi Foundation, «Raspbian,» [En línea]. Available: https://www.raspbian.org/. [Último acceso: 14 08 2019].

[8] Tech make electronics, «Tech make electronics,» [En línea]. Available: http://www.techmake.com/intro-gps-neo#targetText=GPS%20se%20refiere%20a%20las,o%20Sistema%20de%20posicionamiento%20global.. [Último acceso: 09 09 2019].

[9] Nivian, «Nivian,» [En línea]. Available: https://www.nivianhome.com/es/que-es-una-camara-ip/#targetText=Las%20c%C3%A1maras%20IP%20son%20videoc%C3%A1maras,incluso%20grabar%20las%20im%C3%A1genes%20remotamente.. [Último acceso: 09 09 2019].

[10] Raspberry, «Camera Module V2,» [En línea]. Available: https://www.raspberrypi.org/products/camera-module-v2/. [Último acceso: 07 09 2019].

[11] ZoneMinder, «ZoneMinder,» [En línea]. Available: https://zoneminder.readthedocs.io/en/1.32.3/installationguide/index.html. [Último acceso: 07 09 2019].

[12] ZoneMinder, «Componentes ZoneMinder,» [En línea]. Available: https://zoneminder.readthedocs.io/en/1.32.3/userguide/components.html#binaries. [Último acceso: 18 03 2020].

[13] Blue Iris, «Blue Iris,» [En línea]. Available: https://blueirissoftware.com/. [Último acceso: 09 09 2019].

[14] Tom Flanagan, «GitHub,» [En línea]. Available: https://github.com/Knio/pynmea2. [Último acceso: 12 09 2019].

[15] PM Code Works, «PM Code Works,» [En línea]. Available: https://www.pm-codeworks.de/pamfingerprint.html. [Último acceso: 17 09 2019].

[16] Acenswhitepapers, « Bases de datos NoSQL.,» [En línea]. Available: https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf . [Último acceso: 29 08 2019].

[17] PandoraFMS, «Base de datos NoSQL,» 19 04 2016. [En línea]. Available: https://pandorafms.com/blog/es/bases-de-datos-nosql/. [Último acceso: 29 08 2019].

[18] Amazon, «¿Que es NoSQL?,» [En línea]. Available: https://aws.amazon.com/es/nosql/.

[19] Firebase, «Firebase Realtime Database,» [En línea]. Available: https://firebase.google.com/docs/database/?hl=es-419. [Último acceso: 09 08 2019].

[20] Firebase, «Firebase Authentication,» [En línea]. Available: https://firebase.google.com/docs/auth/?hl=es-419. [Último acceso: 12 08 2019].

[21] Firebase, «Firebase Hosting,» [En línea]. Available: https://firebase.google.com/docs/hosting/?hl=es-419. [Último acceso: 12 08 2019].

[22] Firebase, «Planes de precios,» [En línea]. Available: https://firebase.google.com/pricing?hl=es-419. [Último acceso: 12 08 2019].

[23] Dataplicity, «Features,» [En línea]. Available: https://www.dataplicity.com/features/. [Último acceso: 01 09 2019].

[24] Microsoft, «Visual Studio Code,» [En línea]. Available: https://code.visualstudio.com/docs. [Último acceso: 09 08 2019].

[25] HIPERTEXTUAL, «¿Qué es HTML5 ?,» [En línea]. Available: https://hipertextual.com/archivo/2013/05/entendiendo-html5-guia-para-principiantes/. [Último acceso: 20 01 2020].

[26] Uniwebsidad, «Cómo incluir JavaScript en documentos XHTML,» [En línea]. Available: https://uniwebsidad.com/libros/javascript/capitulo-1/como-incluir-javascript-en-documentos-xhtml. [Último acceso: 14 08 2019].

[27] Á. Fontela, «¿Que es Bootstrap?,» 16 07 2015. [En línea]. Available: https://raiolanetworks.es/blog/que-es-bootstrap/. [Último acceso: 14 08 2019].

[28] G. Developers, «Android Developers,» [En línea]. Available: https://developer.android.com/studio/intro/?hl=ES. [Último acceso: 03 09 2019].

[29] Java, «Java,» [En línea]. Available: https://java.com/es/download/faq/whatis_java.xml. [Último acceso: 09 09 2019].

[30] MundoLinux, «Mundo Linux,» [En línea]. Available: https://www.mundolinux.info/que-es-xml.htm. [Último acceso: 09 09 2019].

[31] Billage, «Billage,» [En línea]. Available: https://www.getbillage.com/es/blog/metodologia-kanban-ventajas-y-

caracteristicas#targetText=Kanban%20es%20una%20palabra%20japonesa,de%20determinados%20procesos%20y%20tareas.. [Último acceso: 02 09 2019].

[32] Kanbantool, «Kanbantool,» [En línea]. Available: https://kanbantool.com/es/tablero-kanban. [Último acceso: 02 09 2019].

[33] Nube Bus, «Nube Bus,» [En línea]. Available: https://nubebus.com/. [Último acceso: 20 04 2020].

[34] School Transport Tracker, «Real-time School Transport Tracking,» [En línea]. Available: https://www.schoolbustrackerapp.com/. [Último acceso: 20 04 2020].

[35] Mercado Libre, «Navegadores GPS,» [En línea]. Available: https://articulo.mercadolibre.com.ec/MEC-422028346-mgsystem-modulo-gps-ublox-neo-6m-0-001-ideal-arduino-_JM?quantity=1#position=1&type=item&tracking_id=e22dbab6-20be-4c95-b32d-816a999d6329. [Último acceso: 09 10 2019].

[36] Electrónica, «Lector huella datilar,» 19 08 2019. [En línea]. Available: https://electronica.com.ve/new/catalog/product_info.php?products_id=3547.

[37] ZoneMinder, «Multi-Server Install,» [En línea]. Available: https://zoneminder.readthedocs.io/en/latest/installationguide/multiserver.html. [Último acceso: 07 09 2019].

[38] Lucidchart, «Lucidchart,» [En línea]. Available: https://www.lucidchart.com/pages/es/. [Último acceso: 09 12 2019].

[39] GeeksforGeeks, «Haversine Formule,» [En línea]. Available: https://www.geeksforgeeks.org/haversine-formula-to-find-distance-between-two-points-on-a-sphere/. [Último acceso: 27 09 2019].

[40] CLARO, «CLARO ECUADOR,» [En línea]. Available: https://tienda.claro.com.ec/plan/vermas/1/420/PLAN_CONEXION_SIN_LIMITE_25/NA/0/0. [Último acceso: 07 01 2020].

[41] Raspberrypi, «Raspberrypi,» [En línea]. Available: https://www.raspberrypi.org/documentation/hardware/camera/README.md. [Último acceso: 09 09 2019].

[42] F. M. Strauch. [En línea]. Available: https://upcommons.upc.edu/bitstream/handle/2099/9969/Article006.pdf?sequence=1&isAllowed=y. [Último acceso: 16 09 2019].

[43] Deloitte, «IoT-Internet de las cosas.,» [En línea]. Available: https://www2.deloitte.com/es/es/pages/technology/articles/IoT-internet-of-things.html. [Último acceso: 07 09 2019].

[44] Sistemas, «EEPROM,» [En línea]. Available: https://sistemas.com/eeprom.php. [Último acceso: 08 09 2019].

[45] ABCConsultorio, «¿Qué es una API y para qué sirve?,» [En línea]. Available: https://www.abc.es/tecnologia/consultorio/20150216/abci--201502132105.html. [Último acceso: 20 01 2020].

[46] Digicert, «¿QUÉ SON SSL, TLS Y HTTPS…?,» [En línea]. Available: https://www.digicert.com/es/what-is-ssl-tls-and-https-es/. [Último acceso: 20 01 2020].

[47] compara>hosting, «¿Qué es una CDN y cuando me conviene usar una?,» [En línea]. Available: https://www.comparahosting.com/p/que-es-una-cdn/. [Último acceso: 20 01 2020].

[48] Redes Zone, «Protocolo TLS Handshake,» [En línea]. Available: https://www.redeszone.net/tutoriales/internet/que-es-protocolo-tls-handshake/. [Último acceso: 20 01 2020].

[49] P. Review, «Telnet qué es y para que sirve,» [En línea]. Available: https://www.profesionalreview.com/2019/01/20/telnet-que-es/. [Último acceso: 11 12 2020].

[50] H. Tutoriales, «¿Como funciona SSH?,» [En línea]. Available: https://www.hostinger.es/tutoriales/que-es-ssh. [Último acceso: 11 12 2020].

7. ANEXOS

ANEXO A. Entrevista al administrador

ANEXO B. Encuestas a los estudiantes

ANEXO C. Mapas de cobertura celular

ANEXO D. Script main.py

ANEXO E. Codificación aplicativo web

ANEXO F. Codificación aplicativo móvil

ANEXO G. Encuestas de satisfacción

ORDEN DE EMPASTADO