Soporte y Desarrollo de software bajo requerimientos para la Empresa CINNOV S.A.S

Hanuar Antonio Rubio Cárdenas

Universidad de Pamplona
Facultad de Ingenierías y Arquitectura
2022

Soporte y Desarrollo de software bajo requerimientos para la Empresa CINNOV S.A.S

Hanuar Antonio Rubio Cárdenas

Trabajo de grado para optar
Titulo de Ingeniero Mecatrónico

Director(a):
Prof. Cristhian Ivan Riaño Jaimes, PhD. MSc. Esp. Ing.

Universidad de Pamplona
Facultad de Ingenierías y Arquitectura
2022

Dedico este trabajo a...

A mi padre José Hanuar Rubio León y mi madre María Isolina Cárdenas Rangel, quienes siempre me han apoyado con su amor, paciencia y esfuerzo, me ayudaron a hacer realidad un sueño mas.

Agradecimientos

A mis profesores quienes me guiaron en este proceso y compartieron de sus conocimientos, en especial a mi Director de Tesis, que gracias a sus consejos y correcciones hoy puedo culminar este trabajo. Gracias por su ayuda, paciencia y dedicación.

A mis amigos y colegas, no puedo evitar recordar cuántos días y horas hemos pasado trabajando juntos durante este tiempo. Hoy nos toca cerrar un capítulo en esta historia de vida y no puedo dejar de agradecerles su apoyo y fortaleza durante nuestras horas más difíciles, gracias por estar siempre ahí.
También quiero agradecer a toda mi familia que me ha apoyado en este proceso.

Índice general

Índice de figuras

Índice de cuadros

Siglas

API Interfaz de programación de aplicaciones. 11, 46, 62, 63

APP Aplicacion. 27, 28, 39, 61

AWS Amazon Web Services. 15, 30, 44, 48

DB Policaprolactona. 31

DBMS Sistema de gestion de bases de datos. 30

GIS Sistema de información geográfica. 28

GPS Sistema de posicionamiento Global. 28, 29, 63

GSM Global System for Mobile communications (sistema global para las comunicaciones móviles). 28

IDE Entorno de desarrollo integrado. 60

JSON JavaScript Object Notation. 43, 44, 57

POO Programación orientada a objetos. 23, 25

SDK Kit de desarrollo de software. 15, 38, 60, 63

V2B Vehicle to Bicycle Communication. 22

1 Introducción

1.1. Resumen

En este trabajo de grado se pretende dar a conocer el proceso a realizar con la empresa CINNOV S.A.S durante la práctica profesional, el cual tiene enfoque principal dar apoyo al desarrollo de software para productos como ingeniero junior. En dicho proceso se realizarán actividades para mejoramiento y actualización a una nueva versión de una aplicación móvil desarrollada con Kit de desarrollo de software (SDK) de Flutter,tea se conectará con los productos de seguridad vial que contiene la empresa. Dichas mejoras comprenden conexión de aplicación a base datos con tecnología de Cloud Computing Amazon Web Services (AWS), crear conexión con el dispositivo vía bluetooth, realizar intercambio de datos, generar rutas en mapa real para el usuario y esto a su vez generando las indicaciones al dispositivo.

1.2. Abstract

This degree work is intended to present the process to be carried out with the company CINNOV S.A.S during the professional practice, which mainly focuses on developing software for products as a junior engineer. In this process, activities will be carried out to improve and update to a new version of a mobile application developed with the Flutter SDK; tea will connect with the road safety products the company contains. Improvements presented in this document include connecting the application to the database with AWS Cloud Computing technology, updating and building the backend with the database from both NodeJs and MongoDB, connecting to the device via Bluetooth for data exchange, generation of both routes and accurate maps for the user, and management of directions on the device.

1.3. Planteamiento del problema

En los últimos años se ha incrementado el uso de la bicicleta como medio de transporte para distancias cortas. Esto se debe a que la mayoría de los usuarios lo utilizan no solo como una forma de hacer ejercicio con el, sino también como un medio de transporte barato y saludable para mover el por la ciudad. De manera similar, ahora queremos que la mayoría de las ciudades de sean ciclo inclusivas, lo que significa andar en bicicleta de manera segura y cómoda para todos los viajes de. Esto ayudará a reducir las emisiones de gases de efecto invernadero en el transporte, mejorar la calidad del aire y el ruido y mejorar la socialización de en los espacios públicos. Esto mejorará y proporcionará accesibilidad al transporte público.[1]

Sin embargo, el aumento en el uso de bicicletas como medio de transporte contribuyó a accidentes de bicicleta adicionales. Los ciclistas son los menos protegidos en la carretera. Por lo tanto, necesita mejoras para garantizar la seguridad de los ciclistas, esta es una de las principales razones por las que no utiliza bicicletas como medio de transporte porque tiene miedo de incorporarse al tráfico. El empresario Alexander Nieves, fundador de CiNNOV S.A.S., creó la plataforma BiGo, Como guía para que los automovilistas puedan ver ciclistas, motociclistas y patinadores en la carretera. Esto se debe a la gran cantidad de accidentes que ocurren todos los días. El emprendimiento tiene dos dispositivos, uno en el casco y otro en la espalda usuario para que el conductor pueda verlo fácilmente. Según Alexander, en el casco hay un tipo de sensor que envía señales a través de Bluetooth cuando se identifica movimiento de la cabeza, con el objetivo de indicar al usuario adónde ir. Este dispositivo tiene una aplicación móvil en desarrollo que permite esto conectarse al dispositivo, realice el intercambio de datos, cree rutas de mapas reales para el usuario y este a su vez crea indicadores para el dispositivo, todo en aras de mejorar la seguridad vial de los ciclistas. [25]

1.4. Justificación

La opción de Prácticas Empresariales tiene como objetivo completar la formación académica que permita a los estudiantes trabajar en una empresa, preparándolos para su práctica para ampliar los conocimientos adquiridos y fortalecer sus habilidades para que puedan crecer como profesionales. Por otro lado, brinda un espacio para poner a prueba las habilidades adquiridas durante la formación y desarrollar las relaciones necesarias para el trabajo formal en el campo profesional. Gracias a la oportunidad que brindo empresa CINNOV S.A.S de realizar mis prácticas, me han permitido integrarme a un grupo de trabajo y así poder desarrollar las habilidades para trabajar en equipos multidisciplinarios, lo cual es necesario para cumplir con los requerimientos actuales del entorno. Me da mucho gusto estar en esta etapa de preparación, porque permanecer en la empresa traerá grandes beneficios y hará un aporte indispensable a la vida.

1.5. Objetivos

1.5.1. Objetivo General

• Desarrollar actividades conducentes a la actualización y construcción de software para dispositivos de seguridad vial para la empresa CINNOV S.A.S.

1.5.2. Objetivos Específicos

- Crear lista de necesidades de proyecto y planificar actividades para contribuir en el proyecto de actualización.
- Identificar los requisitos de los dispositivos de la empresa y cambios que requieran.
- Implementar mejoras partiendo de los requerimientos de diseño.
- Desarrollar códigos en mejoramiento y construcción de aplicación móvil que se conectará con los dispositivos de la empresa.
- Validar el funcionamiento de dispositivo y el desarrollo del software.

1.6. Empresa

1.6.1. Historia

La empresa Cinnov Sas tiene como domicilio principal de su actividad la dirección, CARRERA 90 C 127 B 30 en la ciudad de BOGOTA, BOGOTA. Esta empresa fué constituida como SOCIEDAD POR ACCIONES SIMPLIFICADA y se dedica a Fabricación de aparatos electrónicos de consumo.
Dicha empresa nace nace en noviembre 2018 con la creacion de Bigo después de que Alexander Nieves creador del dispositivo, casi pierde su vida cuando fue envestido por un bus que no lo vio mientras se movilizaba en su bicicleta. Crea un primer prototipo muy artesanal que soluciona el problema de falta de visibilidad con algunos materiales que tenía a su alcance y desarrolla un sistema novedoso y sencillo de control a partir de los movimientos de la cabeza para activar las señales luminosas.[1]

1.6.2. Mision

Mejorar la calidad de vida de la humanidad por medio de servicios en software y hardware, creando experiencias de seguridad y compañía para los usuarios con productos de calidad.[1]

1.6.3. Visión

Ser empresa líder para el 2024 en innovación tecnológica, para satisfacer las necesidades de la humanidad.[1]

1.6.4. Bigo

Bigo es un desarrollo tecnológico que integra hardware y software a través de una plataforma digital que asegura todos los elementos de la nueva movilidad y movilidad sostenible con importantes riesgos para los que están en mayor riesgo en la vía, es un tema de salud pública. Nuestra plataforma es capaz de integrar con dispositivos de asistencia y garantiza la seguridad de cualquier usuario en el mundo, ya sean empleados, colaboradores, ciudadanos o familiares, los dispositivos tecnológicos que han desarrollado así como también con los equipos del mercado..[1]
Las Practicas se basaran principalmente en el desarrollo complementario como mejora para este dispositivo que se vinculara con una aplicación que esta en desarrollo.

Dicha aplicación tiene el nombre de Bisafe BiSafe es el software que se desarrollara,y este tendrá como propósito permitir ofrecer servicios en salud y seguridad, por ejemplo en la creación y recorrido de rutas seguras, él envió de señales SOS cuando se presentan accidentes, como también el monitoreo del estado de salud del usuario, así mismo una comunidad conectada que se apoya y se integra a los servicios de emergencia y seguridad, la captura, medición y análisis en tiempo real de información de mucho valor en las ciudades que le apuestan a tener una movilidad inteligente y segura. [16]

2 Estructura del documento

En el **Capitulo 1 Introducción** se da una entrada al documento dando a conocer la idea en un resumen de lo que sera desarrollado en las practicas empresariales.

En el **Capitulo 2 Reseña** se da a conocer algunos datos generales de la empresa y del dispositivo principal en el que se estará desarrollando las practicas.

En el **Capitulo 3 Marco teórico y Estado del arte** se presenta de forma general los antecedentes y las consideraciones teóricas de temas que se deben tener en cuenta para el desarrollo de las practicas. Estos antecedentes son una revisión resumida de los estudios existentes que de manera directa o indirecta abordan una similitud.

En el **Capitulo 4 Tecnologías utilizadas** se aborda de una manera mas especifica las tecnologías utilizadas y conocimientos adquiridos para el desarrollo de las practicas.

En el **Capitulo 5 Fase de planificación** se tomaran en cuenta los requisitos y casos de usos a tener en cuenta sobre desarrollo de la aplicación.
En el **Capitulo 6 Implementación** Se da a conocer los pasos que se siguieron para la creación del proyecto.
En el **Capitulo 7 Resultados** Se evidencia cada uno de los requisitos sean cumplidos y se evidencia la aplicación en una ruta dando indicación a dispositivo indicador.

3 Marco Teórico y Estado del Arte

En esta sección se abordará la temática a nivel teórico sobre todos los conceptos e información que se deben conocer y tener en cuenta para la realización de práctica profesional

3.1. Estado del arte

Diseño y presentación de chaqueta inteligente destinada para los ciclistas

Vodafone Smart Jacket, un prototipo de la chaqueta inteligente desarrollada por La Universidad de Delft y la Asociación de Ciclistas Holandeses combinan un ordenador conectado a los teléfonos inteligentes haciendo que la conducción urbana sea más segura. La aplicación móvil guía el camino por la ciudad y con su sistema de luces LED, la carcasa indica los movimientos que realizará el ciclista y también le permite agradecer a otros vehículos por facilitar las maniobras. Esta aplicación, Además, evita recibir llamadas mientras se conduce lo que limita la distracción, que es una de las principales causas de accidentes. [6]

3.1.1. Diseño de aplicación tecnológica para el reporte de incidentes con vehículos no motorizados; Bicifor.

Bicifor, llamado así con el propósito de "Formato para biciusuarios.[es] una propuesta para encontrar una solución a un problema al que se enfrentan los ciclistas todos los días que operan en la ciudad de Bogotá ,mediante el uso de tecnología y la georreferenzacion, Bicifor es una plataforma donde cada usuario brinda información sobre su bicicleta, pero al mismo tiempo cualquier evento que ocurra en el uso diario, accidentes, robos, condiciones climáticas o infraestructuras viales, para que pueda estar informado de lo que sucede a su alrededor y crear soluciones a través de las distintas entidades responsables. Como resultado, la ciudad mantiene un control organizado como lo hace sobre los sistemas motorizados, lo que también reduce el uso de papel y mejora el tiempo, porque es una aplicación activa en tiempo real y no deja información "perdida", de lo contrario, todos los

datos se utilizan para procesar y optimizar cualquier condición que afecte a los ciclistas. [19]

3.1.2. Vehicle to Bicycle Communication

"Vehicle to Bicycle Communication (V2B) es una tecnología emergente que se espera mejorar los estándares de seguridad de las bicicletas y reducir el número de muertes causadas por motocicletas accidentes cada año, es una tecnología que permite a los vehículos comunicar su posición, velocidad y dirección de aceleración a otros vehículos en su ciudad."[24]

3.1.3. Light on the road: chaqueta inteligente para evitar accidentes en moto.

El dispositivo es una chaqueta que incluye dispositivos electrónicos que permiten conectar el vehículo para proyectar los frenos y las señales de giro con una serie de luces ubicadas estratégicamente en una prenda aumentando el área de iluminación por el conductor, está previsto diseñar un conjunto completo de ropa y equipo de seguridad con los mismos principios y a largo plazo, se harán propuestas con las entidades relevantes para regular el uso de dispositivos que aumenten la visibilidad de los ciclistas y conductor de motocicleta. [18]

3.2. Marco Teórico

3.2.1. Programación estructurada y programación orientada a objetos

El paradigma de la programación estructurada (o procedimental) se basa en el hecho de que la creación de un programa que se ocupa de identificar procesos que se ejecutan en un conjunto independiente datos. La división entre datos y funciones o procesos supone demasiada carga de trabajo en el desarrollo de software. La estructura de datos (representa la relación lógica entre elementos de datos individuales) requieren funciones y modificaciones frecuentes. En este caso, los controles deben volver a probarse para ver si todavía siguen estando coordinadas con los datos. En el modelado de objetos, el objetivo del diseño cambia del comportamiento del modelado el científico para modelar las cosas en el mundo y sus comportamientos individuales . El diseño orientado a objetos es una nueva forma de pensar en problemas con patrones sobre conceptos del mundo real. Lo

principal es la construcción del objeto, la combinación, la estructura y el comportamiento de los datos como una sola entidad. [14]

Lenguajes de programación orientados a objetos

Los lenguajes de programación orientada a objetos más populares son:

- Java
- JavaScript
- Python
- C++
- Visual Basic .NET
- Ruby
- Scala
- PHP

Clase

Las clases son el concepto básico de la Programación orientada a objetos (POO). Una clase recopila objetos idénticos (que tienen las mismas propiedades y proporcionan las mismas operaciones) excepto por su estado específico en un momento dado.

- Los atributos representan los datos internos del objeto.
- Las operaciones (llamadas métodos en POO) representan su comportamiento: los mensajes que pueden recibir y procesar de otros objetos.

Por ejemplo,En la Figura 3.1 representaremos 3 objetos de usuario, caracterizados por su nombre, años y ocupación. También queremos que los objetos puedan responder a los mensajes preguntando su nombre, edad u ocupación.
José, 23 años, carpintero.
Laura, 37 años, estudiantes.
Cristina, 19 años, estudiante. [20]

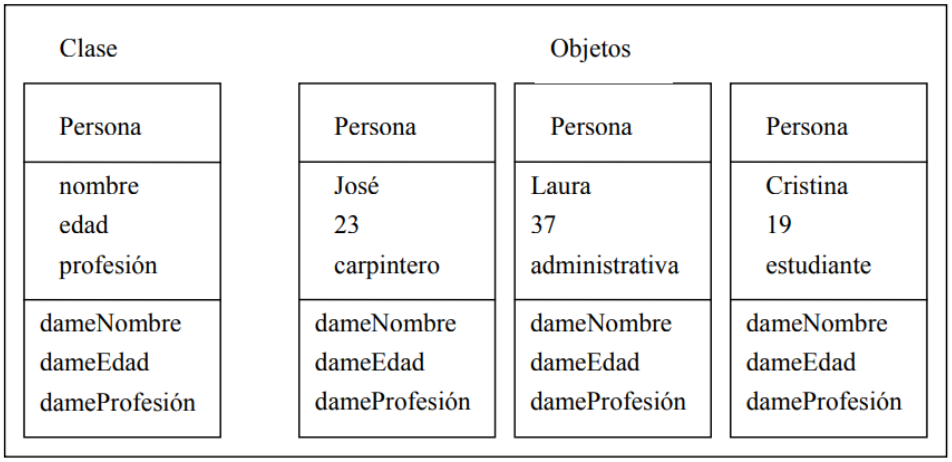


Figura 3.1: Representación de clases y objetos
Fuente:Tomado de [20]

Objeto

Los objetos son entidades claramente distinguibles e identificables, por ejemplo El objeto puede ser una persona, un libro o un botón de comando.
Estructura interna de un objeto: La estructura interna de un objeto está compuesta por tres elementos fundamentales como son:

- Propiedades:Estas son las propiedades observables del objeto. Se reconocen porque describen un aspecto de un objeto con el que podemos medir escala predefinida. A cada atributo se le debe asignar un valor Le permite identificar de forma única el objeto.
- Métodos:Se define como un conjunto de acciones que puede realizar un objeto. para lograr un objetivo. Los métodos son la parte vital e interesante de un objeto se usa comúnmente para modificar las propiedades del objeto. La modificación de la propiedad de un objeto cambia su apariencia y se genera un cambio visible para el usuario de la aplicación.
- Eventos: Todos los objetos están conectados con el mundo que los rodea, lo que significa que ninguno está aislado y no siempre se ve afectado por otros. Los eventos son estímulos que un objeto ejerce sobre otro. [20]

En la Figura 3.2 se muestra una representación de un objeto.

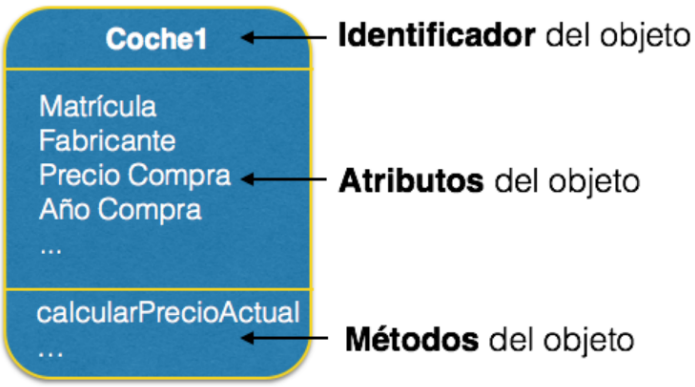


Figura 3.2: Representación objeto
Fuente:Tomado de [20]

Pilares de la POO

Para una gestión eficaz de clases y objetos creados con Programación orientada a objetos, algunos principios básicos nos ayudarán a reducir la complejidad, ser más eficiente y evitar problemas. Todos los lenguajes orientado a objetos se implementa de una forma u otra y es fundamental conocerlos bien.

- **Encapsulación:** El objeto está separado de su entorno por una envoltura. Esta separación determina la unidad del objeto, es decir, crea algo independiente de él. La encapsulación oculta detalles sobre la estructura interna de un objeto, Esto significa que solo sabemos sobre el objeto lo que se revela a través de sus métodos. Su nombre encapsulación son las propiedades del objeto que tiene para ocultar los detalles internos, por lo que garantiza que el contenido de la información de un objeto esté oculto al mundo (por ejemplo, el objeto A no sabe lo que hace el objeto B y viceversa). Por otro lado, un programador que está construyendo un objeto tiene acceso a todas sus partes: propiedades, métodos y definiciones de eventos. El programador tiene acceso a sus propiedades, métodos (si son públicos) y a la programación de eventos ya definidos para el objeto. Varios métodos y propiedades de un objeto son privados, lo que significa que solo son útiles para el público y no son conocidos por el usuario. [20][14]
- **Herencia de clases:** La herencia asume una clase base y una jerarquía de clases que contiene clases derivadas de la clase base, de modo que las clases derivadas pueden heredar propiedades y los métodos de la clase base, agregan o incluso modifican sus propios métodos y propiedades, los elementos de la clase base deben ser diferentes. La herencia es un mecanismo para determinar la similitud entre clases de una clase base o más clases derivadas.

En la Figura 3.3 se muestra una representación de Herencia de clases. La clase

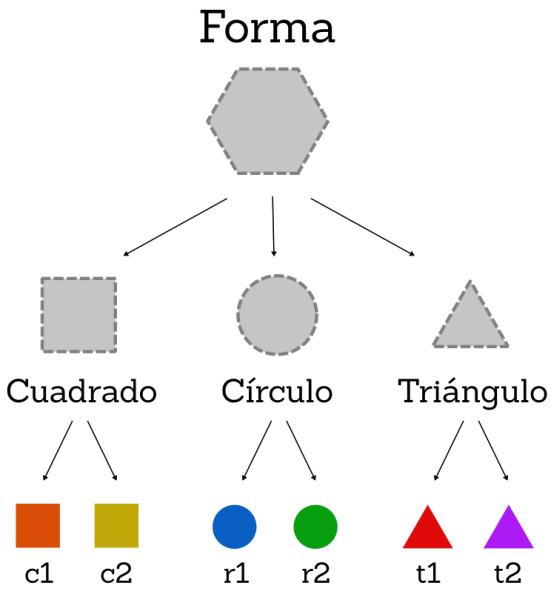


Figura 3.3: Herencia de clases
Fuente:Tomado de [20][14]

base contiene todas las propiedades y comportamientos comunes a las clases derivadas . Las clases derivadas representan diferentes formas de lograr este núcleo. La clase base contiene todas las propiedades (propiedades) y comportamientos (métodos) compartidos por clases derivadas (el núcleo de clases "similares"). Las clases derivadas representan diferentes formas de lograr este núcleo.

- **Polimorfismo:** El polimorfismo ocurre cuando creas objetos que pertenecen a clases con derivados de otras clases. En otras palabras, el polimorfismo ocurre en clases derivadas. También decimos que el polimorfismo es una posible propiedad de una entidad que tiene alguna forma, de hecho el polimorfismo permite referencias a objetos de diferentes clases usando el mismo elemento del programa y realizar la misma operación de diferentes maneras, dependiendo del elemento especificado en el tiempo. Por ejemplo, al describir una clase de mamíferos, se puede observar que comer es una actividad esencial en la vida de los mamíferos, por lo que cada tipo de animal puede realizar una actividad o función de forrajeo. Por otro lado, Una vaca o cabra pastando en el campo, un niño comiendo dulces o golosinas y Un león comiendo otro animal, existen diferentes métodos que utilizan animales con diferentes pechos para realizar la misma función. (comer).[20][14]

3.2.2. Aplicación Móvil

Una Aplicacion (APP) es una herramienta diseñada para el desarrollo de funciones específicas para una plataforma específica: teléfono móvil, tableta, TV, computadora, etc. Puede descargar o acceder a aplicaciones desde teléfonos u otros dispositivos móviles, como tableta o reproductor MP3. Para acceder a él desde el móvil, el teléfono inteligente debe tener conexión y acceso a Internet. [29]

Aplicaciones Nativas

Una aplicación nativa se desarrolla específicamente para sistemas específicos y plataformas de desarrollo de fabricantes . Estas aplicaciones se adaptan al 100 % a las funciones y características de los dispositivos, generando así una mejor experiencia de usuario. Se puede decir que desarrollar una aplicación nativa tiene un mayor costo , ya que al desarrollar una aplicación multiplataforma, se deben crear una versión para cada uno de los sistemas operativos, lo que produce un aumento en el costo y en el tiempo de desarrollo. Algunos ejemplos de aplicaciones nativas son WhatsApp o Facebook. [28]

En la Figura 3.4 se muestra una representación de los sistemas operativos y sus lenguajes de programación nativos..

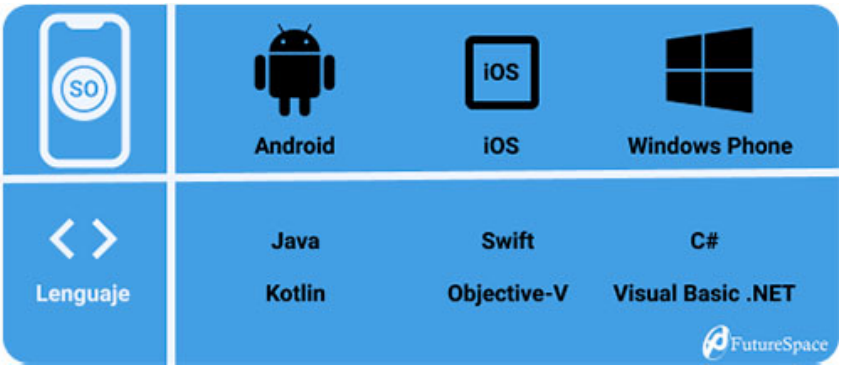


Figura 3.4: Sistemas operativos y sus lenguajes nativos.
Fuente:Tomado de [28]

Aplicaciones Web

Una aplicación web es la opción más fácil para desarrollar aplicaciones porque el desarrollo de una sola aplicación mantiene los costos más bajos posible. Con esta aplicación es posible utilizar Responsive Web Design, desarrollando así una aplicación apta para todos los dispositivos. Por el contrario, la aplicación web proporciona una experiencia

de usuario más débil porque ignora las características del dispositivo y es menos segura porque se basa en la seguridad proporcionada por el propio navegador.[28]

En la Figura 3.5 se muestra una representación de funcionamiento Aplicación web.

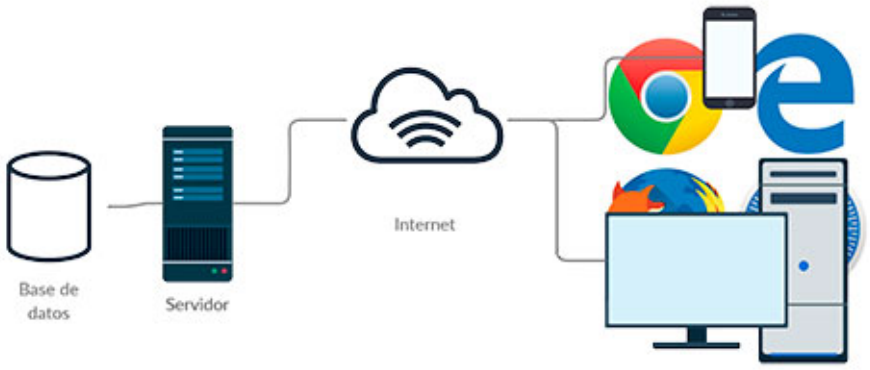


Figura 3.5: Aplicación web

Aplicaciones Híbridas

La APP hibrida es una mezcla de las dos aplicaciones anteriores Aprovechan la flexibilidad de evolucionar aplicaciones web con HTML5 CSS y JavaScript lo que permite aplicar algunas funciones de hardware que están disponibles en aplicaciones nativas que no se pueden gastar en aplicaciones web. Este servicio solo se desarrolla en lugar de mostrarse en un navegador web ya que si es necesario el caso de trabajar en la web se presenta en un WebView y así poder hacer uso de estos recursos de hardware a través los complementos. La principal ventaja de las aplicaciones híbridas además de la posibilidad de aplicar directamente recursos de hardware es que se trata de una arquitectura independiente del sistema operativo (iOS y Android).No requiere mantener un número de versión diferente dependiendo de la organización del sistema operativo. [28]

3.2.3. Geocalizacion

La geolocalización es la capacidad de conocer un lugar o una ubicación geográfica. A partir de un objeto se representa mediante un vector o un punto. Este proceso es comúnmente utilizado por Sistema de información geográfica(GIS), una colección organizada de hardware y software. Hay varios tipos de geolocalización como Global System for Mobile communications (sistema global para las comunicaciones móviles) (GSM), WIFI y Sistema de posicionamiento Global(GPS) que es el más importante.

La geolocalización es fundamental para el desarrollo de las practicas empresariales ya que necesitamos en una APP obtenga la geolocalización del dispositivo móvil para trazar el camino, el usuario crea su viaje, luego se guarda en la base de datos que se se puede ver en el mapa. Además, los usuarios pueden ver regiones de robo y tráfico más cercano. [15]

Sistema de posicionamiento Global (GPS)

El GPS utiliza el principio de triangulación y los componentes son satélites, estaciones de investigación espacial, control terrestre y monitoreo, y finalmente el receptor GPS es propiedad del usuario. Los satélites envían señales ,y estas son recibidas e identificadas por un receptor terrestre que proporciona coordenadas precisas de latitud y longitud.

3.2.4. Servidores de Almacenamiento Cloud

El concepto de nube hace referencia a una red global de servidores, cada uno con una función específica y sin ubicación física, sino una vasta red de servidores remotos interconectados. Actúa como un ecosistema, creado para almacenar, administrar datos, ejecutar aplicaciones y brindar contenido o servicios, como video streaming, webmail, scripts, etc., salas o redes sociales. La nube permite acceder a la información desde cualquier dispositivo con conexión a internet, lo que significa que estará en línea y disponible desde cualquier parte del mundo. Un servicio en la nube es una infraestructura, o un sistema de software alojado por servidores externos y disponible para que los usuarios accedan a través de Internet. En este tipo de servicio sus ventajas pueden ser numerosas ya que su uso no se limita a las computadoras, y sus capacidades de seguridad, como de almacenamiento y de recursos son superiores a las de las computadoras.
Este sistema de información es fundamental para el desarrollo actual y para el almacenamiento de datos que son generados en tiempo real por los usuarios como velocidad, coordenadas, tiempo, etc. [7]

Tipos de Servicios de la nube

Los proveedores de servicios en línea se pueden clasificar en una de tres categorías, dependiendo de si el servicio ofrecido es más “bajo.º “alto”. Estos son:

- Infraestructure–as–a–Service (IaaS)
- Platform–as–a–Service (PaaS)
- Software–as–a– Service (SaaS)

Infraestructure–as–a–Service

Estos servicios proporcionan la infraestructura sobre la que podemos construir nuestra arquitectura de aplicaciones. El proveedor proporciona al cliente una versión del sistema operativo. Por lo tanto, el desarrollador tiene todos los derechos sobre esta versión y es responsable de configurar y ejecutar muchos de los servicios que faltan.
Un proveedor tiende a clasificar muchas opciones en función del rendimiento de las instancias virtualizadas. En otras palabras, el servidor costará más, brindándole más opciones en términos de capacidad de CPU, RAM y disco duro. Uno de los más conocidos es AWS, que tiene instancias en centros de datos de todo el mundo.

Platform–as–a–Service

En lugar de proporcionar un sistema operativo completamente funcional sobre el cual el desarrollador tiene control total, esta cartera de servicios proporciona un servidor preinstalado capaz de ejecutar aplicaciones web basadas en una plataforma pública determinada tecnología.
Classic Apache + PHP + MySQL alberga la mayoría de los sitios web personales y profesionales construidos de este tipo. Sin embargo, el número de plataformas en uso ha aumentado, al igual que los proveedores. Hay proveedores en esta categoría que ofrecen planes gratuitos para proyectos más pequeños o que recién comienzan. [7]

3.2.5. Base de datos

La mayoría de los sistemas informáticos utilizan bases de datos para administrar la información, por lo que es fundamental que los desarrolladores estén capacitados para diseñarlos, crearlos y usarlos. Una base de datos es un lugar para almacenar datos relacionados con diferentes tipos. Organo. La base de datos representa ciertos aspectos del mundo real que son de interés para el usuario. Y almacena datos para un propósito específico. Son de "Datos"se refiere a hechos conocidos que se pueden registrar, como números de teléfono, direcciones, nombres, etc. Por lo tanto, un programa de computadora puede seleccione rápidamente y podrá: extraer, actualizar, insertar y eliminar en (Sistema de gestion de bases de datos (DBMS).[23]

Datos

Técnicamente, los datos son hechos y sin procesar, se pueden registrar, como números de teléfono , direcciones, nombres, pedidos y pagos, procesados para obtener información,

por ejemplo: , saldo adeudado y cantidad disponible. Nota: Los datos no contienen información. Los datos en Base de datos (DB) se refieren a archivos codificados digitalmente, bases de datos, documentos de texto, imágenes y video y voz.

Información

La información es una colección de eventos importantes y relevantes para el organismo u organización de los que tenga conocimiento. Definir información como: La información es un conjunto de datos, una descripción significativa y apropiada de eventos o entidades. Al contrario de los datos, la información es significativa para el destinatario siendo el resultado de un procesamiento de datos extraídos de la DB. Por lo general, se dan los términos y la información y estos pueden ser confundidos con sinónimos. [23]

Tipos de bases de datos

Existen diferentes formas de clasificar las bases de datos teniendo en cuenta sus características específicas:
según su transformación. De acuerdo con los procedimientos de retención y recuperación de datos, Podemos hablar de:

- Base de datos estática. Típicas de la inteligencia empresarial y otros campos del análisis histórico, se trata de bases de datos de sólo lectura de las que se puede extraer información, pero no modificar las listas existentes.
- Base de datos dinámica. Además de las operaciones básicas de consulta, estas bases de datos manejan la actualización, reordenación, adición y eliminación de información.[8]

Según su contenido. De acuerdo a la naturaleza de la información contenida, pueden ser:

- Bibliográficas. Contiene diversos materiales de lectura (libros, revistas, etc.). Ordenado a partir de información básica como datos de autor, Editorial, año de publicación, campo o título, en Muchas otras posibilidades
- De texto completo.Tratar con textos o documentos históricos, incluyendo La conservación debe realizarse a todos los niveles teniendo debidamente en cuenta las fuentes primarias.
- Directorios. :lista de datos personales o direcciones, correo electrónico, número de teléfono, etc. Empresas que gestionan un gran grupo de clientes.

- Especializadas. Bases de datos de información técnica o especializada, Adaptado a las necesidades específicas de una audiencia específica que consume esa información.[8]

Algunos modelos con frecuencia utilizados en las bases de datos:

- Bases de datos jerárquicas.
- Bases de datos de red.
- Bases de datos relacionales.
- Bases de datos orientadas a objetos.

- **Las bases de datos jerárquicas** primer modelo de base de datos lógica. Es un modelo rígido basado en un árbol con conexiones. padre/hijo, base de datos jerárquica diseñada para modelar relaciones jerarquía en el mundo real. Gracias a este tipo de base de datos, obtienes los resultados de los modelos, incluida una relación 1: n. En bases de datos descentralizadas, el ministerio confirma las actuaciones gráficas. El árbol tiene un botón que representa las unidades de información y las partes relacionadas de la relación 1: N. El modelo jerárquico utiliza dos conceptos estructurales: Registros y relaciones entre padres e hijos. Un registro es una colección de valores de campo que proporcionan información sobre una entidad. Un tipo de comunicación padre-hijo es la comunicación 1:N entre dos tipos de registro. El modo de grabación lateral 1 se denomina modo de grabación. Uno de los pares principales y la otra muestra en el lado N se denomina subtipo de registro.[8][9]

  En la Figura 3.6 se muestra un Esquema de base de datos jerárquicos.

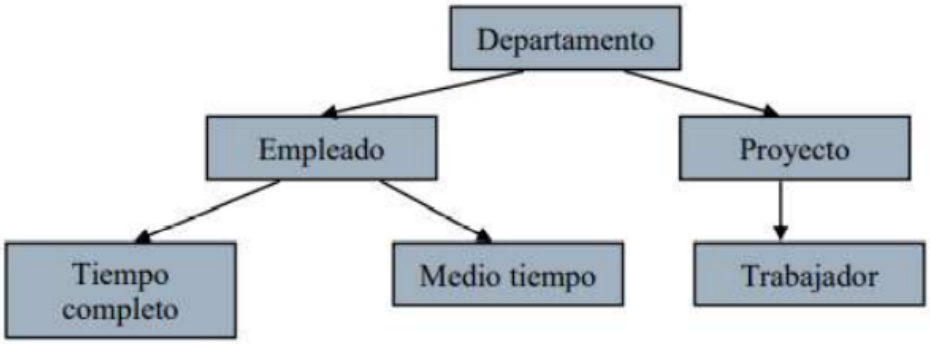


Figura 3.6: Esquema de base de datos jerárquicos
Fuente:Tomado de :[8][9]

- **Las bases de datos de red** Se basan en registros y conjuntos. Cada registro incluye un grupo de valores de datos relacionados. Hay tipos diferentes de registros, cada uno con un nombre. Las relaciones entre los datos se representan mediante enlaces,

que pueden verse como punteros. Los registros están organizados en conjuntos de gráficos arbitrarios. El tipo de conjunto es una asociación 1:N entre los dos tipos de registros. Cada definición de tipo de conjunto consta de 3 elementos: El nombre para el tipo de conjunto. Un tipo de registro de propietario. Algunos tipos de registros miembros. Cada instancia del conjunto asocia un registro de propietario, con un conjunto de registros de miembros.[8][9]

En la Figura 3.7 se muestra un Esquema de base de datos de tipo red.

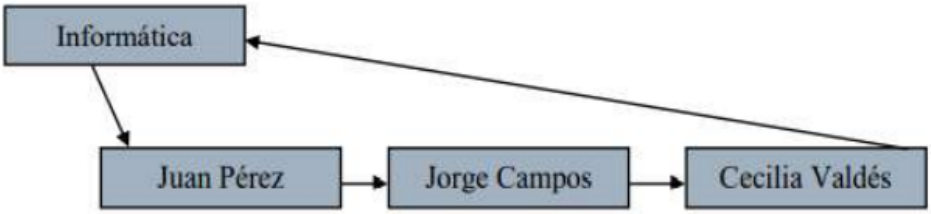


Figura 3.7: Esquema de base de datos de tipo red
Fuente:Tomado de :[8][9]

- **Las bases de datos relacionales** usan un conjunto de tablas para representar datos y La relación entre ellos. Cada tabla tiene muchas columnas y cada columna tiene un Nombre. El modelo relacional es un ejemplo de un modelo de registro. Se denominan así porque la base de datos está organizada en registros de formato fijo de varios tipos. Cada tabla contiene registros de un determinado tipo. Cada tipo de registro define un número fijo de campos o propiedades. Las columnas de la tabla corresponden a las características del tipo de registro. El modelo relacional oculta los detalles de las implementaciones de bajo nivel a los desarrolladores y usuarios de la base de datos. En una base de datos relacional, los datos se almacenan en diferentes tablas por tema o tarea. pero están entrelazados y pueden combinarse de cualquier forma dependiendo de para que toda esta información pueda ser descargada y agregada en cualquier momento. Conceptualmente, los sistemas relacionales trabajan con relaciones o matrices de datos, no datos individuales contenidos en el fichero.

  En la Figura 3.8 se muestra un Esquema de base de datos relacionales.

  Las relaciones o tablas le permiten presentar información de una manera más concisa. En una base de datos relacional, la información de un conjunto de datos con respecto a la información relevante de otro conjunto de datos, se trata de optimización cómo los usuarios ingresan, buscan y reportan datos. Los cuatro elementos principales de una base de datos se describen a continuación:
  1. Una relación o tabla almacena datos en filas y columnas. Todas las bases de datos contiene una o más tablas.

Figura 3.8: Esquema de base de datos relacionales
Fuente:Tomado de :[8][9]

2.Recopilación de datos y requisitos de procesamiento. Pueden combinar datos de diferentes tablas, actualizar los datos y realizar cálculos sobre ellos.
3. Los formularios administran la entrada y visualización de datos. Proporcionan señales visuales para ayudarlo a trabajar con los datos.
4. Informar resumir e imprimir datos. Convierta los datos en tablas y las consultas en documentos de comunicación de ideas.[8][9]

- **El modelo de datos orientado a objetos** es otro modelo de datos que está recibiendo una atención creciente, éste se puede observar como una extensión del modelo entidad-relación con los conceptos adicionales de encapsulación, métodos(funciones) e identidad de objetos, que son parte fundamental del diseño orientado a objetos. Las bases de datos orientadas a objetos se propusieron con la idea de satisfacer las necesidades de aplicaciones complejas, como por ejemplo estructuras complejas de datos, transacciones de mayor duración que las tradicionales y accesos a múltiples bases de datos. Las bases de datos orientadas a objetos permiten al diseñador especificar tanto la estructura de objetos complejos como las operaciones que se pueden aplicar entre los mismos. Una base de datos orientada a objetos provee una identidad única a cada objeto independiente almacenado en la base de datos y se parte de la base de que los objetos complejos pueden construirse a partir de otros más simples. A diferencia de las entidades en el modelo relacional, cada objeto tiene su propia identidad única independiente de los valores que contiene. Así, dos objetos que contienen los mismos valores son, sin embargo, distintos.

3.2.6. Patrones de diseño de software

En ingeniería de software, la solución estándar se denomina patrón de diseño. Eficiencia y reproducibilidad comprobadas para un problema común en el diseño de software. El estilo de diseño es una descripción o patrón de resolución de problemas que se puede utilizar en una variedad de situaciones. El uso de patrones de diseño le permite acelerar el proceso de desarrollo al proporcionar patrones comprobados en el diseño de software. La estandarización de soluciones para problemas comunes también mejora la comunicación

entre los desarrolladores, y cuando se utilizan nombres familiares y comprensibles para describir la solución y el problema de diseño. Los modelos se dividen en diferentes grupos según el tipo de problema que resuelven:

Patrones creacionales

Estas son aquellos que facilitan el trabajo de crear nuevos objetos, para que el proceso de construcción se puede separar del resto de la implementación del sistema. Los modelos creativos se basan en dos conceptos:
1. Encapsular el conocimiento de ciertos tipos utilizados por el sistema . Estos patrones generalmente funcionarán con interfaces, por lo que las implementaciones específicas de que usamos están aisladas.
2. Oculte cómo se deben crear estas implementaciones particulares y cómo se combinan. Los modelos creativos más famosos son:

- Abstract Factory: Proporciona una interfaz que permite crear una colección de objetos relacionados con sin tener que reconocerlos en ningún momento.
- Factory Method: Muestra cómo crear y delegar en subclases que implementa este método.
- Builder: Separa la creación de objetos complejos de su estructura. Puede crear una representación usando el mismo proceso de construcción
- Singleton: limite el número de posibles instancias de clase a uno programa y proporcionar acceso global a él.
- Prototype: Permite la creación de objetos basados en "plantillas". Se crea un nuevo objeto copiando otro objeto.[17][12]

Patrones estructurales

Estas plantillas facilitan el modelado de nuestro software seleccionando formas. Donde unas categorías están relacionadas con otras. Estos son los modelos estructurales identificados por la Gang of Four:

- Adapter: Permite que dos clases con diferentes interfaces trabajen juntas, por un objeto intermediario con el que se comunican e interactúan.
- Bridge: Elimina la abstracción parcial de su implementación, por lo que ambos se puedan desarrollar de forma independiente.
- Composite:Facilita la creación de estructuras de objetos en forma de árbol, donde todo los elementos utilizan la misma interfaz. Cada uno de ellos puede contener a su vez Una lista de estos objetos.

- le permite agregar funcionalidad adicional a un objeto (dinámico o estático) sin cambiar el comportamiento de otros objetos del mismo tipo. Es un objeto que crea una interfaz simple para trabajar en una pieza de código diferente y más compleja de manera simplificada y separada. Un ejemplo sería crear una fachada para trabajar con capas. bibliotecas externas.
- Flyweight:Múltiples objetos comparten el mismo objeto con propiedades generales de ahorro de memoria.
- Proxy: Aquí está la clase que actúa como una interfaz para todo lo demás: Conexión a Internet, archivos de disco o cualquier otro recurso que sea costoso o no se puede repetir.[17][12]

Patrones de comportamiento

La mayoría de las muestras están en el segundo grupo y se utilizan para el control. algoritmos, relaciones y responsabilidades entre objetos. Los patrones de comportamiento son:

- Command: Son objetos que encapsulan una acción y los parámetros que necesitan para ejecutarse.
- Chain of responsibility: evita combinar el remitente y el receptor de una solicitud, lo que hace posible que algunos destinatarios utilicen la solicitud. Cada destinatario tiene la opción de usar esta solicitud o pasarla a la siguiente persona en la cadena.
- Interpreter: Define una gramática, así como el mecanismo para evaluarla. Los árboles de sintaxis del lenguaje a menudo son modelados por el modelo.
- Iterator: Se utiliza para navegar por los elementos del conjunto en consistencia sin tener que revelar su implementación específica.
- Mediator:Un objeto encapsula cómo una colección de otros objetos interactúan y se comunican entre sí.
- Memento: Este patrón otorga la capacidad de restaurar un objeto a un estado anterior
- State: Permite modificar la forma en que un objeto se comporta en tiempo de ejecución, basándose en su estado interno.
- Strategy: Permite la selección del algoritmo que ejecuta cierta acción en tiempo de ejecución.
- Template Method: Especifica el esqueleto de un algoritmo, permitiendo a las subclases definir cómo implementan el comportamiento real.

- Visitor:Separe el algoritmo de la estructura de datos que se utilizará para ejecutarlo. De esta manera, se pueden agregar nuevas características a estas estructuras. sin tener que modificarlos. [17][12]

4 Tecnologías Utilizadas y Alternativas

4.1. Elección de la plataforma de desarrollo móvil

A la hora de iniciar las practicas se nos informo sobre la plataforma para la continuación del desarrollo de la aplicación en la que el proyecto iba a funcionar, dicha plataforma escogida fue el SDK de flutter . La aplicación del sistema se desarrolla de forma nativa, la decisión basada en hechos se basa en los dos factores más determinantes: rendimiento y entrega. [13]

4.1.1. Flutter

Figura 4.1: Logo de Flutter
Fuente:Tomado de [13]

Flutter es una plataforma de desarrollo de aplicaciones móviles multiplataforma creada por el navegador de Google. Es un software de código abierto que le permite desarrollar aplicaciones tanto para Android como para iOS. Su versión 1.0 se lanzó al mundo el 4 de diciembre de 2018, por lo que es una tecnología completamente nueva. A pesar de su corta edad, es una tecnología muy madura porque Google la usa para construir herramientas internamente.
Está integrado en clases, estando escritas en C / C++ y bibliotecas en Dart. El propósito de este SDK es permitir a los desarrolladores crear aplicaciones a partir de una base de

código que se ejecuta en código nativo para cada plataforma. Además, aprovecha la flexibilidad de Dart en el montaje y la implementación para ciclos de desarrollo más rápidos y tiempos de implementación más cortos.
A diferencia de otras soluciones, la aplicación está completamente construida en Flutter con Dart, incluida la interfaz de usuario. Para hacer esto, depende principalmente del modelo de la programación orientada a objetos, más concretamente en composición y herencia , que constituye interfaces de usuario. Estos elementos son esenciales para cualquier aplicación móvil. Lo que distingue a Flutter de otras soluciones es que no utiliza las herramientas de la plataforma, sino que proporciona las suyas propias. Flutter crea widgets a nivel de aplicación, lo que permite a los desarrolladores personalizarlos y expandirlos fácilmente, dándoles más libertad en el diseño de aplicaciones. Esto es posible porque Flutter solo necesita fondo de lienzo para "dibujar.[el]ementos de la interfaz de usuario.[10]

4.1.2. Alternativas de plataforma para desarrollo móvil

Desarrollaremos esta aplicación móvil para iOS y Android usando Flutter. Sin embargo, existen otras herramientas en el mercado para desarrollar aplicaciones de multiplataforma. En las siguientes secciones, se mostrarán algunos de ellos.

React Native

React Native, un framework creado por Facebook, se usa principalmente para crear aplicaciones móviles, pero también permite el trabajo multiplataforma. La idea principal es poder utilizar la conocida biblioteca React. Creación de una interfaz de usuario impulsada por Facebook en una plataforma nativa Por lo tanto, una aplicación creada con este motor utiliza componentes de React como interfaz de usuario y JavaScript como lógica. Para decirlo de otra manera, puede usar la misma biblioteca que se usa para desarrollar aplicaciones web. Sin embargo, la biblioteca React tiene una curva de aprendizaje claramente definida, lo que dificulta la aceptación de los nuevos usuarios. Las aplicaciones desarrolladas con este marco no están incluidas en la plataforma. Por otro lado, su código es interpretado en tiempo real por el dispositivo. Esto hace que sea difícil encontrar errores en su código durante el desarrollo. [11]

Ionic

Es un marco basado en la tecnología de código abierto Apache Cordova que permite la creación de APP híbridas. Las aplicaciones híbridas se implementan utilizando tecnología web (HTML, CSS y JavaScript) y se ejecutan en el dispositivo de destino utilizando las

Figura 4.2: Logo de React Native
Fuente:Tomado de [11]

capacidades del motor del navegador. El código HTML utilizado para definir la interfaz que se muestra en tiempo de ejecución, como el código JavaScript, se interpreta rápidamente en el dispositivo. La principal desventaja de este enfoque, similar a React Native, es una pérdida de rendimiento. Sin embargo, al comparar estas dos herramientas,React Native es mejor ya que se basa en un solo lenguaje, JavaScript, en lugar de la combinación de tecnologías utilizadas por Iónico. [26]

Figura 4.3: Logo de Ionic
Fuente:Tomado de [26]

Xamarin

En sus inicios, Xamarin era una empresa que desarrollaba aplicaciones de la plataforma Microsoft .NET para dispositivos móviles. Esta empresa fue adquirida por Microsoft, y su nombre fue utilizado para definir un conjunto de herramientas que permiten a los desarrolladores para usar el lenguaje de programación C# para crear aplicaciones multiplataforma, permitiendo que el código sea compartido entre ellos. Con Xamarin, puede usar C# también para proporcionar controles de interfaz personalizados para cada plataforma[5].Pero, esto hace que sea imposible implementar estas interfaces en C#; sin

embargo, se pueden compartir entre plataformas, lo que permite compartir el porcentaje del token se comparta Mucho más pequeño que otras soluciones, como Flutter. Finalmente, otro inconveniente de Xamarin es el gran tamaño de sus aplicaciones. Las aplicaciones creadas con este motor están hechas para cada plataforma, pero siguen siendo más grandes que la aplicación original. [4]

Figura 4.4: Logo de Xamarin
Fuente:Tomado de [4]

4.1.3. Dart

Figura 4.5: Logo de Dart
Fuente:Tomado de [27]

Usaremos Dart para la implementación porque es el lenguaje utilizado en Flutter. Esta decisión también fue alentada por la simplicidad del lenguaje y la similitud que tiene a otros lenguajes conocidos como Java y JavaScript.
Dart es un lenguaje de programación de código abierto desarrollado por Google que permite a los programadores utilizar lenguajes orientados a objetos. Desde su primer lanzamiento estable en 2011, Dart ha cambiado mucho, en términos de lenguaje y objetivos principales. Con la versión 2.0, el sistema de escritura Dart pasó de opcional a estático, y desde la llegada de Flutter ha sido el objetivo principal del lenguaje. [27]
Este es el lenguaje utilizado para desarrollar aplicaciones con Flutter. ¿Por qué este

lenguaje es perfecto para esta herramienta? La razón principal es su versatilidad. Dart le permite agregar código AOT (Ahead Of Time), lo que da como resultado un código nativo con un rendimiento mucho mejor que el uso de un lenguaje comprensible como JavaScript. El código Dart también se puede compilar sobre la marcha (JIT, Just In Time), lo que le permite usar el compilador rápido de Flutter. para actualizar el código mientras se está ejecutando.
Otro aspecto importante es el hecho de que Dart, al igual que JavaScript, funciona en uno. hilo único, impidiendo la ejecución de código seguro. Esto es especialmente importante cuando se implementan interfaces o conexiones entre el cliente y el servidor porque Esto permite a los desarrolladores asegurarse de que las funciones críticas se implementen por completo.

4.1.4. Visual Studio Code

Figura 4.6: Logo de Visual Studio Code
Fuente:Tomado de [2]

Visual Studio Code es un editor de código que le permite trabajar con una variedad de lenguajes de programación, administrar sus métodos abreviados de teclado y refactorizar su código. Es gratuito y de código abierto y nos proporciona un medio para descargar y administrar extensiones con las que podemos personalizar y mejorar la herramienta. Visual Studio Code nos brinda muchas opciones, como pestañas de colores, etiquetas . También hay extensiones que nos ayudan con los lenguajes de programación que utilicemos, como Python, C/C, JavaScript, y así sucesivamente.
En este caso, usaremos las extensiones Dart y Flutter, que admiten tanto el lenguaje como las funciones. Lo mejor de todo es que le permiten iniciar la aplicación en la que

está trabajando y depurar el código directamente desde el editor.[2]

Alternativa Android Studio

Aunque solo pudimos desarrollar aplicaciones con Android Studio porque este IDE debe instalarse para cargar las herramientas de Android, se eligió Visual Studio Code por su velocidad. El número limitado de funciones del editor de texto lo hace mucho más rápido. Además, no tendremos que usar la mayoría de las funciones de Android Studio. Android Studio es un IDE o entorno de desarrollo integrado oficial español para plataforma Android. Se utiliza para crear aplicaciones nativas El 16 de mayo de 2013, se anunció la conferencia Google I/O para reemplazar a Eclipse como el IDE oficial para desarrollar aplicaciones de Android.

Figura 4.7: Logo Android Studio
Fuente:Tomado de [21]

4.1.5. JSON

Para la transmisión de datos entre nuestra aplicación móvil y nuestro servicio en línea, usaremos JavaScript Object Notation(JSON) como formato.

Al enviar datos a través de la red, debemos usar un formato para describir la información que tanto el remitente como el receptor entienden para comprender el contenido. Uno de esos formatos es JSON. fue hecho para ser fácil de generar y procesar por máquinas y fácil de leer y escribir para humanos. Aunque está basado en JavaScript, es un formato de texto completamente independiente del idioma utilizado. Consta de dos cons-

tructos: objeto y conjunto. Los elementos no están en un orden particular y no tienen pares clave-valor. La figura muestra la sintaxis utilizada para guardar el objeto.[21]

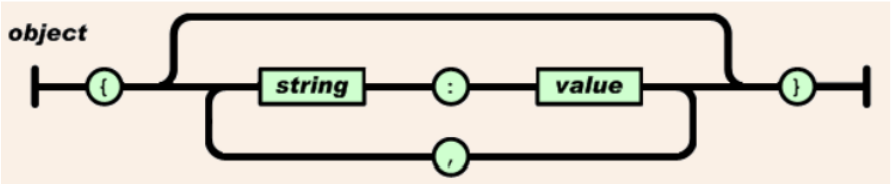


Figura 4.8: Definición de un objeto JSON
Fuente:Tomado de [21]

En JSON, un valor puede ser un par de palabras, un número, un valor booleano (verdadero o falso), un valor valor nulo, o incluso objetos y colecciones. Esto permite que las estructuras sean anidadas, lo que permite representaciones más complejas.

4.1.6. Servidor Cloud

AWS es la plataforma de nube en la que se estará desarrollando para el manejo de datos que requiere la aplicación a desarrollar en las practicas. Este tipo de sistema de información es esencial para el proyecto actual ya que permite al usuario enviar y guardar datos en tiempo real como velocidad, coordenadas y tiempos, entre otras cosas. [5]

Figura 4.9: Logo Amazon
Fuente:Tomado de [5]

Amazon Web Services

Es la plataforma en la nube más completa y confiable del mundo con más de 200 servicios en todo el mundo, servicios de infraestructura como almacenamiento, redes, bases de

datos, servicios de aplicaciones, informática avanzada, mensajería, inteligencia artificial, servicios móviles, seguridad, identidad e Internet. cosas incluidas. Como resultado, las empresas reciben servicios innovadores a tiempo y con un costo mínimo.[5]

De dicha plataforma tomaremos el conjunto de herramientas llamado AWS Amplify, el cual nos permitirá tener conexión aplicación con un Backend nube.

- **Amplify**
  AWS Amplify es un conjunto de herramientas y características diseñadas específicamente para desarrolladores web y móviles. Con Amplify, puede configurar backends de aplicaciones web y móviles, conectar aplicaciones en minutos, crear interfaces de usuario web fáciles de usar y administrar fácilmente el contenido de las aplicaciones fuera de la consola de AWS. Carga más rápida y fácil medición. No se requiere experiencia en la nube.[5]

  La herramienta tomada en nuestro fue Amplify DataStore que se utiliza como un motor de almacenamiento y esta sincroniza datos entre la aplicación y la nube.

  - **Amplify DataStore**
    Amplify DataStore es un almacén de datos accesible desde el dispositivo desde el que pueden realizar consultas. Amplify DataStore proporciona un modelo de programación que aprovecha los datos distribuidos y compartidos sin código adicional en escenarios en línea y fuera de línea, lo que ayuda a procesar los datos distribuidos entre los usuarios y el uso de datos. localmente más fácilmente, lo que permite a los desarrolladores crear excelentes aplicaciones.[5]

Alternativa

Una posible alternativa que se a de tener en cuenta son los servicios de Huawei Cloud, aunque en le momento no se conoce sobre estos servicios, dicha empresa nos dará una capacitación de los servicios que ofrecen y así poder analizar si es viable una migración a ella.
Huawei Cloud: es una plataforma de desarrollo segura, dinámica y de acceso rápido que le permite crecer en un entorno estable al mismo tiempo que adquirir recursos informáticos y de almacenamiento en la nube. La arquitectura hace uso de software y tecnología avanzados, incluidos servidores, dispositivos de almacenamiento y dispositivos de red, así como virtualización y herramientas de gestión de redes.

Figura 4.10: Logo Google Maps
Fuente:Tomado de [3]

API de Google Maps

La API de Google Maps es una colección de API que le permiten superponer sus propios datos en un mapa personalizado de Google Maps. Con la poderosa plataforma de mapas de Google, puede crear atractivas aplicaciones en línea y móviles, completas con imágenes satelitales, Street View, perfiles de elevación, direcciones, mapas con estilos, datos demográficos, análisis y una gran base de datos de ubicación. Los usuarios se beneficiarán de un servicio que siempre está mejorando, gracias a la cobertura mundial más precisa del mundo y a una comunidad de mapas activos que incorpora actualizaciones diarias.[3]

4.1.7. Google Play

Google Play es la principal tienda de distribución de aplicaciones de Android. Es desarrollado y operado por Google LLC. Las aplicaciones publicadas en Google Play se pueden usar en teléfonos inteligentes, tabletas, televisores inteligentes y relojes inteligentes que se ejecutan en el sistema operativo Android y se descargan desde Play Store. Pero no solo aplicaciones, también puedes encontrar juegos, películas, música, libros y revistas disponibles en Play Games, Play Movies, Play Music, Play Books y Play Kiosco, respectivamente. Las aplicaciones y otros contenidos de la plataforma son gratuitos y premium. Google Play se basa en tecnología de computación en la nube donde todo su contenido está alojado en servidores dedicados. Además, permite a los desarrolladores poner sus aplicaciones a disposición de más de mil millones de usuarios activos de Android en más de 190 países y territorios de todo el mundo.[22]

5 Fase de planificacion

5.1. Análisis Requisitos

Los requisitos son la base de toda gestión de proyectos. Todos los demás aspectos de un proyecto se basan en los requisitos. Deben ser claros e inequívocos para evitar malentendidos y complicaciones en el futuro.

Un requisito es una condición que debe cumplirse o una capacidad que debe estar presente para que un elemento, proceso o sistema satisfaga las necesidades establecidas o implícitas.

Los requisitos de una aplicación se pueden dividir en dos categorías: requisitos funcionales y requisitos no funcionales. Los requisitos funcionales describen lo que hace el sistema, mientras que los requisitos no funcionales describen cómo debe comportarse.
Se describirá la funcionalidad e interacción que tendrán los usuarios con la aplicación móvil.
Se comentaran los requerimientos que en el momento han sido asignados en dichas practicas, dicha requerimientos existen otros como acceso y registro ,entre otros, pero no los mencionare ya que no son de mi parte a desarrollar , es un trabajo colaborativo.

5.1.1. Requerimientos funcionales

Los requisitos funcionales consisten en declarar los servicios prestados por la plataforma, de forma que los servicios utilizados por el usuario se puede identificar. A continuación en la tabla se mostraran estos requerimientos.

Cuadro 5.1: Requerimientos Funcionales.

	Requerimientos Funcionales	
Ubicación y Rutas		
	Google Maps	Mostrar mapa de Google Colocar destino o buscarlo y seleccionarlo
	Búsqueda de rutas	Rutas encontradas mostrarán la siguiente información: nombre de la ruta, nivel, distancia y tiempo.
	Generar ruta por defecto de Google Maps	-
	Iniciar ruta	Poder utilizar una ruta generada
	Generar indicaciones	Se mostrara indicaciones en la aplicación de derecha e izquierda según sea el giro que deba realizar
	Enviar indicaciones a Dispositivo Bigo	Se enviaran indicaciones vía bluetooth al Dispositivo Bigo
	Crear Ruta libre	Permite crear una ruta personalizada según un recorrido que hagas
Almacenamiento en nube		
	Guardar Rutas personalizadas	Permite guardar las rutas personalizadas en servidor de nube AWS
	Mostrar ruta guardada	Permite extraer de la nube las rutas guardadas y mostrarlas en mapa.
	Compartir Rutas	A travez de los datos guardados en nube compartiremos rutas personales a otros usuarios

5.1.2. Requisitos no funcionales

Los requisitos no funcionales son aquellos que no se refieren directamente a las funciones específicas proporcionadas por el sistema, sino más bien a la propiedades que surgen de él, como la confiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. A continuación en la tabla se mostraran estos requerimientos.

Cuadro 5.2: Requerimientos no Funcionales.

	Requerimientos no Funcionales	
Sistema		
	Version Android	El sistema funcionara para cualquier versión de sistema operativo Android.
	Conexión a internet	Para que el sistema funcione correctamente, debe tener una conexión a Internet. Se necesita acceder a mapa de Google y acceder a servicios de nube.
	Ubicación	El sistema debe usar GPS para determinar la ubicación del usuario.
	Bluettoh	Para poder conectarse a dispositivos, el sistema debe utilizar Bluetooth.
Interfaz		
	Texto	Se debe tener una fuente clara y legible
	Visual	El sistema debe tener interfaz simple y organizada para la comprensión del usuario.
	Notificaciones	El sistema debe enviar notificaciones push del usuario.
Rendimiento		
	Velocidad	Se espera que el tiempo de respuesta entra pantallas de aplicación sea eficiente
	Almacenamiento	Los datos se almacenaran de forma persisten te y se mostraran la información recuperada directamente del servidor, por lo tanto siempre estará actualizada.
	Error	En caso de error, se mostrara un mensaje

5.2. Fase de diseño

La práctica básica es utilizar diseños tan simples como sea posible. El principio es usar el diseño más simple que haga que todo funcione y evitar agregar funciones que consumen mucho tiempo.
En este paso, usaremos el modelo UML porque se puede usar para modelar diferentes tipos de sistemas: sistemas de software, hardware y organizaciones reales. UML proporciona muchos tipos de diagramas de modelado de sistemas y los desarrolladores lo eligen porque es un lenguaje que ayuda a analizar problemas y soluciones relacionados con la construcción de sistemas y porque es un modelo de programación universal, variable y el más utilizado.
Para el proyecto se realiza esquemas más apropiados tal como:

- Diagramas de Casos de Uso.

Además de los modelos UML, planeamos complementar nuestro proyecto con modelos arquitectónicos para brindar una vista simplificada del sistema, de modo que una persona pueda mirar el diagrama y comprender de inmediato lo que quiere lograr o desarrollar, y el Mapa de Navegación para ayudarnos a comprender cómo se muestran las pantallas de nuestra aplicación con contenido y enlaces en cada pantalla.

5.2.1. Diagrama de Casos de uso

En esta sección, se presenta el proceso de análisis y diseño de cómo un sistema interactúa con un usuario o un grupo de sistemas en respuesta a un evento, mostrando, en particular, sus relaciones y comportamiento.Este diagrama tiene como objetivo ilustrar los requisitos funcionales, destacando que el esquema propuesto depende de si el usuario está logueado.

En la Figura 5.1 se muestra un Diagrama UML con casos de uso de la aplicacion.

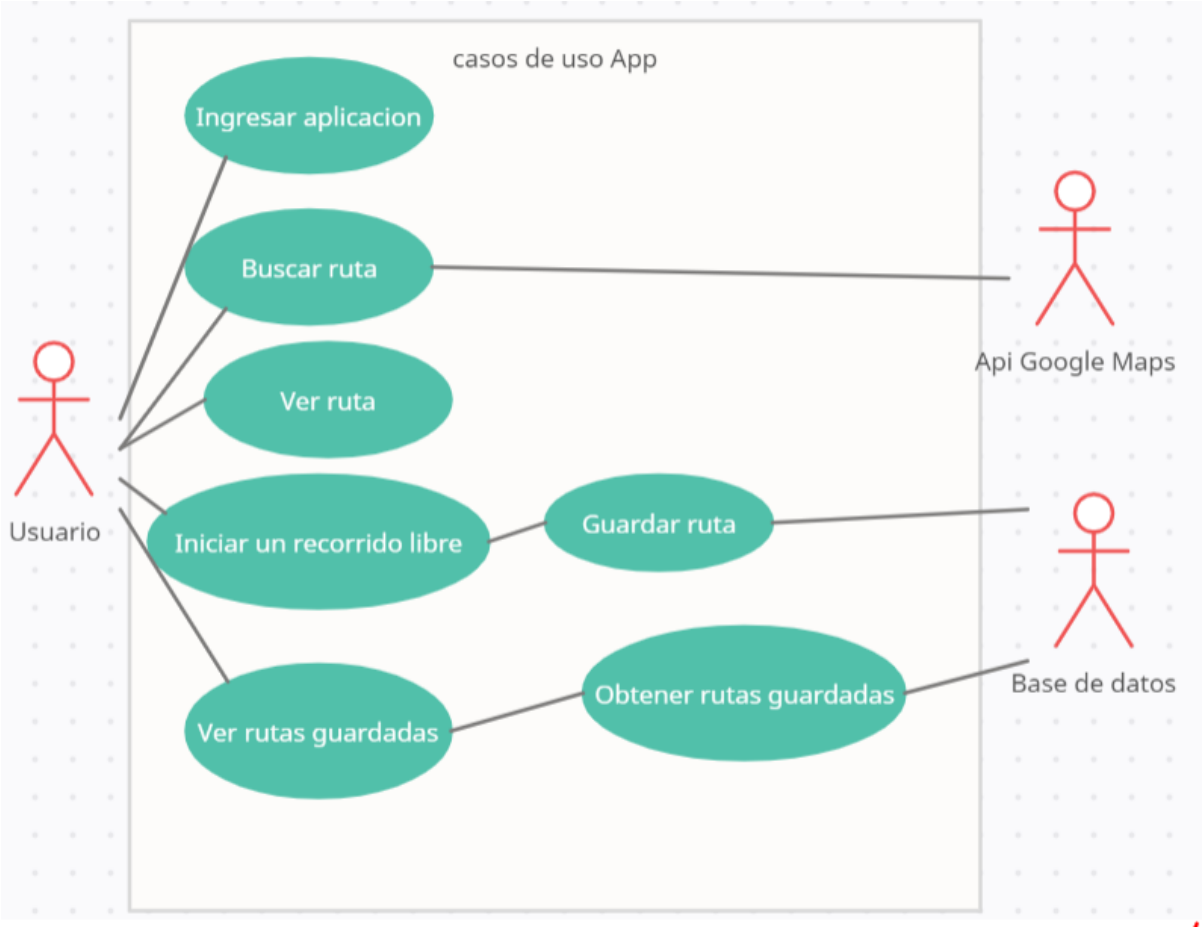


Figura 5.1: Casos de uso
Fuente:Elaboración propia

Caso de uso numero 1	
Nombre	Ingresar a la Aplicación
Precondición	- Conexión a la base de datos - Tener un usuario registrado - Que la aplicación se haya ejecutado
Flujo del sistema	Acción
	1. La Aplicación despliega ventana dew Inicio de sección 2. El usuario ingresa su respectivo usuario y clave 3. La aplicación valida los datos ingresados ala base de datos 4. La aplicación muestra la interfaz gráfica al vendedor

Cuadro 5.3: Caso de uso numero 1
Fuente: Elaboración propia

Caso de uso numero 2	
Nombre	Buscar ruta
Precondición	- Conexión a internet - Conexión Api Google Maps
Flujo del sistema	Acción
	1. El sistema muestra el menú principal 2. El usuario pulsa el botón “Rutas” 3. El usuario pulsa Opción “iniciar ruta” 4.La aplicación accede a la pantalla de búsqueda de rutas por lugar 5. El usuario selecciona un lugar encontrado

Cuadro 5.4: Caso de uso numero 2
Fuente: Elaboración propia

Caso de uso numero 3	
Nombre	Ver ruta
Precondición	El usuario debe haber seleccionado un lugar que contiene rutas en la pantalla de búsqueda de rutas
Flujo del sistema	Acción
	1. La Aplicación muestra ruta encontradas en la pantalla de búsqueda de rutas 2. La aplicación accede a la pantalla de detalle de ruta

Cuadro 5.5: Caso de uso numero 3
Fuente: Elaboración propia

Caso de uso numero 4	
Nombre	Iniciar un recorrido libre
Precondición	Ubicacion en tiempo real
Flujo del sistema	Accion
	1. El usuario realiza un recorrido de una ruta que quiera guardar 2. El usuario finaliza ruta y esta se guarda en servidor 3. La aplicación da una visualización de la ruta realizada

Cuadro 5.6: Caso de uso numero 4
Fuente: Elaboracion propia

Caso de uso numero 5	
Nombre	Ver rutas guardadas
Precondicion	Conexión con Servidor
Flujo del sistema	Acción
	1. El usuario selección opción “Rutas guardadas” 2. El usuario finaliza ruta y esta se guarda en servidor 3. La aplicación da una visualización de la ruta realizada

Cuadro 5.7: Caso de uso numero 5
Fuente: Elaboración propia

5.3. Arquitectura

Crear una plataforma con una comunicación eficiente entre los componentes , se tiene en cuenta la arquitectura del software, Quién es responsable de proporcionar la información de los componentes, donde características, funciones y conexiones entre cada uno de ellos uno de los diferentes componentes.

El sistema que conforma la aplicación así como los sistemas externos de los que hace uso, se compone fundamentalmente de dos partes:

- Aplicación Android
- Api Google Maps
- Bakend

En la Figura 5.2 se muestra un Representación de la Arquitectura del sistema.

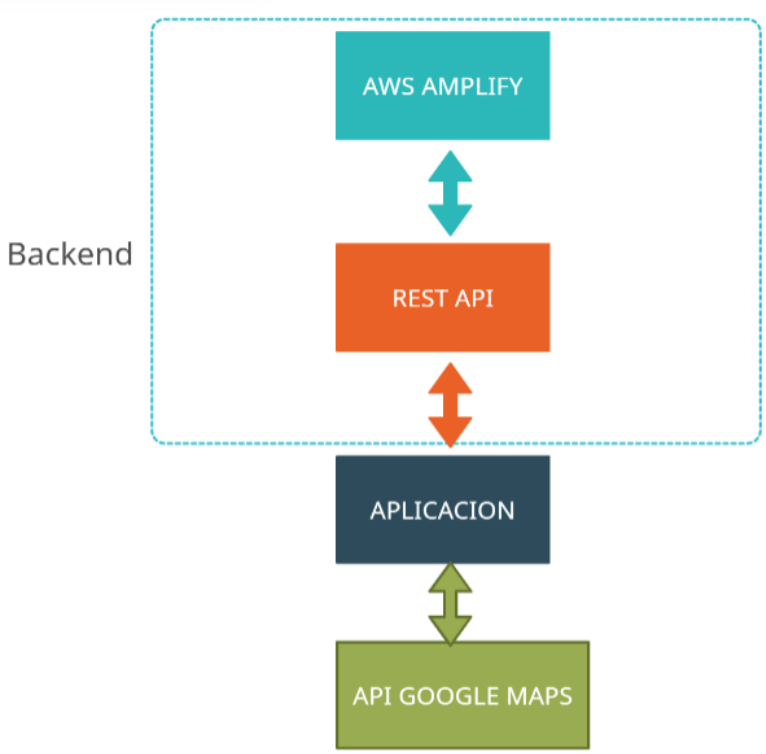


Figura 5.2: Arquitectura del sistema
Fuente:Elaboración propia

Como puede verse en la Figura 5.2, los componentes de la aplicación está separada

entre la propia aplicación,parte de la nube alojada en Google Cloud Platform y conexión AWS Amplify . El servidor de la aplicación se encuentra alojado en la base Amazon AWS, que ejecuta el backend proporcionado por Amplify.
En concreto, la aplicación Android realiza tareas de geolocalizacion y visualización sobre datos GPS obtenidos, y ambos se conectan a través de la REST API como se muestra en la figura.
El cliente de Android utiliza la API REST de Amplify, con la que guarda o carga las rutas guardadas de los ciclistas en el historial de ciclismo de la aplicación. También utiliza el servicio de la API de GoogleMaps, la aplicación hace las llamadas y esta a su vez llama a otras API. La respuesta generada de todas ellas en un único json que es enviado de vuelta al dispositivo necesario para mostrar mapas y rutas en teléfonos inteligentes.

5.3.1. Tipo de arquitectura

Después de estudiar los requisitos de la plataforma, se seleccionó el estilo arquitectura. En este caso se define una arquitectura de 3 capas que se puede observar en la Figura 5.3, ésta la componen la capa de presentación, capa de negociación y capa de datos.

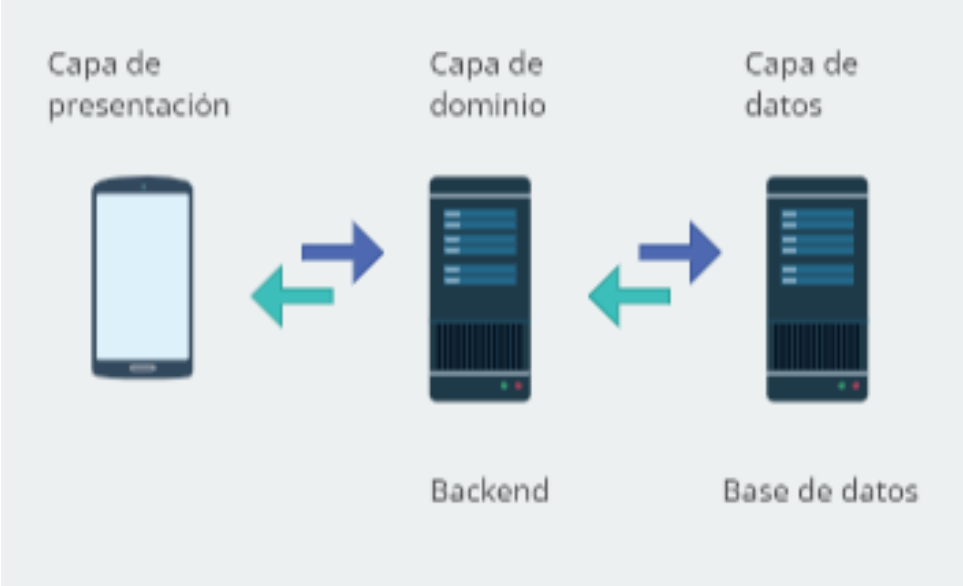


Figura 5.3: Mapa de navegación
Fuente:Elaboración propia

Como se muestra en la figura anterior, la arquitectura consta de tres capas: la capa de presentación, la capa de negociación y la capa de datos. La capa de presentación

es responsable de mostrar la información del usuario, comunicarse con la plataforma y recopilar datos del usuario.
La capa de negociación es donde se ubican los programas a ejecutar, llamada así porque es donde se deben establecer todas las reglas. Esta capa se comunica con la capa de visualización para recibir solicitudes y proporcionar resultados, y con la capa de datos para indicar al administrador de la base de datos que almacene o recupere datos.
Finalmente, la capa de datos es responsable del respaldo y acceso a los datos. Incluye un administrador de base de datos que realiza todo el almacenamiento o recuperación de información de la capa empresarial.

Capa de presentación

Como se sabe, la capa de presentación es todo lo que el usuario puede ver y la base se crea en función de los criterios de usabilidad y diseño que se tienen en cuenta.

Capa de dominio

Esta clase es principalmente responsable de permitir que el usuario obtenga información sobre el sistema donde recibe la solicitud del usuario y envía la respuesta. Para que los usuarios puedan enviar solicitudes y recibir información de la mejor manera, el backend se ha desarrollado para crear todos los servicios de manera óptima y con alta confiabilidad. Esta capa contiene los servicios de plataforma necesarios, algunos de los cuales se enumeran en la Tabla 5.8.

Tabla de servicios	
Nombre del servicio	**Descripción del servicio**
Crear ruta	Este servicio permite crear una ruta seleccionada por un usuario.
Guardar ruta	Este servicio permite al usuario guardar sus propias rutas.
Obtener rutas	Este servicio permite obtener todas las rutas que se encuentran guardadas en la base de datos sin obtener información de los usuarios.

Cuadro 5.8: Tabla de servicios
Fuente: Elaboración propia

Capa de datos

En la capa de datos, hay un servicio de contenedor de base de datos no relacional en Amplify, y eso es para obtener datos de forma segura, ya que solo se puede acceder a la información desde la capa de dominio. Su función es poder almacenar todos los datos del usuario y toda la información de latitud y longitud del mapa, relacionadas con carreteras, zonas de accidentes y zonas de vuelo.
Hablando específicamente de la base de datos, se decidió que no debería ser relacional porque Amplify permite guardar datos en grupos que contienen tablas de longitud y latitud en formato JSON y esta es una característica de la tabla generada en un conjunto de información generada por el usuario.

5.3.2. Mapa de navegación

El mapa de navegación se refiere o muestra el orden de las relaciones de las pantallas , y se utilizan para comprender el orden en que se muestran las pantallas y el contenido de la aplicación. La Figura 5.4 muestra la relación entre los diferentes pantallas.

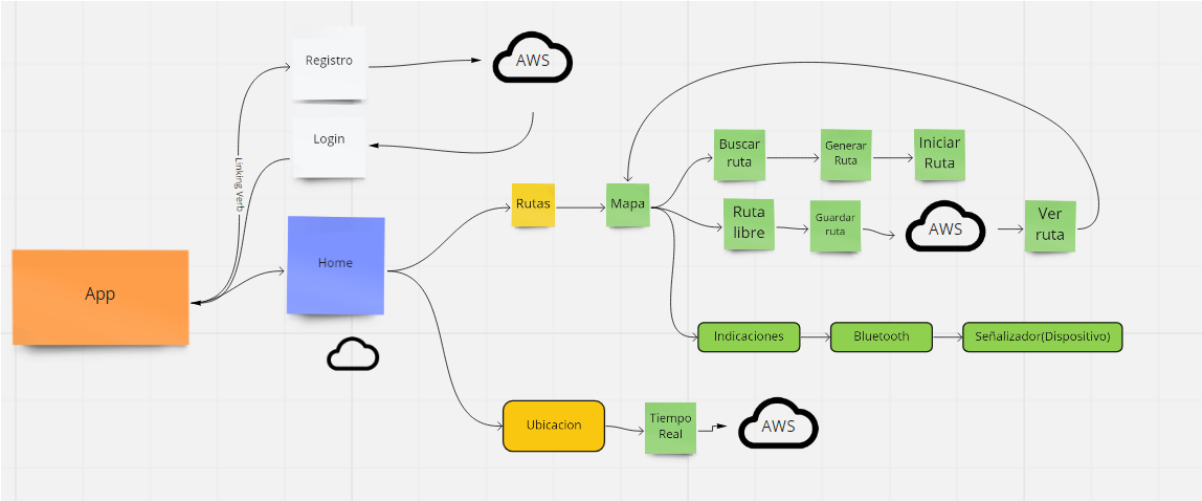


Figura 5.4: Mapa de navegación
Fuente:Elaboración propia

5.3.3. Arquitectura de la base de datos

Con el desarrollo de requisitos y casos de uso, se ha recopilado suficiente información para poder diseñar una tabla en la base de datos que incluya todo lo necesario para un desarrollo posterior.

Figura 5.5: Tabla routes_user
Fuente:Elaboración propia

Parametro	**Tipo**	**Descripción**
id_user	Int	Identificador de usuario en la base de datos
name_route	String	Nombre asignado a la ruta guardada
long	String Array	Arreglo de posiciones (longitude) generadas por el usuario en el recorrido.
lat	String Array	Arreglo de posiciones (latitude) generadas por el usuario en el recorrido.

Cuadro 5.9: Tabla route_user
Fuente: Elaboración propia

6 Implementación

Después de implementar los requisitos y características que debe seguir la arquitectura y definir la apariencia que desea, puede comenzar la implementación. Las siguientes secciones presentarán las partes más importantes del código y la interfaz para cada implementación.

6.1. Configuración del entorno de trabajo

Antes de comenzar a discutir las implementaciones de diferentes funciones, se guiara a través de los pasos necesarios para configurar nuestro entorno de trabajo, crear una aplicación con Flutter.
Lo primero que debe hacer es instalar el SDK de Flutter. Para hacer esto, siga las instrucciones aparece en el sitio web oficial de nuestro sistema operativo. Para Windows Simplemente descargue el archivo usando el SDK y extráigalo a una carpeta accesible y agregue el directorio bin a la variable de entorno PATH de nuestro sistema. Ahora podemos usar la herramienta flotante desde la línea de comando. En la figura 6.1 muestra la salida comando FLutter doctor que es verificacion de que todos los requisitos este correctos.

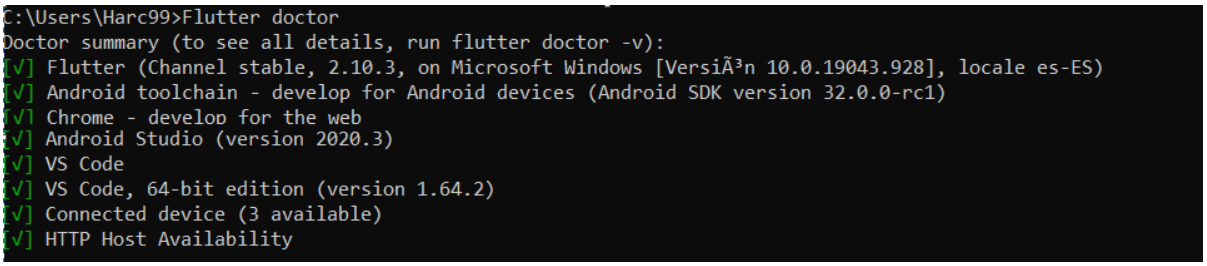


Figura 6.1: Salida comando Flutter doctor
Fuente:Elaboración propia

Después de instalar el SDK de Flutter, debe instalar las herramientas de Android para que compile nuestra aplicación en esta plataforma y pruebe nuestro código con un

simulador. La forma más fácil de instalar Android SDK y herramientas relacionadas es instalar Android Studio. Este Entorno de desarrollo integrado(IDE), que discutimos en la Sección 4.1.4, incorpora el SDK y demas herramientas necesarias.

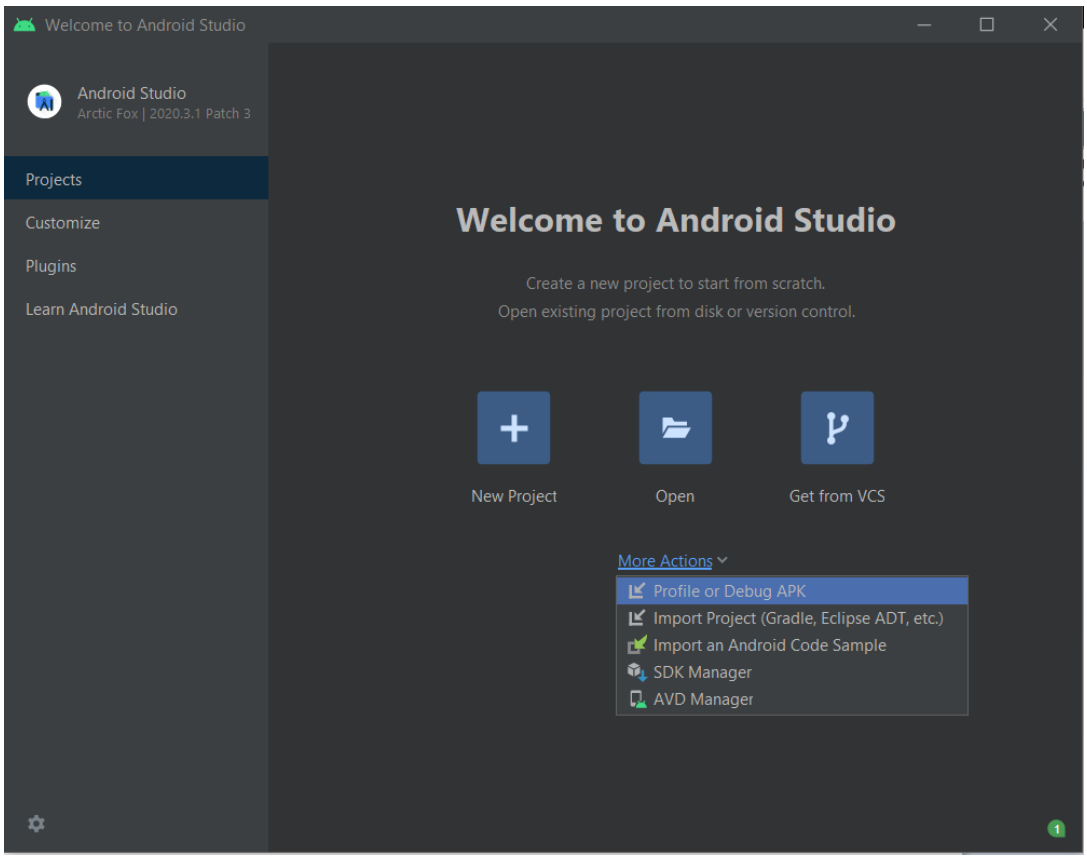


Figura 6.2: Pantalla inicio Android Studio
Fuente:Elaboración propia

Una de las herramientas más importantes es el emulador que podemos ejecutar Emular un teléfono Android corriendo en nuestra computadora personal y lo estamos usando Probar el código desarrollado. Podemos hacer emuladores para cualquier versión. SDK de Android que queremos desde Android Studio.
El editor utilizado será Visual Studio Code, que se discutió en la Sección 4.1.4. En Visual Studio Code, Dart y Flutter tenemos soporte completo para el idioma (formato, sugerencias y detección de errores) y acceso a herramientas de depuración de aplicaciones Flutter desde el editor. Esto nos permite ir emulador de Android y ver que la actualización de la aplicación se están reflejando en los cambios que se realizan en el editor.
Todos los códigos de nuestro proyecto estarán alojados en GitHub, por lo que lo necesitaremos crear una nueva cuenta y repositorio. Visual Studio Code incluye soporte listo para usar para Git, lo que nos permite estar listos para comenzar a desarrollar.

6.2. Construccion y codificacion

En esta sección se publican los resultados obtenidos al codificar la aplicación en lenguaje Dart, en base al diseño y funcionalidad propuestos en la etapa anterior junto con las herramientas utilizadas en la construcción del proyecto, en este caso.

6.2.1. FLutter

Se utiliza como complemento Flutter, una herramienta creada por Google y de código abierto en el editor Visual Studio Code, que me permite desarrollar un prototipo para el sistema operativo Android.

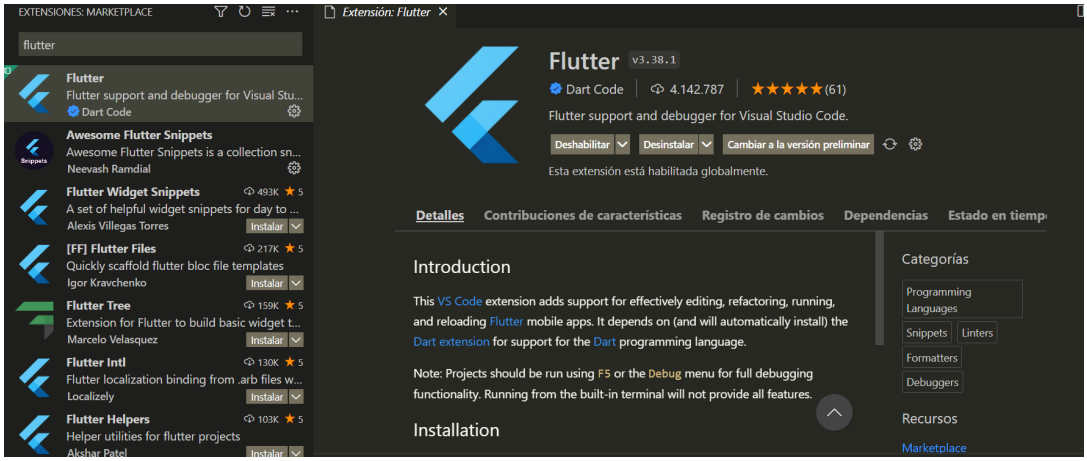


Figura 6.3: Extensión Flutter
Fuente:Elaboración propia

6.2.2. Estructura del proyecto

Dentro del proyecto existen archivos correspondientes a la arquitectura de la aplicación, que contienen las carpetas con la configuración de la APP y otros archivos como lib, que contiene todos los archivos fuente de nuestra interfaz.

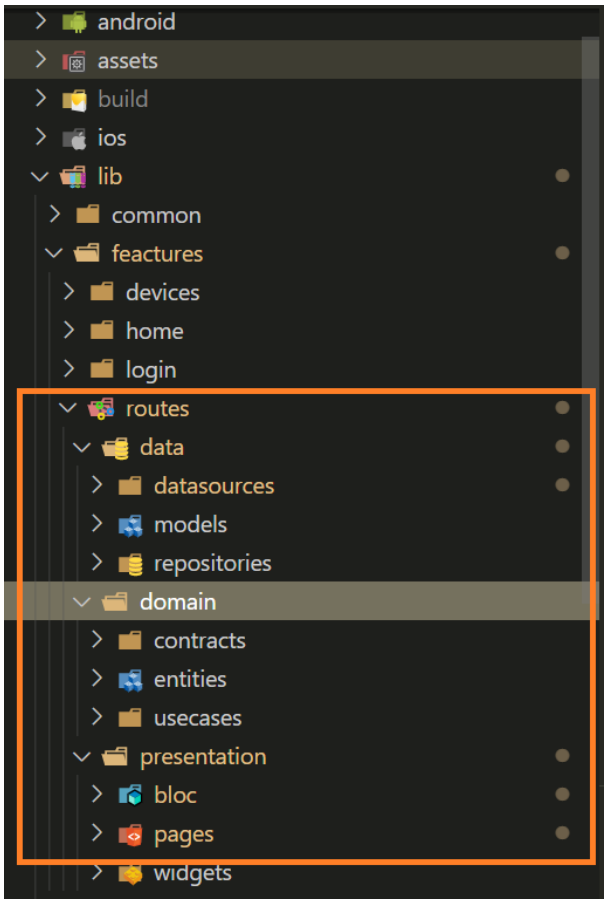


Figura 6.4: Estructura Aplicación
Fuente:Elaboración propia

Como se puede ver en la Figura 27, la estructura del proyecto va de acuerdo a lo mencionado en la sección 6.1, donde se menciono que el tipo de arquitectura es de 3 capas y en dicho proyecto se tiene creado carpeta para cada una de ellas.

6.2.3. API de Google Maps

Para esta sección ya que el proyecto depende en su mayoría del mapa y sus funciones de geolocalización y rutas. Para esto es necesario remitirse a la plataforma dispuesta por Google para la administración de recursos y servicios cloud; Para configurar el acceso a los recursos de Google Maps es necesario tener una clave para la API de Google Maps Platform. Para ello se requiere una cuenta de Google Clound Platform. Al cumplir estos requisitos basicos, el procedimiento para acceder a todos los componentes es sencillo.

1. Crea un nuevo proyecto en Google Clound Platform.

2. Agregue Google Maps Platform al proyecto.
3. Cree una clave de API para el acceso a Google Maps.
4. Agregue las API requeridas:
    - Maps SDK para Android - Se utiliza para ver mapas en una sola aplicación para la plataforma Android.
    - API de geolocalización: nos permite obtener una geolocalización más precisa, usando del sistema de Google (GPS total, Wi-Fi, datos móviles).
    - Maps Static:Nos permite obtener el tamaño exacto de ubicación en el mapa como una imagen
    - API de lugares: Nos muestra puntos de interés en el mapa para el usuario. También traduce coordenadas GPS (definido por el usuario) en direcciones físicas legibles por un humano como (calles, caminos...).
    - Geocoding API: le permite traducir una dirección escrita legible por humanos (calle, avenida...) en coordenadas. Se utiliza para traducir direcciones almacenadas en coordenadas

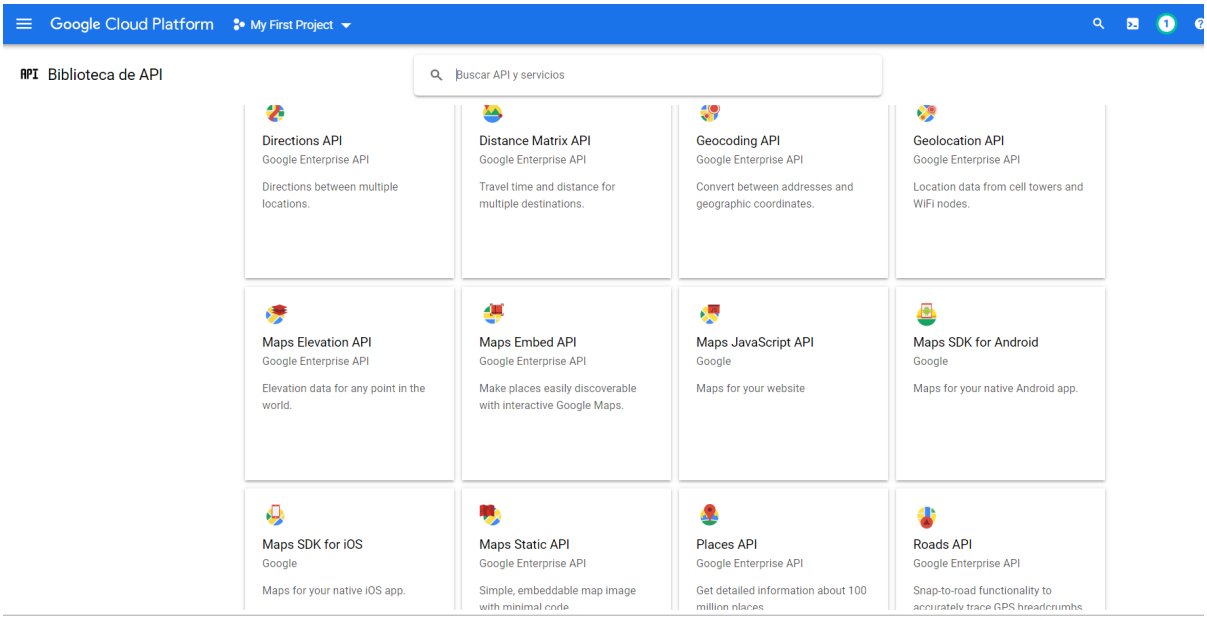


Figura 6.5: Opciones de API Google Mapas
Fuente:Elaboración propia

Para agregar el mapa a la aplicación se toma la credencial o clave creada en el paso 2 la cual va a identificar proyecto, el usuario y permite cargar todas las funciones a la aplicación, esta credencial se crea fácilmente en la opción “credenciales” y luego “crear

credenciales", de esta forma arroja una clave con el formato que se ve en la figura 29.

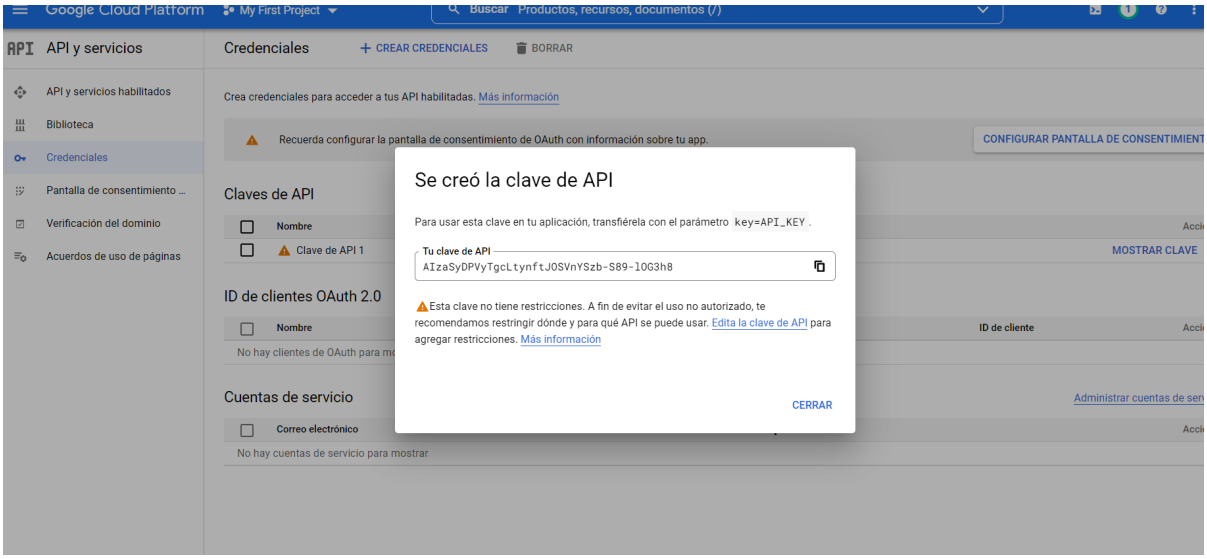


Figura 6.6: Creacion Credencial API
Fuente:Elaboración propia

Una vez que tenga las credenciales, debe integrarlas en la aplicación y configurar permisos necesarios para obtener ubicación. Como lo indica la figura 30.

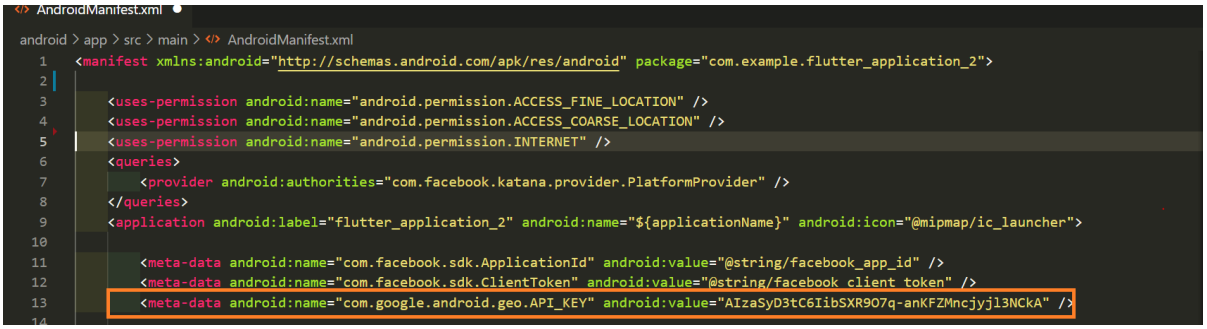


Figura 6.7: Integración clave API en aplicación
Fuente:Elaboración propia

6.3. Previsualización del Mapa

```
Widget build(BuildContext context) {
    return SafeArea(
  child: Stack(children: <Widget>[
    Positioned(bottom: 183, top: 0, right: 0, left: 0,
        child: Stack(  children: [
            Positioned( bottom: 0, top: 0, right: 0,left: 0,
              child: Container(
                child: GoogleMap(
                  myLocationButtonEnabled: false,
                  zoomControlsEnabled: false,
                  initialCameraPosition: CameraPosition(
                    target:LatLng(37.43296265331129, -122.08832357078792),
                    zoom: 15,
                  ),
                  onMapCreated: (GoogleMapController controller) async {
                    _controller.complete(controller);
                  },
                  myLocationEnabled: true,
                ),
              ), ),], )),
```

Figura 6.8: Codigo Previsualización mapa
Fuente:Elaboración propia

Figura 6.9: Previsualización mapa
Fuente:Elaboración propia

6.4. Integración del motor de almacenamiento

Otro paso importante en el proceso de creación de proyectos es la integración del motor de base de datos de AWS Amplify Datastore; A continuación se muestra la implementación e integración con la aplicación utilizando Flutter.
Para instalar y configurar su aplicación con Amplify DataStore, debemos cumplir con los requisitos previos

- Una aplicación Flutter dirigida a Flutter SDK mayor o igual 2.0.0
- Una configuración de iOS dirigida al menos a iOS 13.0
- Una configuración de Android dirigida al menos al nivel de API de Android 21 (Android 5.0) o superior.

6.4.1. Instalación Bibliotecas

A Continuación se procede a agregar dependencias a su archivo pubspec.yaml

```
2    sdk: ">=2.12.0 <3.0.0"
3
4  dependencies:
5    flutter:
6      sdk: flutter
7    amplify_flutter: ^0.4.0
8    amplify_datastore: ^0.4.0
```

Figura 6.10: Dependencias Amplify Datastore
Fuente:Elaboración propia

6.4.2. Configuración del entorno de desarrollo local

Para configurar se utiliza la CLI de Amplify, siendo esta una opción util para proyectos ya creados y configurados que se deseen agregar Amplify Datastore.
La estructura de la aplicación DataStore se crea agregando una nueva API GraphQL a la aplicación.

```
# For new APIs
amplify add api

# For existing APIs
amplify update api
```

Figura 6.11: Add Api GraphQL
Fuente:Elaboración propia

La CLI le pedirá que configure su API.

```
? Please select from one of the below mentioned services:
    `GraphQL`
? Here is the GraphQL API that we will create. Select a setting to edit or continue:
    `Name`
? Provide API name:
    `BlogAppApi`
? Here is the GraphQL API that we will create. Select a setting to edit or continue:
    `Authorization modes: API key (default, expiration time: 7 days from now)`
? Choose the default authorization type for the API
    `API key`
? Enter a description for the API key:
    `BlogAPIKey`
? After how many days from now the API key should expire (1-365):
    `365`
? Configure additional auth types?
    `No`
? Here is the GraphQL API that we will create. Select a setting to edit or continue:
    `Conflict detection (required for DataStore): Disabled`
? Enable conflict detection?
    `Yes`
? Select the default resolution strategy
    `Auto Merge`
? Here is the GraphQL API that we will create. Select a setting to edit or continue:
    `Continue`
? Choose a schema template
    `Single object with fields (e.g., “Todo” with ID, name, description)`
```

Figura 6.12: Configuración Api
Fuente:Elaboración propia

El primer paso para crear una aplicación habilitada para el almacenamiento de datos es definir un esquema. El almacén de datos utiliza archivos de esquema de GraphQL como definición del modelo de datos de la aplicación. El esquema contiene los tipos de datos y las relaciones que representan la funcionalidad de la aplicación.

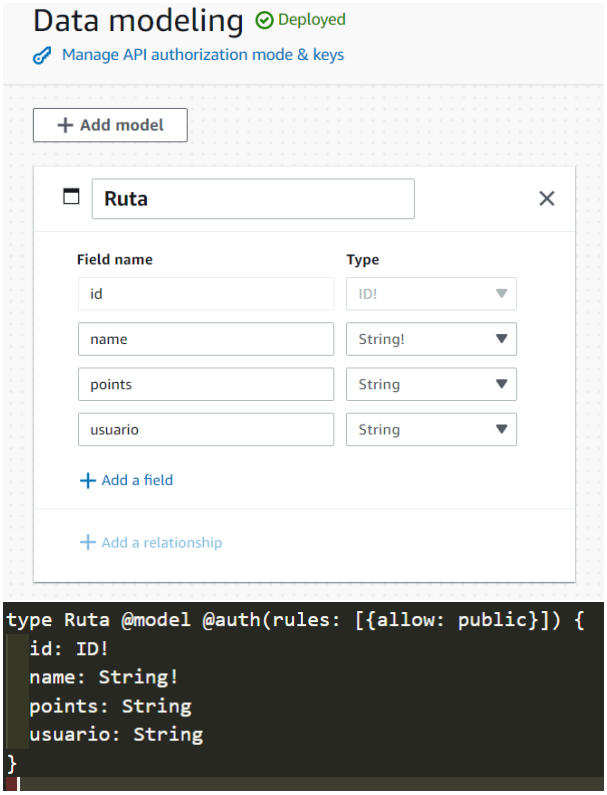


Figura 6.13: Esquema y modelo Ruta
Fuente:Elaboración propia

6.5. Interfaz

Con el boceto de pantallas suministrados por la empresa, se crean interfaces fáciles de usar con una excelente experiencia de usuario, y las interfaces creadas en el proyecto se muestran a continuación.
Explicaré cuáles son las pantallas que componen la aplicación y algunas de las funciones que contienen. Se definen widget para cada ventana, ya que asi es como se trabaja con con Flutter y asi cada ventana que la aplicación muestra son un conjunto de widgets(incluso la pantalla mismo) en toda la superficie disponible.

6.5.1. Pantalla de permisos ubicación

Al iniciar la aplicación para su correcto funcionamiento nos pedirá que aceptemos los permisos necesarios, el permiso que se exige es el de obtener la ubicación del usuario y además de se debe aceptar que la aplicación pueda también funcionar obteniendo ubicación en segundo plano, es decir la aplicación estará activa así esa este minimizada o el dispositivo este bloqueado.Al iniciar la aplicación para su correcto funcionamiento nos pedirá que aceptemos los permisos necesarios, el permiso que se exige es el de obtener la ubicación del usuario y además de se debe aceptar que la aplicación pueda también funcionar obteniendo ubicación en segundo plano, es decir la aplicación estará activa así esa este minimizada o el dispositivo este bloqueado.

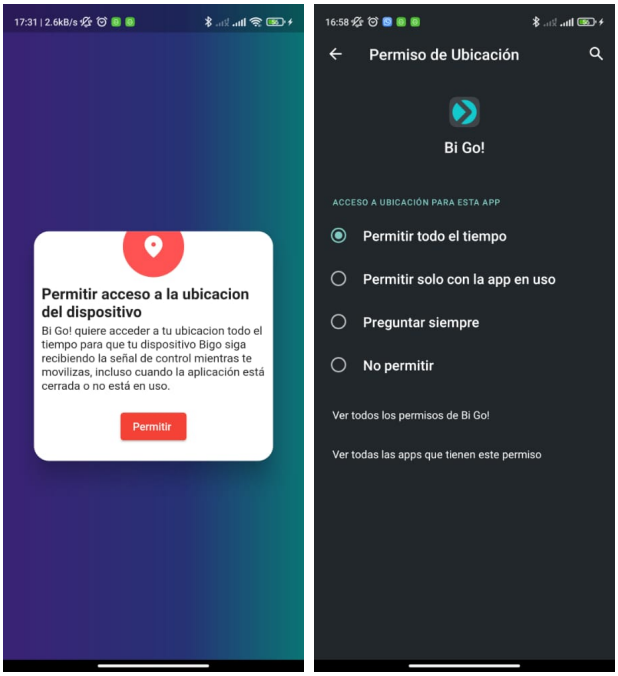


Figura 6.14: Pantallas Permisos
Fuente:Elaboración propia

Esta pantalla está compuesta por dos widgets:

- Fondo
  - Imagen: corresponde a imagen de bienvenida con logo de app
- Ventana emergente
  - Texto informativo sobre permisos que usuario debe aceptar

- Botón: aceptar permisos requeridos por App

```
        void alertUbicacion(BuildContext context) {
    final alert3 = Dialog(
        shape: RoundedRectangleBorder( borderRadius: BorderRadius.all(Radius.circular(20.0))),
        child: Stack(alignment: Alignment.topCenter,
          children: [ Container(
              height: 300,
              child: Padding( padding: const EdgeInsets.fromLTRB(10, 70, 10, 10),
                child: Column(
                  children: [
                    Text('Permitir acceso a la ubicacion del dispositivo',
                      style: TextStyle(fontWeight: FontWeight.bold, fontSize: 20),
                    ),
                    SizedBox(height: 5),
                    Text(
                      'Bi Go! quiere acceder a tu ubicacion todo el tiempo para que tu dispositivo Bigo
                       siga recibiendo la señal de control mientras te movilizas, incluso cuando la
                       aplicación está cerrada o no está en uso.',
                      style: TextStyle(fontSize: 15),
                    ),
                    SizedBox( height: 20,  ),
                    ElevatedButton(
                        child: Text("Permitir "),
                        style: ElevatedButton.styleFrom(primary: Colors.red),
                        onPressed: () {... //   onPressed
                        }),
                  ],
                ),
              ),
            ),
            Positioned(  top: -20,
                child: CircleAvatar(
                  backgroundColor: Colors.redAccent,
                  radius: 40,
                  child: Icon(
                    Icons.location_on_rounded,
                   color: Colors.white,
                    size: 35,
                  ), )),
          ], ));
    showDialog(
        barrierDismissible: false,
        context: context,
        builder: (BuildContext context) => alert3);
  }
```

Figura 6.15: Codigo Permisos
Fuente:Elaboración propia

6.5.2. Pantalla Rutas

En esta pantalla se puede visualizar unas opciones que maneja la aplicación.

Esta pantalla está compuesta por 3 widgets:

- Barra superior.
  - Texto: nombre de la pantalla.
- Una lista formada por widgets que muestran 3 opciones a elegir.
  - Texto: nombre de la acción.
  - Iconos relacionados a la acción.
- Barra de navegación inferior
  - Varios elementos en forma de etiquetas de texto proporcionan una navegación rápida entre las pantallas principales

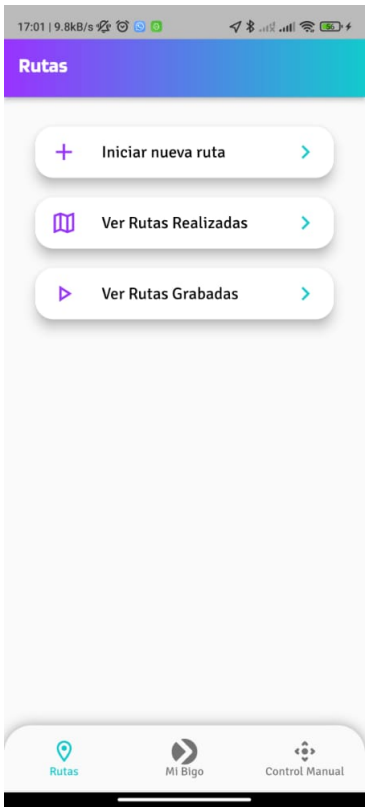


Figura 6.16: Pantallas Rutas
Fuente:Elaboración propia

- Opción 1 Iniciar Ruta: En esta opción no redirige a pantalla mapa en la cual podremos interactuar
- Opción 2 Ver Rutas Guardadas: Esta Opción nos mostrara una pantalla con un listado de las rutas personalizadas que fueron guardadas.

```
        Widget build(BuildContext context) {
    return Consumer2<MapsDirectionsState, RemoteDeviceState>(
        builder: (context, appState, bleState, w) {
      List<String> litems = [
        "Iniciar nueva ruta",
        "Ver Rutas Grabadas"
      ];
      List<Icon> licons = [
        Icon(Icons.add, color: HexColor("#9933FF"),  size: 30, ),
        Icon( Icons.map_outlined, color: HexColor("#9933FF"),  size: 30,    ),
        Icon(
          Icons.play_arrow_outlined,
          color: HexColor("#9933FF"),
          size: 30,
        ),   ];
      return SafeArea(
          child: Scaffold(
              appBar: Utils().appbar('Rutas'),
              body: new ListView.builder(
                  padding: const EdgeInsets.symmetric(
                      vertical: 21.5, horizontal: 30),
                  itemCount: litems.length,
                  itemBuilder: (BuildContext context, int index) {
                    return Padding(
                      padding: EdgeInsets.symmetric(vertical: 6.5),
                      child: Card(
                        shape: RoundedRectangleBorder(
                            borderRadius: BorderRadius.circular(20)),
                        elevation: 10,
                        child: ListTile(
                          title: Text(
                            litems[index],
                            style: signikarutalistTextStyle(),
                          ),
                          leading: licons[index],
                          trailing: Icon(
                            Icons.chevron_right,
                            color: HexColor("#13CACE"),
                            size: 30,
                          ),
                          onTap: () { .... //onTap
                          },
                        ),),);
                  })));
    });
  }
```

Figura 6.17: Codigo Opciones Rutas
Fuente:Elaboración propia

6.5.3. Pantalla Dispositivo

En esta pantalla se hace la respectiva conexión bluetooth con dispositivo indicador.

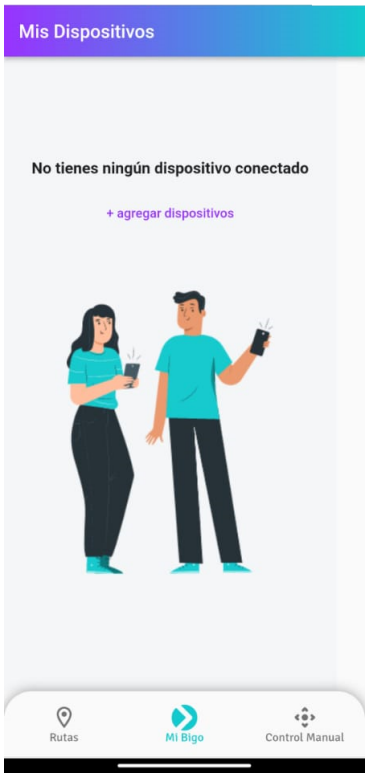


Figura 6.18: Pantallas Dispositivos
Fuente:Elaboración propia

6.5.4. Pantalla Control Manual

En esta pantalla fue diseñada para utilizar unos botones y este de forma manual dar indicación al dispositivo indicador.

Esta pantalla está compuesta por 4 widgets:

- Barra superior.
  - Texto: nombre de la pantalla.
- Botón para iniciar Control Manual.
  - Acción: Inicia el control manual y despliega otro widget con botones

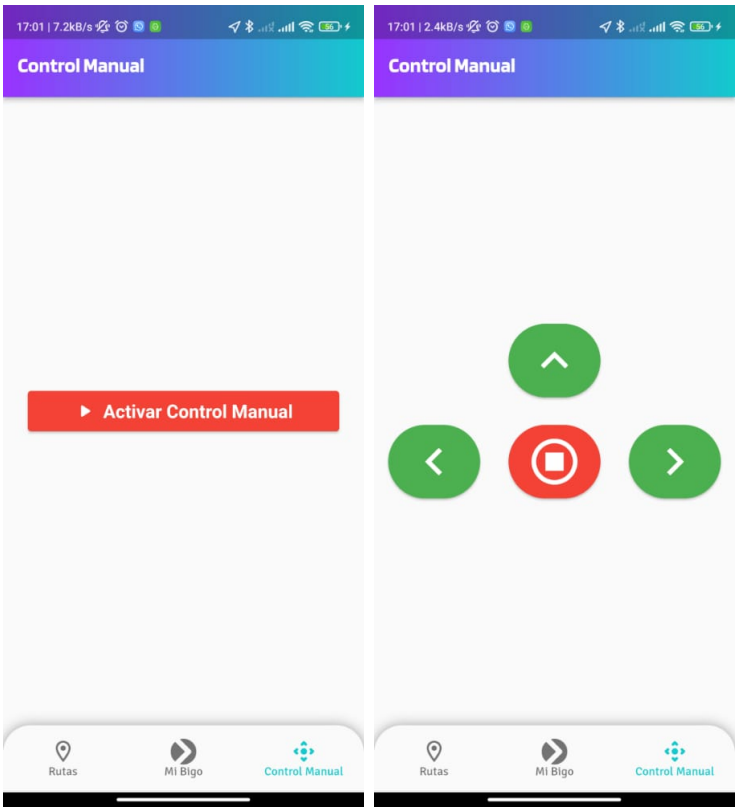


Figura 6.19: Pantallas Control Manual
Fuente:Elaboración propia

- Barra de navegación inferior
  - Varios elementos en forma de etiquetas de texto proporcionan una navegación rápida entre las pantallas principales
- Botones de control manual
  - Botón: se define un tamaño fijo y se duplica en 4 botones cada uno con sus respectivos características como iconos color y función.

```
        Widget build(BuildContext context) {
    return Stack(
      children: <Widget>[
        _modo == mode.INICIOCONTROL
            ? Positioned(
                child: Column(
                    mainAxisAlignment: MainAxisAlignment.center,
                    children: [
                      Row(
                        mainAxisAlignment: MainAxisAlignment.center,
                        children: [
                          ElevatedButton.icon(
                            onPressed: () {.... //onPressed
                                    },
                            icon: const Icon(Icons.play_arrow_rounded),
                            style: ElevatedButton.styleFrom(
                                primary: Colors.red,
                                padding: const EdgeInsets.symmetric(
                                    horizontal: 50, vertical: 10),
                                textStyle: const TextStyle(
                                    fontSize: 20, fontWeight: FontWeight.bold)),
                            label: Text('Activar Control Manual'),
                          ), ], ), ]), )
            : null,
        _modo == mode.MANDO
            ? Positioned(
                child: Container(
                  child: Column( mainAxisAlignment: MainAxisAlignment.center,
                    children: [
                      Row( mainAxisAlignment: MainAxisAlignment.center,
                        children: [
                          ElevatedButton(
                            onPressed: () {...../onPressed Indicacion Adelante
                                    },
                            child: Icon( Icons.keyboard_arrow_up, size: 60,
                            ),
                            style: ElevatedButton.styleFrom(
                                fixedSize: const Size(100, 80),
                                primary: _flag1 ? Colors.green : Colors.blue,
                                shape: RoundedRectangleBorder(
                                    borderRadius: BorderRadius.circular(50))),
                          ) ],   ),
                      SizedBox( height: 30,),
                      Row(  mainAxisAlignment: MainAxisAlignment.spaceAround,
                        children: [
                          ElevatedButton(
                            onPressed: () {....// onPressed Indicacion izquerda
                            },
                            child: Icon(Icons.keyboard_arrow_left, size: 60,
                            ),
                            style: ElevatedButton.styleFrom(
                                fixedSize: const Size(100, 80),
                                primary: _flag2 ? Colors.green : Colors.blue,
                                shape: RoundedRectangleBorder(
                                    borderRadius: BorderRadius.circular(50))),
                          ),
                          ElevatedButton(
                            onPressed: () { ... //onPressed indicacion Stop

                            },
                            child: Icon( Icons.stop_circle_outlined,  size: 60,
                            ),
                            style: ElevatedButton.styleFrom(
                                fixedSize: const Size(100, 80),
                                primary: _flag3 ? Colors.red : Colors.blue,
                                shape: RoundedRectangleBorder(
                                    borderRadius: BorderRadius.circular(50))),
                          ),
                          ElevatedButton(
                            onPressed: () {.... // onPressed Indicacion Derecha
                                    },
                            child: Icon(Icons.keyboard_arrow_right, size: 60,
                            ),
                            style: ElevatedButton.styleFrom(
                                fixedSize: const Size(100, 80),
                                primary: _flag4 ? Colors.green : Colors.blue,
                                shape: RoundedRectangleBorder(
                                    borderRadius: BorderRadius.circular(50))),
                          ), ], ) ],), ),)
            : null,
      ].where((element) => element != null).toList(),
    );
  }
```

Figura 6.20: Código Control Manual
Fuente:Elaboración propia

6.5.5. Pantalla Iniciar Ruta

En esta pantalla el usuario podrá ver su ubicación actual y interactuar para dar su destino. Como puede verse en las figuras, la pantalla cuenta con la vista de un mapa navegable. Puede verse la ubicación actual y un widget inicial para ingresar destino.
Esta pantalla está compuesta por 7 widgets y se utiliza widget suministrado por el SDK para hacer una superposición de widgets y asi permitiendo colocar varias capas de widgets en la pantalla.

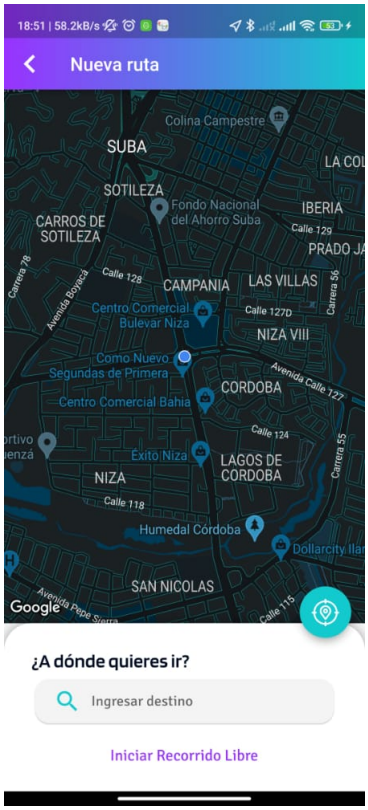


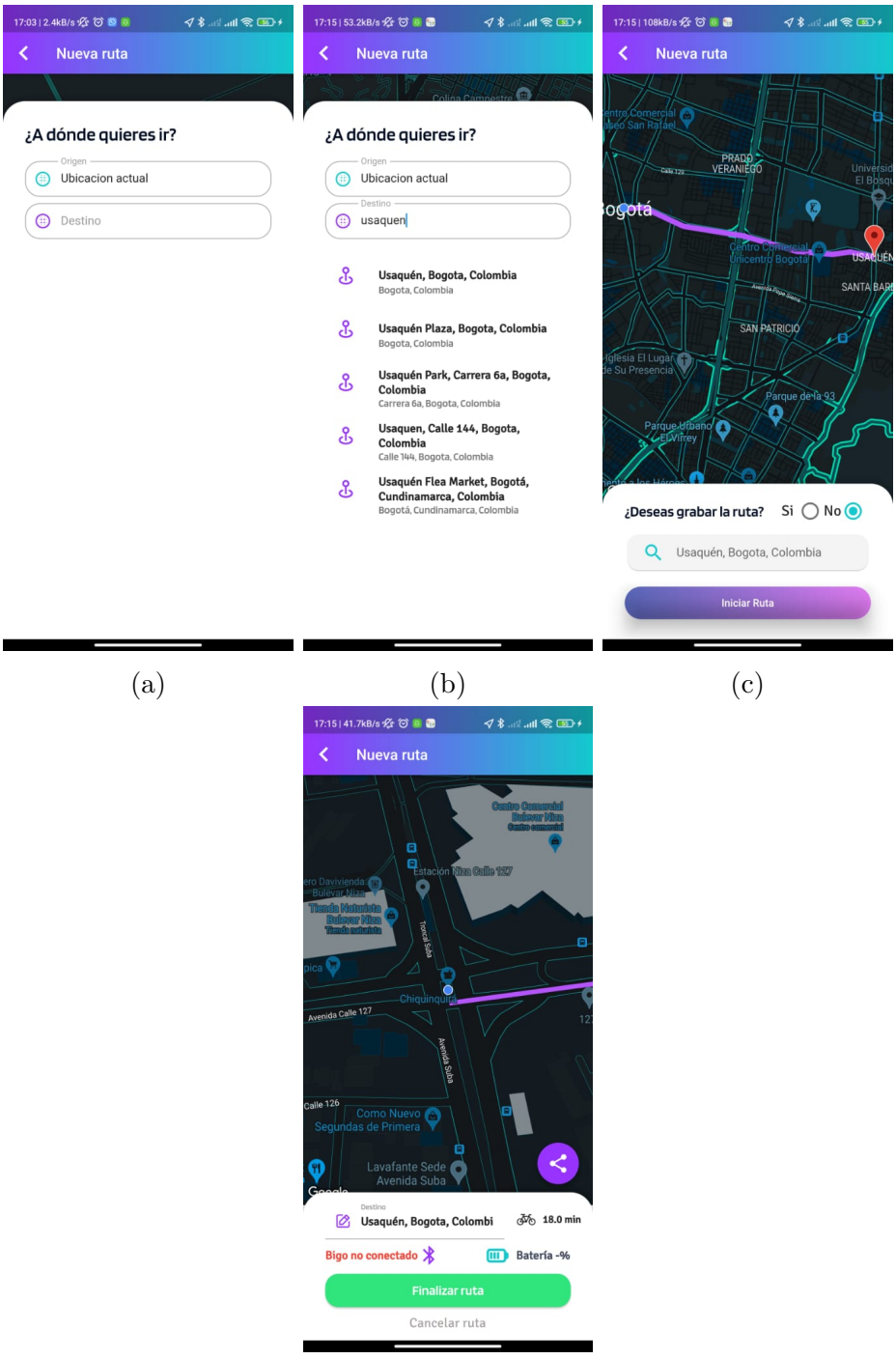
Figura 6.21: Pantallas Iniciar Ruta
Fuente:Elaboración propia

- Barra superior.
  - Texto: nombre de la pantalla.
  - Botón: volver a pantalla principal
- Un widget de mapa.
  - Contiene puntero de ubicación actual del usuario en el mapa.

- Contenedor de opción de ruta a realizar
  - Texto: pregunta al usuario.
  - Botón Opción 1 (Figura 6.23): Despliega abrir otro widget donde puedes ingresar lugar destino.

```
mapStatus == MapStatus.IDLE
      ? Stack(children: <Widget>[
          Positioned(   bottom: 0,    left: 0,  right: 0,
              child: Container(
                child: Container(
                    decoration: BoxDecoration(.... //BoxDecoration
                        ),
                  ],
                )),
                height: 183,
                decoration: BoxDecoration(... //BoxDecoration
                    ),
              )),
            Positioned(bottom: 70.0,     right: 0,    left: 0,
              child: Container(
                padding: const EdgeInsets.symmetric(horizontal: 30),
                child: Card(
                  shape: RoundedRectangleBorder( borderRadius: BorderRadius.circular(15)),
                  color: HexColor('#F4F4F4'),
                  child: TextField(
                    onTap: () async {
                                  });
                    },
                    cursorColor: Colors.black,
                    textInputAction: TextInputAction.go,
                    readOnly: true,
                    decoration: InputDecoration(
                      icon: Container(
                        margin: EdgeInsets.only(left: 20, top: 0),
                        width: 10,
                        child: Icon( Icons.search,size: 30, color: HexColor("#13CACE"),
                        ),
                      ),
                      hintText: "Ingresar destino",
                      hintStyle: TextStyle( fontFamily: 'Signika', fontSize: 16),
                      border: InputBorder.none,
                      contentPadding: EdgeInsets.only(left: 15.0, top: 0.0),
                    ), ),), ),),
              Positioned(  bottom: 15,left: 0,  right: 0,
                child: TextButton(
                    onPressed: () {... // onPressed
                                    },
                    child: Text(
                      'Iniciar Recorrido Libre',
                      style: TextStyle(
                          color: HexColor('#9933FF'),... // TextStyle
                          ),
                    ))),
            Positioned(  bottom: 130, left: 30,  right: 0,
                child: Text(
                  '¿A dónde quieres ir? ',
                  style: TextStyle(
                      color: HexColor('#000B22'),....
                    ),   )),
            Positioned(    bottom: 167,    right: 15,
              child: FloatingActionButton(
                heroTag: "my_location",
                backgroundColor: HexColor('#13CACE'),
                onPressed: () { ... // onPressed
                },
                child: Icon(
                  Locationicon.locationdi,  ...
                  ), ),  )  ])
```

Figura 6.22: Código Iniciar Ruta
Fuente:Elaboración propia



(a) (b) (c)

(d)

Figura 6.23: Pantallas Iniciar Ruta Opción 1
Fuente:Elaboración propia

a) Widget donde el usuario puede ingresar como texto su destino

b) Lista desplázale que muestra predicciones de lugares según búsqueda realizadas.

```
mapStatus == MapStatus.BUSQUEDA
    ? Stack(children: <Widget>[
        Positioned( top: 37, left: 0,  right: 0, bottom: 0,
            child: Container(
              child: Container(
                  decoration: BoxDecoration(...//BoxDecoration
             ),
              height: 183,
              decoration: BoxDecoration( ...//BoxDecoration
            )),
        Positioned(  top: 66,  left: 30,  right: 0,
            child: Text(
              '¿A dónde quieres ir? ',
              style: TextStyle(... //TextStyle
            )),
        Positioned(  top: 118,  right: 0,  left: 0,
          child: Container(
            padding: const EdgeInsets.symmetric(horizontal: 30),
            child: TextFormField(
              onTap: () async {},
              cursorColor: Colors.black,
              controller:
                  TextEditingController(text: 'Ubicacion actual'),
              textInputAction: TextInputAction.go,
              readOnly: true,
              decoration: InputDecoration(
                prefixIcon: Icon(
                  Icons.blur_circular_outlined,
                  color: HexColor('#13CACE'),
                ),
                border: OutlineInputBorder(
                  borderRadius: BorderRadius.circular(20.0),
                ),
                focusedBorder: OutlineInputBorder(
                  borderSide: const BorderSide(..../BorderSide
                ),
                contentPadding:  EdgeInsets.only(left: 15.0, top: 0.0),
                labelText: 'Origen',
                labelStyle: TextStyle(... //  TextStyle
                ),
              ),), ),   ),

        Positioned( top: 175, right: 0,  left: 0,
          child: GestureDetector(
            onTap: () {FocusScope.of(context).requestFocus(new FocusNode());
            },
            child: Container( padding: const EdgeInsets.symmetric(horizontal: 30),
              child: TextFormField(
                controller: appState.destinationController,
                textInputAction: TextInputAction.search,
                onChanged: (places) {... //    onChanged
                },
                  decoration: InputDecoration(
                  prefixIcon: Icon(  Icons.blur_circular_outlined,
                    color: HexColor('#9933FA'),  ),
                  hintText: "Ingresar destino",
                  border: OutlineInputBorder(
                    borderRadius: BorderRadius.circular(20.0),),
                  focusedBorder: OutlineInputBorder(....// OutlineInputBorder
                                                      ),
                  contentPadding:
                      EdgeInsets.only(left: 15.0, top: 0.0),
                  labelText: 'Destino',
                  labelStyle: TextStyle(
                    color: Colors.grey,
                    fontSize: 16,
                    fontFamily: "verdana_regular",
                    fontWeight: FontWeight.w400,
                  ),
                ),),),),),),
        _searching
            ? Positioned( top: 245, left: 30, right: 30,    bottom: 0,
                child: Container(
                  child: Container(
                      child: findAutoComplete(
                          context, appState, busqueda)),
                  decoration: BoxDecoration(...//BoxDecoration
            )
            : Center(
              child: Text(''),
              )
      ])
    : null,
```

Figura 6.24: Código Widget Búsqueda
Fuente:Elaboración propia

c) Al seleccionar una opción de la lista aparece otro widget con una confir-

mación de ruta para el usuario

```
mapStatus == MapStatus.SHOWING_ROUTE_PREVIEW
     ? Stack(children: <Widget>[
         Positioned(bottom: 0,  left: 0,  right: 0,
             child: Container(
               child: Container(
                   decoration: BoxDecoration(...//BoxDecoration
           ),
               height: 210,
               decoration: BoxDecoration( ...//BoxDecoration),
         Positioned( bottom: 161, left: 30, right: 0,
             child: Text( '¿Deseas grabar la ruta? ',
               style: TextStyle(...//TextStyle
           )),
         Positioned( bottom: 149,  right: 30,
           child: Row( children: <Widget>[
               Transform.scale(scale: 1.3,
                 child: Radio(
                   value: 1,
                   groupValue: id,
                   onChanged: (val) {   ...//  onChanged
                               },
                 ),
               ),
               SizedBox( width: 10,),
               Transform.scale(  scale: 1.3,
                 child: Radio(  value: 2,
                   activeColor: HexColor('#13CACE'),
                   groupValue: id,
                   onChanged: (val) { ...//  onChanged
                   },
               ), ),  ],  ), ),
         Positioned( bottom: 161, right: 70,
             child: Row(
             children: <Widget>[
               Text('Si',
                 style: new TextStyle(fontSize: 20.0, fontFamily: 'Signika'),),
               SizedBox(width: 40),
               Text( 'No',
                 style: new TextStyle(
                     fontSize: 20.0, fontFamily: 'Signika'),
               ),
             ],), ),
         Positioned( bottom: 89.0, right: 0,left: 0,
           child: Container(
             padding: const EdgeInsets.symmetric(horizontal: 30),
             child: Card(
               shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(15)),
               color: HexColor('#F4F4F4'),
               child: TextField(
                 textInputAction: TextInputAction.go,
                 readOnly: true,
                 decoration: InputDecoration(
                   icon: Container(
                     margin: EdgeInsets.only(left: 20, top: 0),
                     child: Icon( Icons.search, color: HexColor("#13CACE"),
                       size: 30,
                     ), ),
                   hintText: appState.destinationController.text,
                   border: InputBorder.none,
                 ),
               ), ),),),
         Positioned( bottom: 27, left: 30, right: 30,
             child: Container( width: 500, height: 44,
                 decoration: BoxDecoration(
                   gradient: const LinearGradient(colors: [
                     Color(0xFF5C6BC0),  Color(0xFFEA80FC),
                   ]),
                   borderRadius: BorderRadius.circular(25),
                 ),
                 child: ElevatedButton( style: ButtonStyle(
                       backgroundColor: MaterialStateProperty.all(
                           Colors.transparent),
                       elevation: MaterialStateProperty.all(20.0),
                     ),
                     onPressed: () async {   ..../// onPressed
                     },
                     child: const Text(
                       "Iniciar Ruta",
                       textAlign: TextAlign.center,
                       style: TextStyle(
                         color: Colors.white,
                       ),
                     )))),
       ])
     : null,
```

Figura 6.25: Codigo Previsualizacion Ruta
Fuente:Elaboración propia

d) Widget contenedor con información: texto nombre de destino, tiempo

aproximado de recorrido, conexión de dispositivo y porcentaje batería dispositivo.
A la vez este widget contiene botón para finalizar la ruta y para cancelar si es necesario.

```
mapStatus == MapStatus.ON_ROUTE
      ? Stack(children: <Widget>[
          Positioned( bottom: 0, left: 0,  right: 0,
              child: Container(
                child: Container(
                    decoration: BoxDecoration(.... //BoxDecoration
                ),
                  height: 200,
                  decoration: BoxDecoration(.... //BoxDecoration
                    ),
          Positioned( bottom: 137.0, right: 120, left: 30,
            child: Container(
              child: TextFormField(
                onTap: () async {},
                controller: opcionRuta == 0
                    ? appState.destinationController
                    : TextEditingController(text: nameSaveRuta.name),
                style: TextStyle(....//TextStyle
                ),
                textInputAction: TextInputAction.search,
                decoration: InputDecoration(
                  prefixIcon: Icon( Onrute.icondirec,
                    color: HexColor('#9933FA'),
                    size: 18,  ),
                  hintText: "Ingresar destino",
                  contentPadding:
                      EdgeInsets.only(left: 15.0, top: 0.0),
                  labelText: 'Destino',
                  labelStyle: TextStyle(...//TextStyle
                                          ),
                ),),  ),  ),
          Positioned(bottom: 155.0, right: 15, left: 250,
              child: Row(  mainAxisAlignment: MainAxisAlignment.end,
                children: [
                  Icon(  Onrute.bici,  size: 15,  ),
                  SizedBox( width: 20, ),
                  Text( '${appState.tempRoute} min',
                    style: TextStyle(...//TextStyle
                                        ),
                ], )),
          Positioned(bottom: 45, left: 30, right: 30,
              child: Container(
                height: 45,
                child: ElevatedButton(
                    style: ButtonStyle(....//ButtonStyle
                                      )),
                      backgroundColor: MaterialStateProperty.all(
                          HexColor('#2FE57C')), ),
                    onPressed: () async {...//onPressed
                                          },
                    child: const Text(
                      "Finalizar ruta",
                      textAlign: TextAlign.center,
                      style: TextStyle(
                        color: Colors.white,
                        fontFamily: 'Blinkersemibold',
                        fontSize: 18,
                      ),
                    )),    )),
```

Figura 6.26: Codigo Widget Información en Ruta
Fuente:Elaboración propia

```
Positioned( bottom: 100, left: 30,  right: 30,
         child: Row(
           mainAxisAlignment: MainAxisAlignment.spaceBetween,
           crossAxisAlignment: CrossAxisAlignment.start,
           children: [
             Row(  mainAxisAlignment: MainAxisAlignment.spaceBetween,
               children: [
                 conexionpreviewroute(bleState.bluetoothStateStr,
                     context, bleState.measurerCargoInfo),
                 Padding(
                   padding: const EdgeInsets.only(right: 10),
                   child: Icon(Icons.bluetooth,
                     color: HexColor("#9933FF"),
                     size: 30,
                   ),
                 ), ],  ),
             Row( mainAxisAlignment: MainAxisAlignment.spaceBetween,
               children: [  Padding(
                   padding: const EdgeInsets.only(right: 10),
                   child: Icon( const IconData(0xe800,
                         fontFamily: 'MyFlutterApp',
                         fontPackage: null),
                     color: HexColor("#13CACE"),
                     size: 30,
                   ), ),
                 baterypreviewroute(bleState.bluetoothStateStr,
                     context, bleState.measurerCargoInfo)
               ],),],),),
       Positioned( bottom: 211, right: 19,
         child: FloatingActionButton(
           heroTag: "compartir",
           backgroundColor: HexColor('#9933FF'),
           onPressed: () { },
           child: Icon( Icons.share, color: Colors.white,
             size: 30,  ),
         ), ),
       Positioned( bottom: 0, left: 30,right: 30,
           child: TextButton(
               onPressed: () {..../onPressed
               },
               child: Text(
                 'Cancelar ruta',
                 style: TextStyle(..../TextStyle
                         )))
               ])
   : null,
```

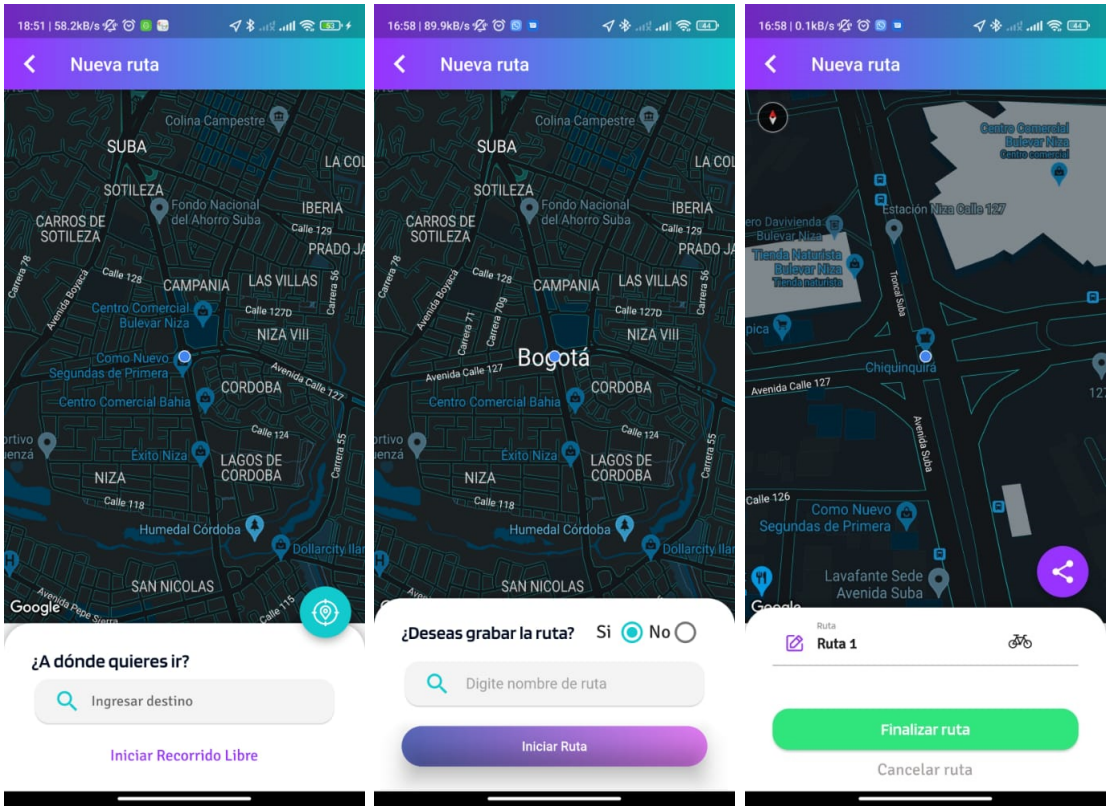
Figura 6.27: Codigo Widget Información en Ruta
Fuente:Elaboración propia

- Botón Opción 2 (Figura 6.27): Despliega otro widget y permite que el usuario graba una ruta libre que desee realizar.

  a) Campo de texto para dar nombre a ruta personalizada

  b) Widget de confirmación de inicio de ruta.

  c) al iniciar ruta nos despliega widget con botones de finalización de ruta para cuando el usuario finalice la ruta personalizada o también esta el botón para cancelar dicha ruta.



(a) (b) (c)

Figura 6.28: Pantallas Iniciar Ruta Opción 2
Fuente:Elaboración propia

7 Resultados

Como resultado de todo el anterior desarrollo se obtienen la aplicacion para sistema operativo Android en formato APK el cual a sido llamadas BIGO!, fácil de instalar y usar, simplemente se debe descargar en el dispositivo móvil, después se debe pulsar en instalar para luego abrirla y usarla.

En base a requerimientos funcionales y casos de uso, se revela el cumplimiento de 4 principios básicos de diseño y funcionalidad, los cuales son:

7.1. Buscar Ruta

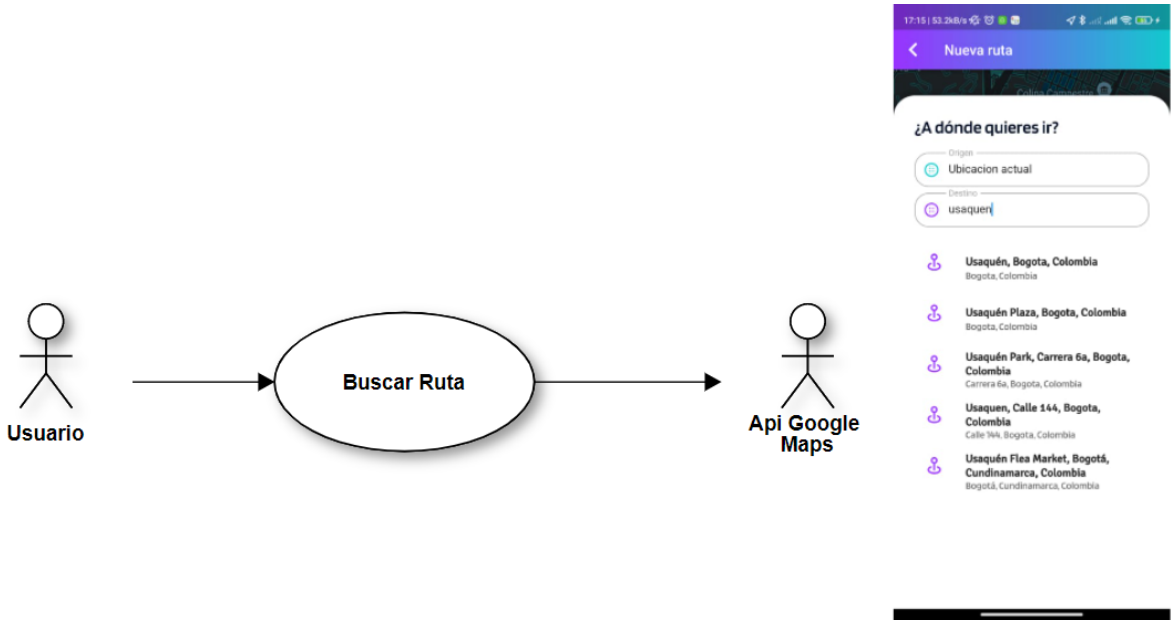


Figura 7.1: Resultado caso de uso (Buscar Ruta)
Fuente:Elaboración propia

Se muestra el correcto funcionamiento de busqueda y predicción de resultado destinos que el usuario dígito.

7.2. Ver Ruta

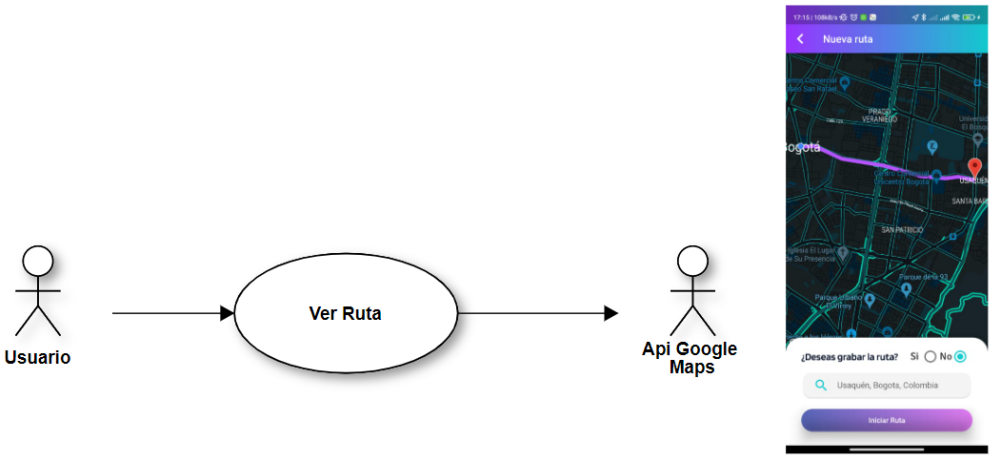


Figura 7.2: Resultado caso de uso (Ver Ruta)
Fuente:Elaboración propia

Para esta prueba se muestra la interfaz Mapa, permitiendo ver una ruta trazada a su destino en función de la distancia más corta, del origen y de llegada.

7.3. Iniciar Recorrido libre y guardar ruta

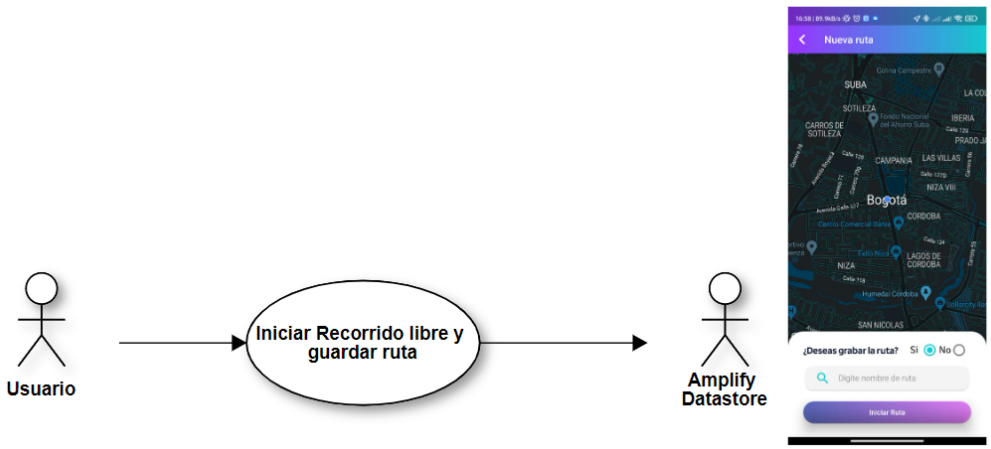


Figura 7.3: Resultado caso de uso (Iniciar Recorrido libre y guardar ruta)
Fuente:Elaboración propia

Para el resultado de esta prueba se dio a conocer de forma satisfactoria su correcto funcionamiento al momento de Iniciar un recorrido personal de usuario y poder guardarlo para luego Utilizarlo.

7.4. Ver Rutas Guardadas

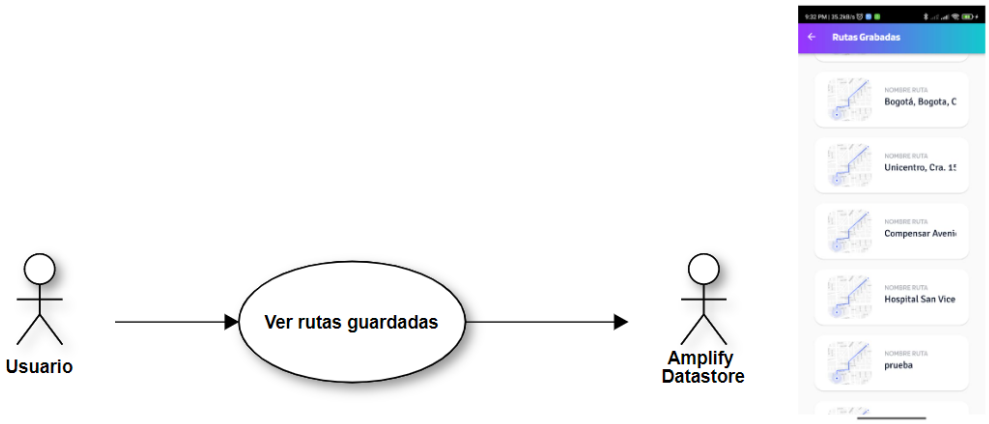


Figura 7.4: Resultado caso de uso (Ver rutas guardadas)
Fuente:Elaboración propia

Por último se realiza la prueba de ver rutas guardadas los cuales se muestran en forma de lista permitiendo editar, eliminar y visualizar la información de cada ruta y su vez utilizar la rutas.

7.5. Pruebas Con Dispositivo de Empresa

Las Pruebas realizadas con dispositivo de empresa (Bigo) corresponde a indicaciones de direcion enviadas desde la aplicación a este dispositivo, siendo este dispositivo un indicador con matriz led el cual nos permite visualizar si el usuario va a girar en algún sentido (Izquierda o Derecha).
A continuación se dara a conocer el dispositvo de la empresa (Bigo).

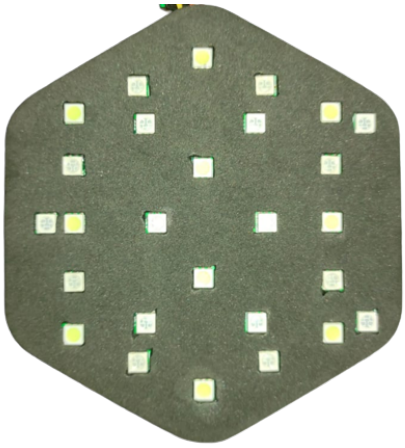
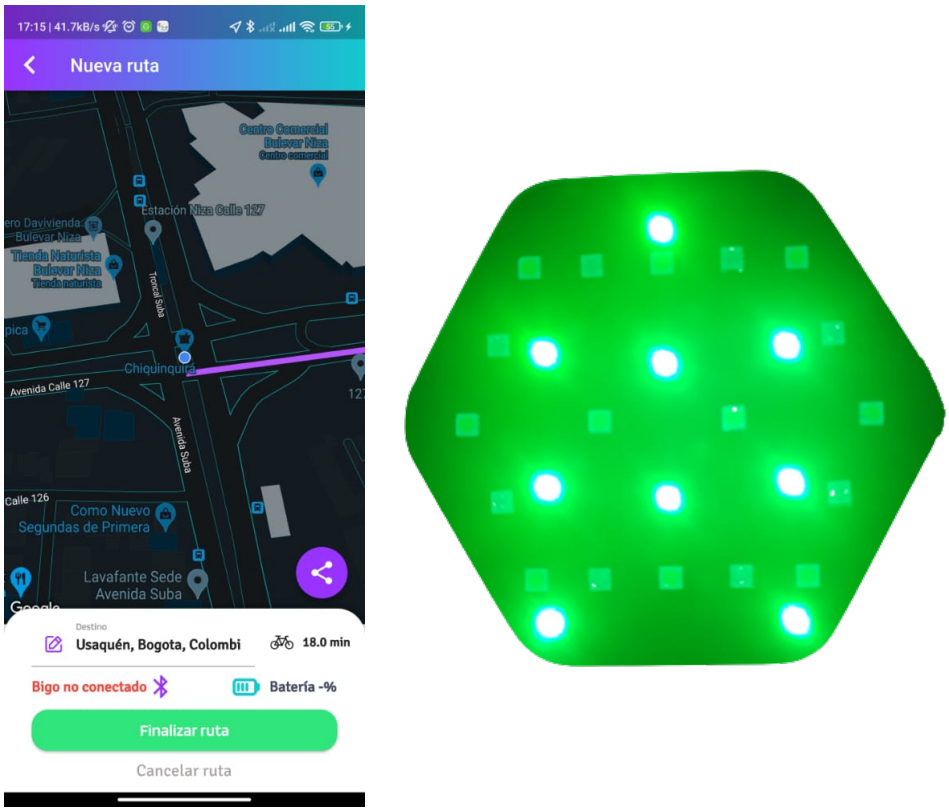


Figura 7.5: Dispositivo (Bigo)
Fuente:Elaboración propia

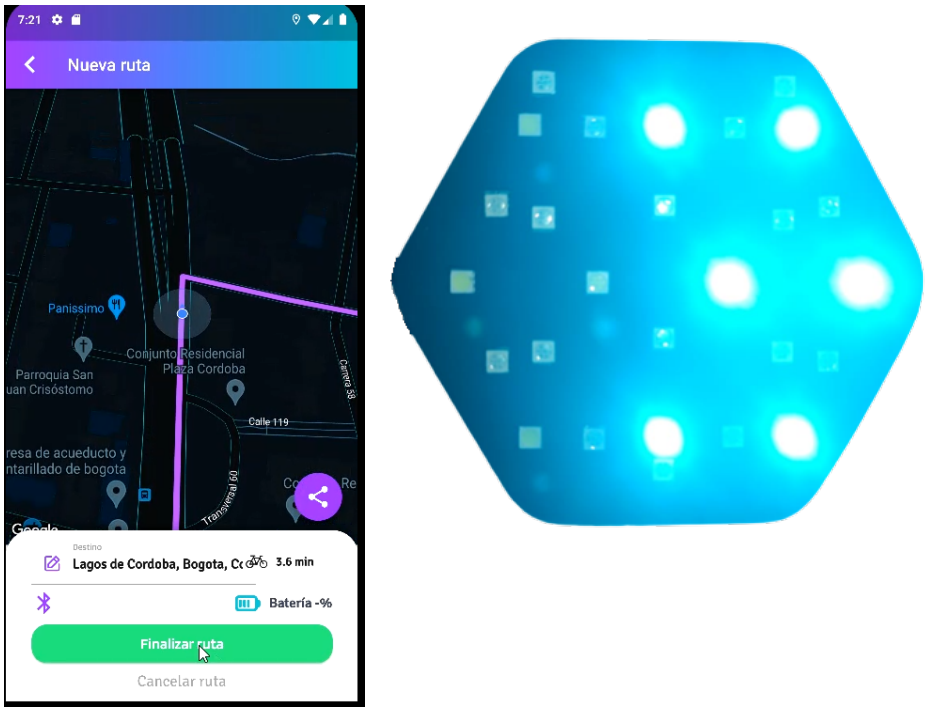
Cuando hemos iniciado la aplicación esta se tiene que conectar con el dispositivo via bluetooth. Una vez conectado el usuario procede a iniciar una ruta y la aplicación dara indicación al dispositivo que el usuario avanza (adelante).
En la siguiente figura se muestra el estado de la aplicación al inciar ruta y la indicación mostrada en el dispositivo.



(a) (b)

Figura 7.6: Aplicacion y Dispotivo Iniciando Ruta
Fuente:Elaboración propia

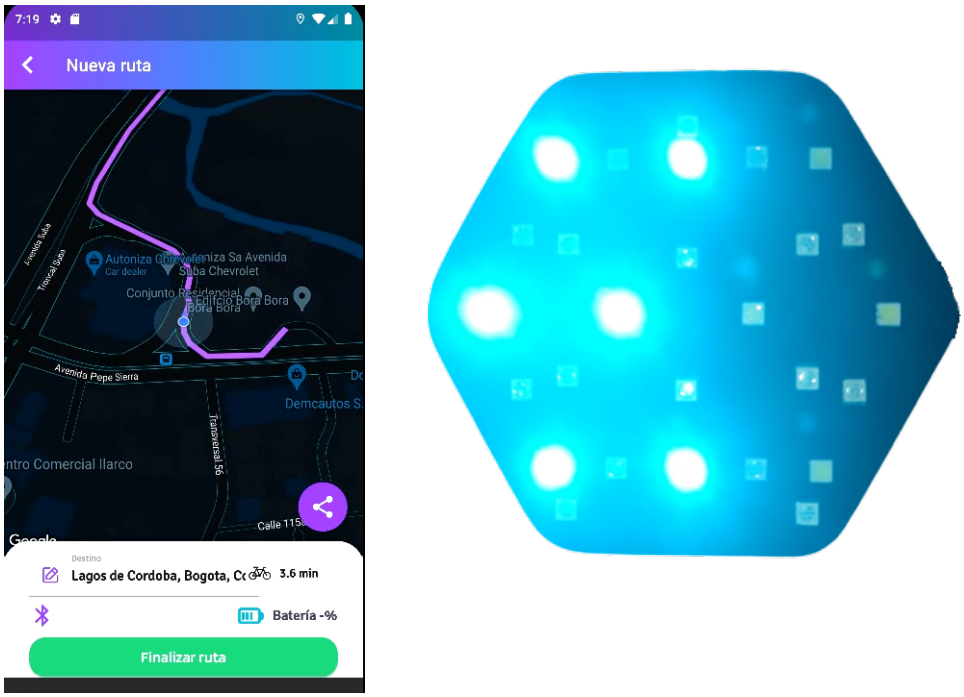
7.5.1. Indicacion Giro Derecha



(a) (b)

Figura 7.7: Aplicacion y Dispotivo Girando Derecha
Fuente:Elaboración propia

7.5.2. Indicacion Giro izquierda



(a) (b)

Figura 7.8: Aplicacion y Dispotivo Girando izquierda
Fuente:Elaboración propia

8 Conclusiones

Iniciar en este proyecto era entrar un campo desconocido para mi en como crear aplicaciones móviles. La oportunidad para trabajar en un proyecto práctico me motivó a hacerlo y tomarlo como un reto.
He adquirido muchos conocimientos sobre ingeniería de software. Tuve reuniones donde se formalizaron requerimientos, e implementado diseños, lo que me permitieron entender mejor la importancia de todos estos pasos y la unificación de técnicas y procesos que permitan llevarlos a cabo.
Por otro lado, también decidí aprovechar esta oportunidad para trabajar en un nuevo campo. aprender nuevas tecnologías. Desde idiomas hasta editores de texto, todo las herramientas utilizadas son todas nuevas para mí y siendo además también Flutter muy reciente. Gracias a eso, aprendí los conceptos básicos de todas estas tecnologías y ampliar mi currículum.

Con esto queda demostrada la importancia del aprendizaje como primer objetivo, realizar una preparación previa con cursos y capacitaciones genera una preparación importante como desarrollador, así mismo conocer la lógica de negocio de la empresa y el entorno tecnológico que se llevaba hasta ese momento da mucha más claridad al momento previo de formar parte del equipo de desarrollo de manera técnica. La capacitación constante conllevó al buen desarrollo de las plataformas requeridas y a satisfactorias entregas de las mismas.

El objetivo principal del proyecto es desarrollar una aplicación para el sistema operativo Android. En concreto, se trata de desarrollar una aplicación para ciclistas basada en geolocalización, cuyas características era que el ciclista tuviera una ruta y mediante un dispositivo indicador poder dar a conocer a los demás la maniobra a realizar.

Bibliografía

[1] Bigo. En: *Recuperado de https://www.bigosafe.com*

[2] Documentation for Visual Studio Code. En: *Recuperado de:https://code.visualstudio.com/docs*

[3] Google Maps Platform Documentación. En: *Recuperado de https://developers.google.com/maps/documentation?hl=es* (2021)

[4] Bennett, Jim: Xamarin in Action. En: *Manning Publications* (2018)

[5] Bhandari, Pragati: The Application Deployment Process Using AWS. (2021)

[6] Clavijo, Manuel Eduardo Patarroyo Santos Juan Pablo C.: Desarrollo tecnológico de un chaleco o chaqueta inteligente que permite a motociclistas ser más perceptibles en las vías. En: *Universidad piloto de Colombia*

[7] Cloud, Clinic: En: *Obtenido de Clinic Cloud: https://clinic-cloud.com/blog/servicios-enla-nube-tipos-ejemplo* (2017)

[8] Date, C. J.: An Introduction to Database Systems. En: *Addison Wesley Publishing Company* (1977)

[9] De Miguel A., Piattini M.: Fundamentos y Modelos de Bases de Datos. En: *.2ª ed., Alfaomega-Ra-ma, México* (2004)

[10] documentation, Flutter: Introduction to widgets. En: *Recuperado de:https://flutter.dev/docs/development/ui/widgets-intro* (2019)

[11] Eisenman, Bonnie: Learning React Native: Building Native Mobile Apps with JavaScript. En: *2nd ed. O'Reilly,* (2017)

[12] Erick Gamma, Ralph J.: Libro Patrones de diseño, Elementos de software orientado a objetos reutilizable. En: *PEARSON EDUCACIONS.A. Madrid* (2003)

[13] Flutter.dev: Flutter. En: *Recuperado de https://flutter.dev/*

[14] Flórez, H.: Programación orientada a objetos, usando Java. (2014)

[15] GEOTAB: GEOTAB. En: *Obtenido de GEOTAB: https://www.geotab.com/eslatam/blog/qu%C3%A9-significa-gps/* (2020)

[16] Impulsa: BiSafe. En: *Recuperado de:https://innpulsacolombia.com/milab/solucionadores/cinnov-sas* (2020)

[17] J, Gamma E. Helm R. Johnson R. V.: Design Patterns. Elements of Reusable Object-Oriented Software. En: *AddisonWesley.Pearson Educación* (1995)

[18] Johan Asdrúbal Parrado Herrera-Jorge Humberto Piedrahita, Camilo Antonio Suárez B.: PROYECTO LYRO: LIGHT ON THE ROAD. En: *Universidad Nacional de Colombia*

[19] Karen Vanessa Rosillo Granados, Paula Andrea Celis Méndez Daniella Rodríguez U.: Diseño de aplicación tecnológica para el reporte de incidentes con vehículos no motorizados; Bicifor. En: *Universidad Piloto de Colombia*

[20] Morero, Francisco: Introducción a la OOP. En: *Grupo EIDOS Ejemplar gratuito* (1999)

[21] Pezoa, et a.: Foundations of JSON schema. En: *Proceedings of the 25th International Conference on World Wide Web. 2016.*

[22] play, Google: The Secrets to App Success on Google Play. En: *Recuperado de:https://commondatastorage.googleapis.com/androiddevelopers/shareables/distribute/secrets_play/v2/web/secrets_to_app_success_v2_en.pdf* (2015)

[23] Pons Capote, Olga: Introducción a las bases de datos. El modelo relacional. En: *Thomson Paraninfo, S.A* (2005)

[24] Qubaisi., Kevin Frazier-Tim Neumann-Michelle Posch-Khaled A.: Vehicle to Bicycle Communication. En: *American University*

[25] Radio, Caracol: BiGo, el emprendimiento que ayuda a los ciclistas. En: *Recuperado de https://caracol.com.co/programa/2019/11/14/hoy$_p$or$_h$oy/*1573747130_2*98972.html*(2019, 14*noviembre*)

[26] Ravulavaru, Arvind: Learning Ionic 2. En: *2nd ed. Packt Publishing* (2017)

[27] Ridjanovic, Dzenan ; Balbaert, Ivo: Learning Dart. En: *2nd ed. Packt Publishing* (2015)

[28] Space, Future: Tipos de aplicaciones móviles: nativas, web e híbridas. En: *Url https://www.futurespace.es/tipos-de-aplicaciones-moviles*

[29] V, Javier Simón Cuello J.: Diseñando apps para moviles. En: *Argentina: TugaMovil* (2013)