Neiva, 22 de septiembre del 2022

Señores

CENTRO DE INFORMACIÓN Y DOCUMENTACIÓN

UNIVERSIDAD SURCOLOMBIANA

Ciudad

El (Los) suscrito(s):

Diana Sofía Afanador Nuñez, con C.C. No. 1075316199. Autor(es) de la tesis y/o trabajo de grado titulado APITIC: DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA APOYAR EL ÁREA TIC DE INGELEC GROUP S.A.S. presentado y aprobado en el año 2022 como requisito para optar al título de Ingeniero Electrónico.

Autorizo (amos) al CENTRO DE INFORMACIÓN Y DOCUMENTACIÓN de la Universidad Surcolombiana para que, con fines académicos, muestre al país y el exterior la producción intelectual de la Universidad Surcolombiana, a través de la visibilidad de su contenido de la siguiente manera:

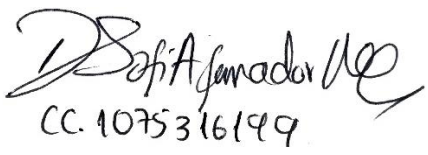
- Los usuarios puedan consultar el contenido de este trabajo de grado en los sitios web que administra la Universidad, en bases de datos, repositorio digital, catálogos y en otros sitios web, redes y sistemas de información nacionales e internacionales “open access” y en las redes de información con las cuales tenga convenio la Institución.

- Permita la consulta, la reproducción y préstamo a los usuarios interesados en el contenido de este trabajo, para todos los usos que tengan finalidad académica, ya sea en formato Cd-Rom o digital desde internet, intranet, etc., y en general para cualquier formato conocido o por conocer, dentro de los términos establecidos en la Ley 23 de 1982, Ley 44 de 1993, Decisión Andina 351 de 1993, Decreto 460 de 1995 y demás normas generales sobre la materia.

- Continúo conservando los correspondientes derechos sin modificación o restricción alguna; puesto que, de acuerdo con la legislación colombiana aplicable, el presente es un acuerdo jurídico que en ningún caso conlleva la enajenación del derecho de autor y sus conexos.

De conformidad con lo establecido en el artículo 30 de la Ley 23 de 1982 y el artículo 11 de la Decisión Andina 351 de 1993, “Los derechos morales sobre el trabajo son propiedad de los autores” , los cuales son irrenunciables, imprescriptibles, inembargables e inalienables.

EL AUTOR/ESTUDIANTE: DIANA SOFÍA AFANADOR NUÑEZ

Firma: 

TÍTULO COMPLETO DEL TRABAJO: APITIC: DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN MOVIL PARA APOYAR EL ÁREA TIC DE INGELEC GROUP S.A.S.

AUTOR O AUTORES:

Primero y Segundo Apellido	Primero y Segundo Nombre
Afanador Núñez	Diana Sofía

DIRECTOR Y CODIRECTOR TESIS:

Primero y Segundo Apellido	Primero y Segundo Nombre
Molina Mosquera	Johan Julián

ASESOR (ES):

Primero y Segundo Apellido	Primero y Segundo Nombre

PARA OPTAR AL TÍTULO DE: INGENIERO ELECTRÓNICO

FACULTAD: INGENIERÍA

PROGRAMA O POSGRADO: ELECTRÓNICA

CIUDAD: NEIVA **AÑO DE PRESENTACIÓN:** 2022 **NÚMERO DE PÁGINAS: 56**

TIPO DE ILUSTRACIONES (Marcar con una **X**):

Diagramas__X__ Fotografías__X__ Grabaciones en discos____ Ilustraciones en general__X___ Grabados___ Láminas___ Litografías___ Mapas___ Música impresa___ Planos___ Retratos___ Sin ilustraciones___ Tablas o Cuadros__

SOFTWARE requerido y/o especializado para la lectura del documento:

MATERIAL ANEXO:

PREMIO O DISTINCIÓN *(En caso de ser LAUREADAS o Meritoria)*:

PALABRAS CLAVES EN ESPAÑOL E INGLÉS:

Español	Inglés	Español	Inglés
1. Desarrollo en Java	Java Development	6. Adaptador	Adapter
2. Bases de datos	Databases	7. Entorno	Environment
3. Android Studio	Android Studio	8. Requisitos Sistema	System Requirements
4. Clases Java	Java Classes	9. Usuarios	Users
5. Almacenamiento Nube	Cloud Storage	10. Foro	Forum

RESUMEN DEL CONTENIDO: (Máximo 250 palabras)

El presente proyecto expone el diseño e implementación de una aplicación móvil desarrollada en Android Studio, utilizando el lenguaje de programación Java y la base de datos Firebase, realizada con el fin de proponer una alternativa para apoyar el área de Tecnologías de la Información y las Comunicaciones (TIC) de la empresa Ingelec Group S.A.S.

La aplicación móvil APITIC cuenta con los siguientes requisitos del sistema: registro e identificación del usuario, perfil del usuario, pantalla de bienvenida, acceso a los manuales de configuración, cronograma de actividades, análisis de productividad y foro interactivo, cuyo desarrollo se plantea detalladamente a lo largo de este documento.

ABSTRACT: (Máximo 250 palabras)

This project exposes the design and implementation of a mobile application developed in Android Studio, using the Java programming language and the Firebase database, to propose an alternative to support the area of Information Technology and Communications of the company Ingelec Group S.A.S.

The APITIC mobile application has the following system requirements: user registration and identification, user profile, splash screen, access to configuration manuals, schedule of activities, productivity analysis and interactive forum, whose development is outlined in detail at throughout this document.

APROBACION DE LA TESIS

Nombre Presidente Jurado:

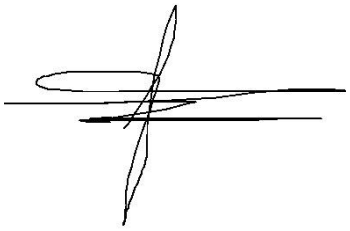
Firma:

Nombre Jurado: MARTIN DIOMEDES BRAVO OBANDO

Firma:

Nombre Jurado: JESUS DAVID QUINTERO POLANCO

Firma:



APITIC: DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA APOYAR EL ÁREA TIC DE INGELEC GROUP S.A.S

DIANA SOFÍA AFANADOR NÚÑEZ

UNIVERSIDAD SURCOLOMBIANA

FACULTAD DE INGENIERÍA

PROGRAMA DE INGENIERÍA ELECTRÓNICA

NEIVA – HUILA

2022

APITIC: DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA APOYAR EL ÁREA TIC DE INGELEC GROUP S.A.S

Autor:
DIANA SOFÍA AFANADOR NÚÑEZ

Trabajo de grado para optar al título de
INGENIERO ELECTRÓNICO

Director:
ING. JOHAN JULIAN MOLINA MOSQUERA

UNIVERSIDAD SURCOLOMBIANA

FACULTAD DE INGENIERÍA

PROGRAMA DE INGENIERÍA ELECTRÓNICA

NEIVA – HUILA

2022

Nota de aceptación:

_____

_____

_____

_____

_____

_____

_____

_____
Firma del Director del Proyecto

_____
Firma del primer jurado

_____
Firma del segundo jurado

Neiva, agosto del 2022

DEDICATORIA

A mi madre, Diela Nuñez Ramos, por su infinito amor, paciencia y fuerza de voluntad. Lo más importante lo aprendí en el hogar que creó para mí.
A Dios, por guiarme siempre por el camino que necesito ir para convertirme en una mejor versión.

A mi hermana, Dzoara Selene Nuñez.

AGRADECIMIENTOS

El más sincero agradecimiento a mi madre que me ha apoyado con todo su amor y de todas las maneras posibles durante este viaje llamado vida. A los ingenieros del programa de Ingeniería Electrónica de la Universidad Surcolombiana, quienes me han brindado sus conocimientos durante estos años.

Igualmente quiero resaltar mi agradecimiento al Ing. Johan Julián Molina por su dirección y seguimiento durante la realización del proyecto, y porque gracias su labor, el desarrollo de aplicaciones se ha convertido en un pasión.

Finalmente, agradezco a la empresa Ingelec Group S.A.S. por permitirme vivir y aprender grandes cosas de esta experiencia.

CONTENIDO

LISTADO DE FIGURAS

RESUMEN

El presente proyecto expone el diseño e implementación de una aplicación móvil desarrollada en Android Studio, utilizando el lenguaje de programación Java y la base de datos Firebase, realizada con el fin de proponer una alternativa para apoyar el área de *Tecnologías de la Información y las Comunicaciones (TIC)* de la empresa Ingelec Group S.A.S.

La aplicación móvil *APITIC* cuenta con los siguientes requisitos del sistema: registro e identificación del usuario, perfil del usuario, pantalla de bienvenida, acceso a los manuales de configuración, cronograma de actividades, análisis de productividad y foro interactivo, cuyo desarrollo se plantea detalladamente a lo largo de este documento.

ABSTRACT

This project exposes the design and implementation of a mobile application developed in Android Studio, using the Java programming language and the Firebase database, to propose an alternative to support the area of *Information Technology and Communications* (TIC in Spanish) of the company Ingelec Group S.A.S.

The *APITIC* mobile application has the following system requirements: user registration and identification, user profile, splash screen, access to configuration manuals, schedule of activities, productivity analysis and interactive forum, whose development is outlined in detail at throughout this document.

1. INTRODUCCIÓN

El desarrollo de la pasantía se hará en torno al proyecto Claro Mintic 7K, que tiene como propósito construir 14 mil Centros Digitales (CD), de los cuales 7 mil están encargados a Claro y que Claro ha contratado los servicios de Ingelec Group para la construcción de algunos CDs en los departamentos del Huila, Tolima, Caquetá y Santander.

Para la implementación de cada Centro Digital, se contemplan por diferentes etapas que son denominadas como: servicios requeridos, gestión de permisos, estudio de campo, construcción de la última milla, instalación, integración e interventoría. Por consiguiente, no todos los centros digitales se encuentran en las mismas etapas y es necesario adaptarse según sea la ocasión, manteniéndose actualizado constantemente de lo que sucede tanto en campo como en la zona administrativa.

Lo anterior es causa de la ausencia de un medio de comunicación preciso que permita conocer la situación actual del técnico en campo para poder hacer un seguimiento de productividad. Actualmente, la herramienta comunicativa que se utiliza es el WhatsApp, pero no es la más optima, ya que la mayoría de los técnicos no siguen la instrucción de reportarse y se generan vacíos en su tiempo laboral, en los que la empresa pierde dinero y se atrasa la construcción de los Centro Digitales, pues a diferencia de la zona administrativa, en campo no hay cómo controlar el rendimiento de los trabajadores.

Por otra parte, en muchas ocasiones, los técnicos de campo se encuentran en situaciones en las que carecen del conocimiento necesario para afrontar la solución de un problema, durante la ejecución de los diferentes proyectos que tiene Ingelec en el área de las telecomunicaciones. Lo cual ocasiona atrasos en la entrega del proyecto y que el ingeniero de soporte se sobrecargue de tareas, disminuyendo su productividad.

En consecuencia, se hace necesario implementar una solución que permita mejorar el área TIC de la empresa Ingelec Group en su organización, control de calidad y evaluación de productividad de lo que se vive en campo. De igual manera, brindar apoyo durante la ejecución de proyectos importantes en los que se espera aplicar los conocimientos adquiridos y aprender otros nuevos.

2. JUSTIFICACIÓN

Durante la pasantía que tendrá una duración de 6 meses, se realizará un proyecto que ofrezca una mejora en el desarrollo de las actividades del área TIC y que, de manera simultánea, se brinde apoyo al proyecto Mintic 7K. Teniendo en cuenta lo anterior, la pasantía consistirá en:

1. Diseño y desarrollo de un aplicativo móvil en Android Studio que permita suplir necesidades que existen actualmente en el área TIC de Ingelec Group, como por ejemplo: tener un control del estado en el cual se encuentra cada centro digital, obtener un contacto más directo e interactivo entre el personal que se encuentra en campo y el personal administrativo para poder hacer un seguimiento de productividad de los empleados, implementar un manual/instructivo al cual puedan acceder los técnicos/tecnólogos para resolver problemas que se puedan presentar en campo (alineación de radios, falta de potencia, validación de frecuencia, entre otros).

2. Realizar un control de calidad de la preparación e instalación de las obras, mantenimientos y servicios desarrollados sobre las redes de telecomunicaciones de COMCEL para el desarrollo del proyecto Centros Digitales MINTIC 7K. Teniendo en cuenta los objetivos para hacer posible la entrega de cada centro digital que llega a manos del pasante, desde el momento en que se le entrega el caso hasta el momento en que Claro realiza la interventoría para finalmente pagar por el sitio.

3. Elaborar informes sobre los procesos de instalación de equipos de acceso (Routers, Access Point, Switches, UTMs, etc.), instalación de solución de energía (Tableros, UPS’s, baterías, celdas solares, rack cuartos de equipos, etc.), estudios de campo y visitas técnicas a los centros digitales del proyecto MINTIC 7K.

3. OBJETIVOS

3.1. OBJETIVO GENERAL

Formular soluciones para proyectos de ingeniería en el área de las telecomunicaciones y del desarrollo de aplicaciones, evaluando constantemente en la práctica los conocimientos adquiridos durante la carrera de ingeniería electrónica.

3.2. OBJETIVOS ESPECÍFICOS.

1. Diseñar el modelo de arquitectura backend – frontend del aplicativo de software.

2. Crear y diseñar una aplicación móvil en Android Studio que proporcione un apoyo al área TIC durante el desarrollo del proyecto Mintic 7K.

3. Identificar necesidades que existen al momento de desarrollar los procesos de instalación y puesta en marcha del servicio de internet en zonas rurales y/o apartadas de diferentes partes del país.

4. Categorizar las etapas de los procesos de implementación de la última milla según el medio (fibra óptica, radio frecuencia o satélite) con base al estudio de campo de los centros digitales.

4. MARCO TEORICO

4.1. ANDROID STUDIO

Android Studio es un entorno de desarrollo poderoso y sofisticado, diseñado con el propósito específico de desarrollar, probar y empaquetar aplicaciones de Android. Se puede descargar, junto con el SDK de Android, como un solo paquete. Es una colección de herramientas y componentes. Muchas de estas herramientas se instalan y actualizan independientemente unas de otras.

Android Studio ofrece la oportunidad de programar en 2 lenguajes de programación: Kotlin y Java. Para propósitos de este proyecto, se eligió Java ya que hay una amplia colección de bibliotecas, marcos y herramientas para el desarrollo de aplicaciones.

4.2. LENGUAJE DE PROGRAMACIÓN: JAVA

Java es uno de los lenguajes de programación más populares del mundo y fue creado hace más de dos décadas (1995), es un lenguaje orientado a objetos que proporciona una estructura clara a los programas y permite reutilizar el código, lo que reduce los costos de desarrollo. Como Java está cerca de C++ y C# , facilita a los programadores cambiar a Java o viceversa. Java es código abierto y aunque es gratuito para el uso personal y el desarrollo a pequeñas escalas, las grandes empresas que lo utilizan deben pagar una licencia.

4.3. PROGRAMACIÓN ORIENTADA A OBJETOS: POO

POO se refiere a lenguajes que usan objetos en la programación como principal fuente para implementar el rumbo del código. Los objetos son vistos por el usuario final, realizando tareas asignadas por el desarrollador. La programación orientada a objetos tiene como objetivo implementar entidades del mundo real como herencia, ocultación, polimorfismo, etc. en la programación. El objetivo principal de POO es unir los datos y las funciones que operan en ellos para que ninguna otra parte del código pueda acceder a estos datos excepto esa función.
Los conceptos más importantes de POO utilizados en el proyecto, son: clases, objetos, métodos.

4.3.1. CLASES Y OBJETOS EN JAVA

Una clase es un plano o prototipo definido por el usuario a partir del cual se crean objetos. Representa el conjunto de atributos o métodos que son comunes a todos los objetos de un tipo. Además, todas las clases tienen un método llamado constructor, el cual se ejecuta automáticamente cuando se crea una instancia.

Un objeto es una unidad básica de Programación Orientada a Objetos y representa entidades de la vida real. Un programa típico de Java crea muchos objetos que, como sabe, interactúan invocando métodos. Un objeto consta de:

- Estado: Está representado por los atributos de un objeto. También refleja las propiedades de un objeto.

- Comportamiento: Está representado por métodos de un objeto. También refleja la respuesta de un objeto con otros objetos.

- Identidad : Da un nombre único a un objeto y permite que un objeto interactúe con otros objetos.

Al principio es complejo entender la diferencia entre un objeto y una clase pero podría tomarse como que la clase es el concepto abstracto de un objeto, mientras que el objeto es el elemento final de la clase. Por ejemplo, se tiene una clase Vehículo, la cual tiene atributos como color, marca, modelo, tipo, etc. Y métodos como encender luces, iniciar marcha, detenerse, etc.

A continuación, entra el concepto de objeto: cuando se instancia una clase, se crea un objeto de la clase. Todas las instancias comparten atributos y métodos pero los valores de esos atributos son únicos para cada objeto, es decir, que una clase puede tener *n* cantidad de instancias y por lo tanto *n* cantidad de objetos.

4.3.2. METODOS DE UNA CLASE EN JAVA

Un método es una colección de declaraciones que realizan una tarea específica y devuelven el resultado al usuario que llama, aunque también puede haber métodos que no devuelvan nada. Los métodos, entonces, permiten reutilizar el código sin volver a escribirlo.

Los métodos tienen un tipo de acceso y pueden ser públicos (*public*) es decir que puede ser accedido fuera de la clase, protegidos (*protected*) es decir que se pueden acceder dentro de la clase en la que se define y en sus subclases y, privados (*private*) que solo se pueden acceder dentro de la clase.

4.4. FIREBASE

Firebase es un software de desarrollo de aplicaciones respaldado por Google que permite a los desarrolladores desarrollar aplicaciones iOS, Android y web. Firebase proporciona herramientas para realizar un seguimiento de los análisis, generar informes y solucionar fallas de aplicaciones, y crear experimentos de marketing y productos.

Los servicios que ofrece Firebase están alojados en la nube por lo que hay componentes de back-end que Google mantiene y opera en su totalidad, de manera que es una herramienta amigable con el desarrollador. Los SDK de cliente proporcionados por Firebase interactúan con estos servicios back-end directamente, sin necesidad de establecer un software intermediario entre aplicación y el servicio.

Firebase es una herramienta poco tradicional ya que generalmente se implica programar tanto en back-end como en front-end. El código del front-end simplemente invoca los puntos finales de la API expuestos por el back-end, y el código del back-end realmente hace el trabajo. Sin embargo, con los servicios de Firebase, se pasa por alto el back-end tradicional y se pone el trabajo en el cliente. Lo mejor es que Firebase Console proporciona acceso administrativo a cada uno de estos servicios.

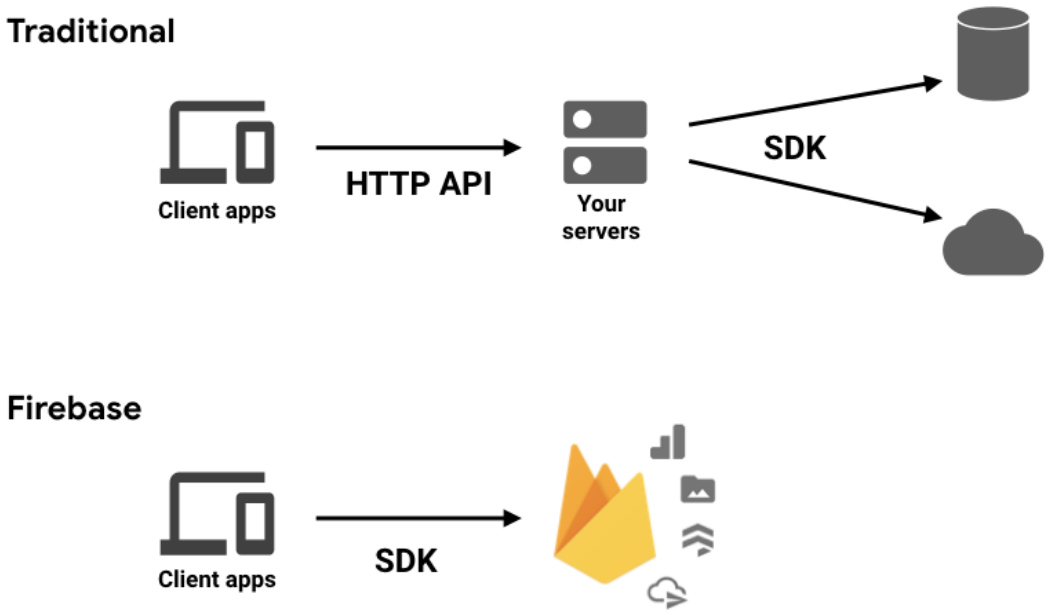


Figura 1. Operatividad Firebase vs bases de datos tradicionales
Fuente: https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0

Servicios de Firebase que se utilizaran en este proyecto:

1. *Firebase Authentication:* facilita la creación de sistemas de autenticación seguros y mejora la experiencia de inicio de sesión y de incorporación para los usuarios. Esta función ofrece una solución de identidad completa, compatible con cuentas de correo electrónico y contraseña, autenticación

telefónica, así como inicio de sesión de Google, Facebook, GitHub , Twitter y más.

2. *Firebase Realtime Database:* es una base de datos NoSQL alojada en la nube que te permite almacenar y sincronizar datos entre tus usuarios en tiempo real. Se incluye en los SDK web y para dispositivos móviles, de manera que puedas crear apps sin la necesidad de usar servidores. Cuando los usuarios se desconectan, los SDK de Realtime Database usan la caché local del dispositivo para publicar y almacenar cambios. Cuando el dispositivo se conecta, los datos locales se sincronizan de manera automática.

3. *Cloud Firestore:* es una base de datos de documentos NoSQL que permite almacenar, sincronizar y consultar fácilmente datos en tus apps web y para dispositivos móviles a escala global. Los datos se almacenan en colecciones y documentos para estructurar con facilidad. Se crean jerarquías para almacenar datos relacionados y recuperar los datos que se necesitan mediante consultas expresivas de manera sencilla. Todas las consultas se escalan con el tamaño del conjunto de resultados (y no con el del conjunto de datos), por lo que la aplicación está lista para escalar desde el primer día.

4. *Cloud Storage:* se diseñó para ayudar a almacenar y procesar con rapidez y facilidad el contenido generado por usuarios, como fotos y videos.

4.5. PROYECTO MINTIC 7K

La empresa Ingelec Group S.A.S. brinda los lineamientos básicos para el diseño, suministro y construcción de todas las obras civiles necesarias, garantizando que sean funcionales, sostenibles y económicamente viables. El proyecto 7K busca construir alrededor de 14 mil centros digitales en los 32 departamentos del país. Ingelec es el aliado de Claro para la construcción de los centros en el Huila, Tolima, Caquetá y algunos en los Santanderes.

Teniendo en cuenta lo anterior, los centros digitales son entonces establecimientos y sede educativas de carácter público ubicados en zonas apartadas del país, en donde se provee el servicio de acceso a internet. Para la construcción de cada centro, Claro provee los siguientes dispositivos: 3 Access Points marca Cambium Network (2 Outdoor referencia CNPilot E510 y 1 Indoor referencia CNPilot E410), 1 routerboard marca Mikrotik (referencia HEX POE), 1 kit de energía de respaldo que se compone de 1 UPS y 1 batería. Para esta último caso, hay 4 tipos de kits con marcas diferentes: K-Star, Eaton, Powersun y Loxus.

En cuanto la dispositivo de ultima milla, Claro también lo provee en ciertos casos. Para el proyecto MINTIC 7K se manejaron 3 tipos de ultima milla: satelital, tercero y radio.

4.5.1. ACCESS POINTS

Un punto de acceso es un dispositivo que crea una red de área local inalámbrica, o WLAN, en una zona abierta o encerrada. Un punto de acceso se conecta a un router, conmutador o concentrador con cable a través de un cable Ethernet y proyecta una señal Wi-Fi a un área designada.

Lo mejor de la marca Cambium Networks es que todos los puntos de acceso de cnPilot son administrados por cnMaestro desde la nube. Esto proporciona un panel de control único para Wi-Fi, Ethernet, banda ancha inalámbrica fija y enrutadores domésticos de proveedores de servicios.

La administración de la nube de cnMaestro está incluida sin costo adicional e incluye estadísticas de red detalladas, gráficos de utilización de canales y herramientas remotas de solución de problemas para garantizar que el servicio esté siempre activo.

4.5.1.1. CAMBIUM NETWORKS: CNPILOT E410 INDOOR

Este Access Point está diseñado para espacios cerrados como empresas o escuelas, cuenta con un soporte integrado que facilita la instalación y se puede administrar por el servicio de la nube cnMaestro.

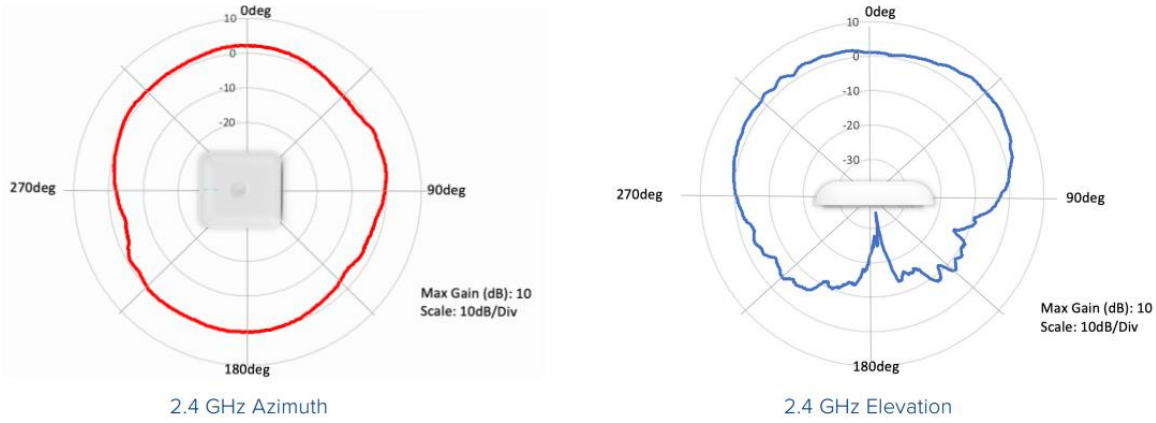


Figura 2. Diagrama de radiación AP cnPilot E410 indoor a 2.4 GHz

Fuente: https://www.cambiumnetworks.com/wp-content/uploads/2021/03/Cambium_Networks_data_sheet_cnPilot_e410_e600_family.pdf

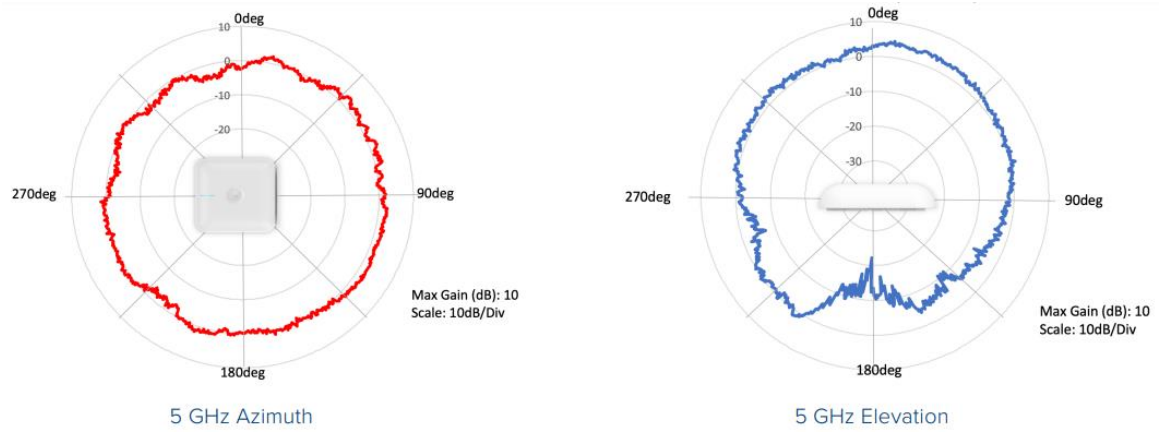


Figura 3. Diagrama de radiación AP cnPilot E410 indoor a 5 GHz

Fuente: https://www.cambiumnetworks.com/wp-content/uploads/2021/03/Cambium_Networks_data_sheet_cnPilot_e410_e600_family.pdf

4.5.1.2. CAMBIUM NETWORKS: CNPILOT E510 OUTDOOR

Este Access Point es ideal para comunidades al aire libre, Wi-Fi público o Wi-Fi empresarial, es omnidireccional y se adapta para aplicaciones de baja, media y alta densidad. Además, tiene certificación de choque y vibración de rieles e incluye protección contra sobretensiones y ESD. Posee un filtro LTE integrado que reduce impacto de las frecuencias LTE cercanas. Los intervalos de temperatura de funcionamiento son -40ºC a 65ºC para una amplia gama de aplicaciones.

Figura 4. AP cnPilot E510 outdoor

Fuente: https://www.cambiumnetworks.com/wp-content/uploads/2019/10/Cambium_Networks_data_sheet_cnPilot_e505_e510_e700_family.pdf

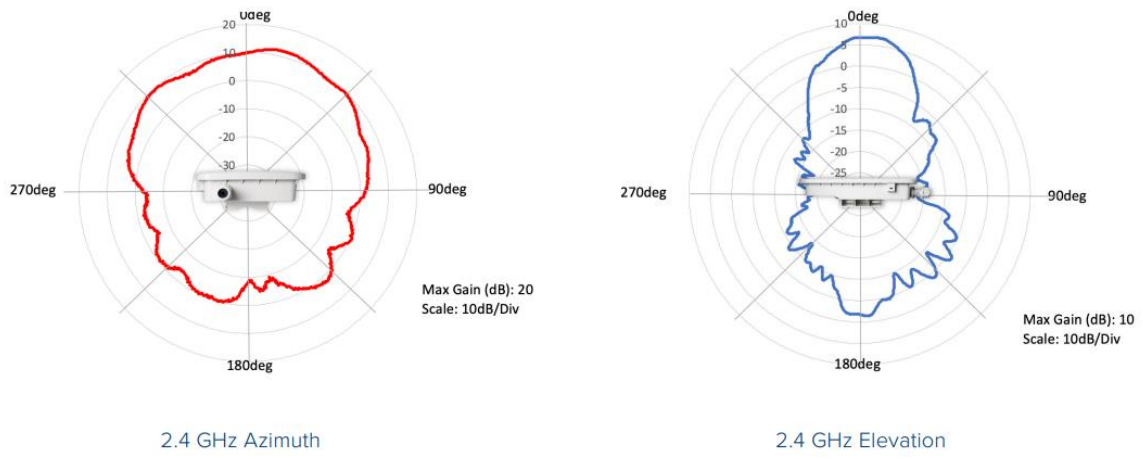


Figura 5. Diagrama de radiación AP cnPilot E510 outdoor a 2.4 GHz

Fuente: https://www.cambiumnetworks.com/wp-content/uploads/2019/10/Cambium_Networks_data_sheet_cnPilot_e505_e510_e700_family.pdf

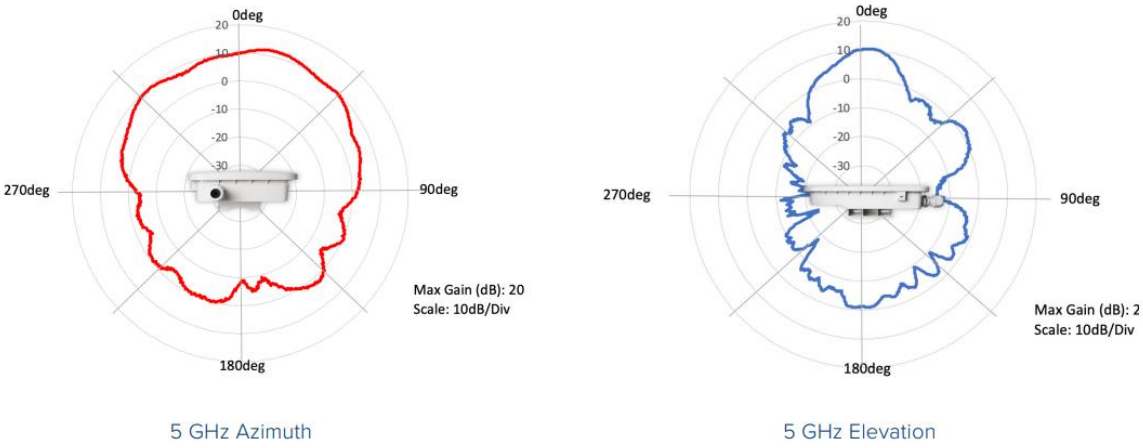


Figura 6. Diagrama de radiación AP cnPilot E510 outdoor a 5 GHz

Fuente: https://www.cambiumnetworks.com/wp-content/uploads/2019/10/Cambium_Networks_data_sheet_cnPilot_e505_e510_e700_family.pdf

4.5.2. ROUTERBOARD

Un router o encaminador es un dispositivo que permite interconectar redes con distinto prefijo en su dirección IP. Su función es la de establecer la mejor ruta que destinará a cada paquete de datos para llegar a la red y al dispositivo de destino. Básicamente, se encarga de llevar la conexión a los dispositivos, más no es el que provee el internet.

4.5.2.1. MIKROTIK: hEX PoE

hEX PoE es un enrutador Gigabit Ethernet de cinco puertos: un puerto USB 2.0 y un puerto SFP para agregar conectividad de fibra óptica. Los puertos 2-5 pueden alimentar otros dispositivos compatibles con PoE con el mismo voltaje que se aplica a la unidad, aquí van conectados los Access Points y la ultima milla.

Es asequible, pequeño y tiene una potente CPU de 800 MHz. La corriente máxima es de 1 A por puerto y todos los puertos Ethernet están protegidos. También admite entrada PoE pasiva y salida PoE pasiva o 802.3af/at. (Mikrotik, 2022)Puede alimentar dispositivos compatibles con modo at/af B (4,5+)(7,8-), si se utiliza un voltaje de entrada de 48-57.

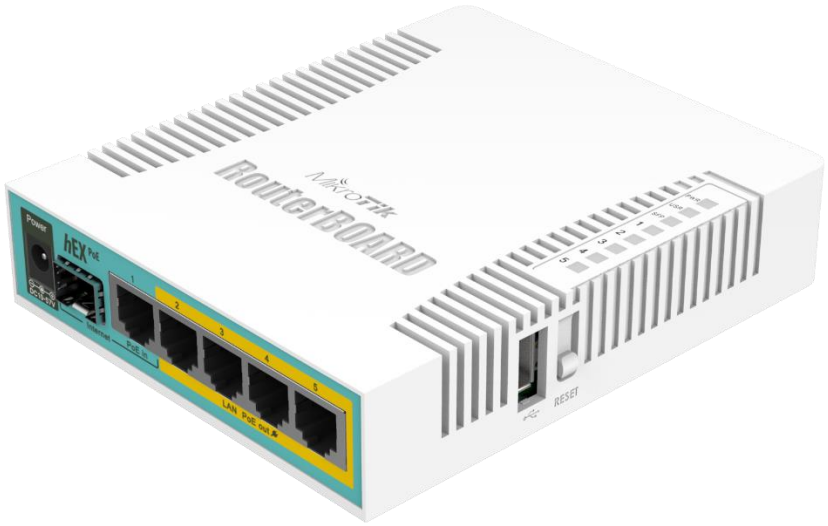


Figura 7. Mikrotik hEX PoE

Fuente: https://www.cambiumnetworks.com/wp-content/uploads/2019/10/Cambium_Networks_data_sheet_cnPilot_e505_e510_e700_family.pdf

4.5.3. KIT DE ENERGIA DE RESPALDO

Un kit de energía de respaldo permite garantizar un suministro sin interrupciones de energía eléctrica para un hogar, comercio u oficina. Para el proyecto MINTIC 7K, estos kits se componen de una UPS y una batería.

Al momento de producirse una bajón o pico del suministro de energía, o que haya un caída total, la UPS desde su banco de baterías genera automáticamente la energía para mantener encendidos los equipos que están conectados a ella. Para este caso, el respaldo debe durar 4 horas de acuerdo con el banco de baterías enviado por Claro para la instalación y la cantidad de equipos que se conectan.

4.5.4. ULTIMA MILLA

La última milla, en telecomunicaciones, se refiere al tramo final de una línea de comunicación, que llega al usuario final. Es decir, es la transmisión de datos desde el punto de inserción a una red preexistente y el centro digital donde se encuentra el usuario final.

El proyecto maneja 3 tipos de última milla: satelital, tercero y radio enlace. La elección de la última milla dependía de factores como ubicación del centro digital y distancia entre el centro y una estación base de Claro.

4.5.4.1. UM SATELITAL

El acceso a Internet mediante un satélite es la única opción viable en muchas zonas, especialmente zonas rurales, montañosas o de difícil acceso donde no existe tendido de cable, ni cobertura 3G, ni una estación base de Claro cercana.

Para la instalación de esta última milla es necesario una antena parabólica y un módem DVB-S específico para establecer una comunicación bidireccional. Para este caso, Claro proporciona un kit de HughesNet que consta de los dos dispositivos mencionados.

Su funcionamiento se basa en que cuando el usuario accede al servicio de internet desde el centro digital, se envía una señal desde tu antena parabólica al satélite de Hughes. El satélite envía la señal al centro de operaciones de la red el cual se dirige al sitio web, servidor de correo, etc., al que el usuario accedió y luego la señal viaja desde el satélite al centro digital, estableciendo una conexión en cuestión de segundos.

4.5.4.1.1. MODEM HUGHESNET: HT2010

El HT2010 es un potente modem satelital diseñado para proporcionar alta velocidad y acceso de banda ancha a usuarios de pequeñas y medianas empresas (PYMES) y redes empresariales distribuidas. Admite un canal directo DVB-S2X de banda ancha, entregando la mejor eficiencia de la industria. El canal de retorno utiliza potente y avanzada codificación de paridad de baja densidad (LDPC). El HT2010 entrega hasta 200 Mbps de rendimiento y la capacidad para admitir la mayoría de ancho de banda intensivo.

Además cuenta con un pequeño chasis de escritorio con un solo Gigabit Ethernet Puerto LAN, un puerto USB y una conexión IFL de un solo cable para la interfaz con la Unidad Exterior (ODU). El HT2010 funciona con radios multimodo HB210 y HB220 que son ideales para soportar canales de retorno de alta capacidad.

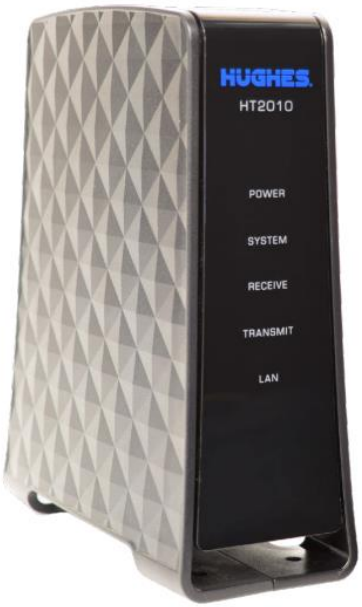


Figura 8. Modem satelital HT2010

Fuente: https://www.hughes.com/sites/hughes.com/files/2022-03/HT2010-Router.pdf

4.5.4.1.2. UNIDAD EXTERIOR (ODU): HB210

La unidad exterior Hughes HB210 (ODU) de última generación es un unidad de radio de banda ancha por satélite de alto rendimiento y bajo coste diseñada para establecer conectividad de alta velocidad. El ODU consiste en el conjunto de radio, que se monta en una serie de antenas, las cuales son controladas y alimentadas por el modem satelital HT2010 a través de un único cable IFL (transmisión integrada, recepción-LNB más OMT). Esta conexión se realiza a través del cable coaxial.

Figura 9. ODU satelital HB210

Fuente: https://www.hughesnet.com.co/sites/hughesnet.com.co/files/2019-09/HB210_Radio.pdf

4.5.4.2. UM TERCERO

El acceso a Internet mediante un tercero se convierte en una opción viable cuando una empresa proveedora de internet tiene cobertura en el zona en la que se encuentra ubicado el centro digital y es entonces cuando Claro contrata este servicio.

Cabe resaltar que el tercero puede establecer el servicio de internet a través de radio o satélite, pero en estos casos el aliado Ingelec Group S.A.S. no tiene que realizar ningún tipo de instalación del equipo de ultima milla y la responsabilidad de garantizar el acceso a internet recae sobre el tercero. Para el Huila y algunas zonas del Tolima, los proveedores de internet eran Stratecsa y Energostech.

4.5.4.3. UM RADIO ENLACE

El acceso a internet mediante radio enlace se refiere a un sistema electrónico de comunicación inalámbrica mediante ondas de radio que permite la transferencia de información entre dos o más puntos.

Hay dos elementos muy importantes para que esta transmisión sea efectiva, tiene que haber un transmisor y un receptor. En el transmisor se produce la señal de microondas de una determinada frecuencia, potencia y modulación, esta señal viaja por la línea de transmisión, que es un cable UTP CAT – 6, y llega a la antena del radio para que sea emitida al espacio libre. En el receptor se recoge esta señal y la envía por la línea de transmisión que conecta al receptor, la demodula y procesa para obtener la información.

Es muy importante que no haya obstáculos en la línea de vista entre la estación base donde se encuentra el transmisor y el centro digital que tiene el receptor. La distancia de funcionamiento está determinada por la frecuencia, el tamaño de la antena y la capacidad del enlace. Cuanta mayor es la frecuencia, mayor es la capacidad para transportar datos y menor es el tamaño de la antena que se necesita, pero más energía demanda y peor alcance tiene.

4.5.4.3.1. CAMBIUM NETWORKS: FORCE 300-19R

El Force 300-19R es un radio punto a punto muy potente y de alto rendimiento, que ofrece hasta 600Mbps de capacidad. Tiene una antena integrada de 19dBi con un ancho de haz estrecho y una mecánica IP67 confiable. Es excelente para áreas donde el clima es severo con frecuencia.

La distancia máxima que debe haber para establecer el radioenlace entre la BTS y el centro digital cuando se está utilizando este radio, es de 10 Km. Es compatible con el Force 3000L y 3000.

Figura 10. Cambium Networks Force 300-19R

Fuente: https://www.cambiumnetworks.com/products/epmp/epmp-force-300-19r/

4.5.4.3.2. CAMBIUM NETWORK: FORCE 300 CSM

El Force 300 CSM es un radio punto a punto muy potente y de alto rendimiento, está diseñado para operar en entornos de alta interferencia y proporciona un rendimiento superior de más de 500 Mbps de datos de usuario reales. Además, ofrece versatilidad en la transmisión ya que tiene distintas opciones para platos parabólicos.

Para el proyecto se utilizó una antena Cambium Networks parabólica de doble polaridad RDH4503C, la cual tiene una ganancia de 29.5 dBi y permite que la distancia máxima para establecer el radioenlace sea de 20 km. Este radio es igualmente compatible con el Force 3000L y 3000.

Figura 11. Cambium Networks Force 300CSM

Fuente: https://www.cambiumnetworks.com/products/epmp/epmp-force-300-csm/

4.5.4.3.3. CAMBIUM NETWORKS ePMP 3000

El ePMP 3000 es una antena de sectorial 4X4 diseñada para trabajar en el espectro de 5GHz, tiene una cobertura de 90 grados y permite la operación MIMO multiusuario 4X4. La antena de sectorial tiene una relación front to back mínima de 30 dB y una ganancia de 17 dBi.

Su mayor virtud es que tiene una cobertura constante, incluso cuando el centro digital se encuentra cerca de la BTS o en los bordes extremos. Además, tiene una ganancia constante dentro de su rango de frecuencias de 4,9 a 6,0 GHz, lo cual permite que se pueda seleccionar cualquier canal y se logre el rendimiento esperado.

Figura 12. Cambium Networks ePMP 3000

Fuente: https://www.cambiumnetworks.com/products/epmp/epmp-3000/

5. REQUISITOS DE LA APLICACIÓN MÓVIL

Inicialmente, se realizó un Mockup (prototipo de la aplicación o mapa del sitio) teniendo en cuenta los colores representativos de la empresa IngelecGroup S.A.S. (azul, blanco y naranja) y los requisitos de la aplicación, siendo estos:

1. Registro e identificación de usuarios
2. Perfil del usuario
3. Pantalla de bienvenida.
4. Acceso a los manuales de configuración.
5. Cronograma de actividades y notas.
6. Análisis de productividad
7. Foro interactivo

En un principio, se planteó desarrollar primero todo el front-end y luego el back-end, pero con la práctica se llegó a la conclusión que era mejor trabajarlo en conjunto, a medida que se fueran añadiendo los requerimientos del sistema. Para la aplicación se eligió utilizar la base de datos NoSQL Firebase, ya que es una herramienta elaborada por Google, al igual que Android Studio y por lo tanto se complementan, facilitando la programación.

5.1. REGISTRO E IDENTIFICACIÓN DE USUARIOS

El propósito de este requisito del sistema es poder identificar los usuarios que ingresan a la plataforma, ya que se vuelve clave para tener un control de los técnicos que hacen uso de la aplicación. Para esto, fue necesario utilizar una base de datos y enlazarla con Android Studio, para almacenar y actualizar la información desde el lado del usuario en tiempo real.

Durante la investigación, se encontró que Firebase brinda dos tipos de bases de datos: Cloud Firestore y Realtime Database, por lo que se tenía que considerar cual era la solución más apropiada para la aplicación, de acuerdo con la función que desempeña la base datos en este requisito, las operaciones con la información, modelos de datos y cantidad de instancias de base de datos. Teniendo en cuenta lo anterior, se eligió Realtime Database debido a que la base de datos va a ser utilizada para sincronizar datos con consultas básicas, la estructura de almacenamiento de datos es un árbol JSON y la cantidad de datos no superará los 100GB por ahora. Esta elección puede cambiar en el futuro en caso de que se requiera y no habría ningún problema para adaptarlas, ya que ambas opciones almacenan los mismos tipos de datos y las bibliotecas del cliente funcionan similar.

Para este requisito, se implementaron 2 *activitys* en Android Studio, uno para que el usuario pudiera iniciar sesión tomando los datos del correo electrónico y contraseña, y otro para que el usuario pudiera registrarse tomando datos como nombre completo, contraseña, número de teléfono y fecha de nacimiento.

La bibliotecas que se utilizaron para la autenticación del usuario en Android Studio son FirebaseAuth, FirebaseUser y AuthResult.

5.1.1. CREACIÓN DE USUARIO CON FIREBASE

El sistema de autenticación de usuarios de Firebase permite distintos métodos de inicio de sesión, divididos en 3 ramas: proveedores de acceso nativos, adicionales y personalizados. Para el proyecto, se deseaba que el inicio de sesión fuera nativo, es decir, que el usuario creara la cuenta a partir del correo electrónico y contraseña. No se implementó inicio de sesión con proveedores adicionales como Google, Facebook, Twitter o Yahoo, a petición de la empresa.

A partir de lo anterior, surgieron dos necesidades: que el usuario pudiera reestablecer la contraseña y que la cuenta de correo electrónico se tuviera que verificar para poder acceder a los beneficios de la aplicación. Para verificar el correo, se utilizó la función de Firebase *sendEmailVerification()*, la cual envía un email verificación al correo proporcionado por el usuario al momento de registrarse.

Para reestablecer la contraseña, se utilizó la función de Firebase *sendPasswordResetEmail(),* la cual envía un email al correo proporcionado para que el usuario cambie la contraseña.

Figura 13. Firebase Authentication: Registro e identificación de usuarios.

5.1.2. ORGANIZACIÓN DE DATOS Y CLASES

Como la creación de la cuenta se hizo a partir de uno de los servicios de Firebase, era necesario almacenar la información otorgada por el usuario en la base de datos Realtime Database para poder acceder a ella al momento de implementar el requisito *perfil del usuario*. No fue necesario definir ningún esquema desde el

servidor ya que la base de datos es NoSQL, es decir no relacional orientada a objetos. Únicamente se debió tener en cuenta que Firebase pide unas reglas de uso para la escritura y lectura de los datos, las cuales fueron:

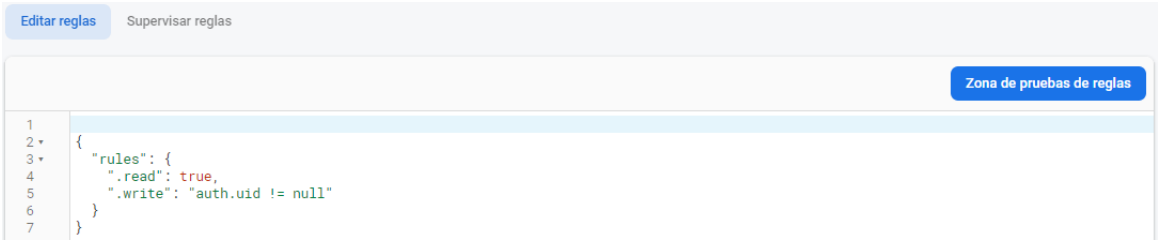


Figura 14. Reglas de lectura y escritura de la base de datos.

Por consiguiente, para la identificación y registro de los usuarios fue necesario crear una clase publica *Tecnicos*, para que se pudiera llamar en cualquier parte del programa y se generó el método o función constructor, el cual se ejecuta siempre que se crea una instancia. La clase *Tecnicos* tiene 5 atributos tipo *String*: id, nombre, correo electrónico, número de teléfono y fecha de nacimiento.

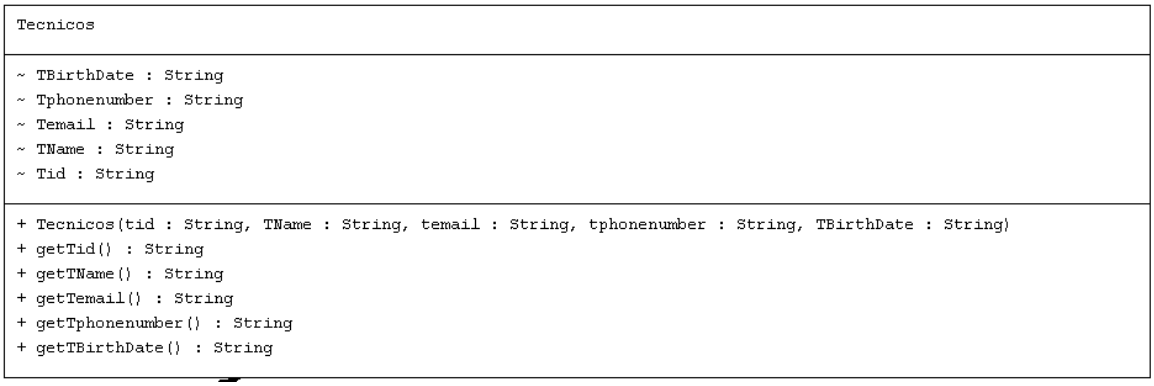


Figura 15. Diagrama UML clase pública Tecnicos

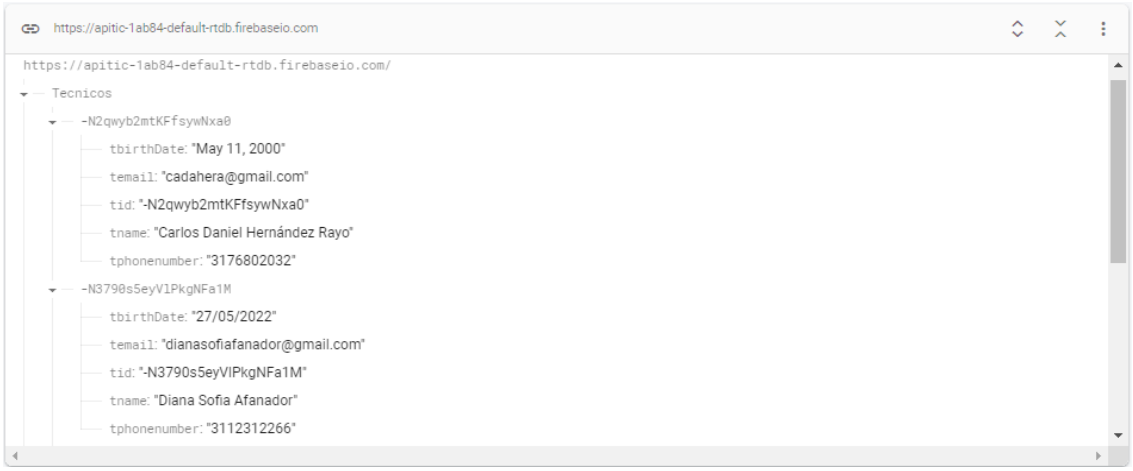


Figura 16. Realtime Database: Registro e identificación de usuarios.

A continuación el diagrama de flujo de este requisito del sistema:

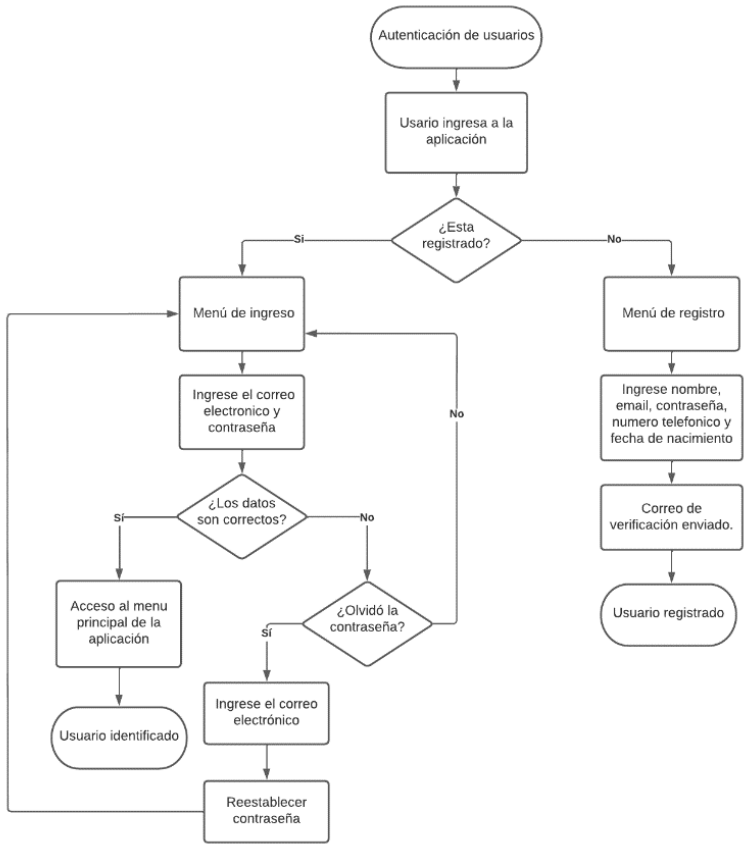


Figura 17. Diagrama de flujo autenticación usuarios

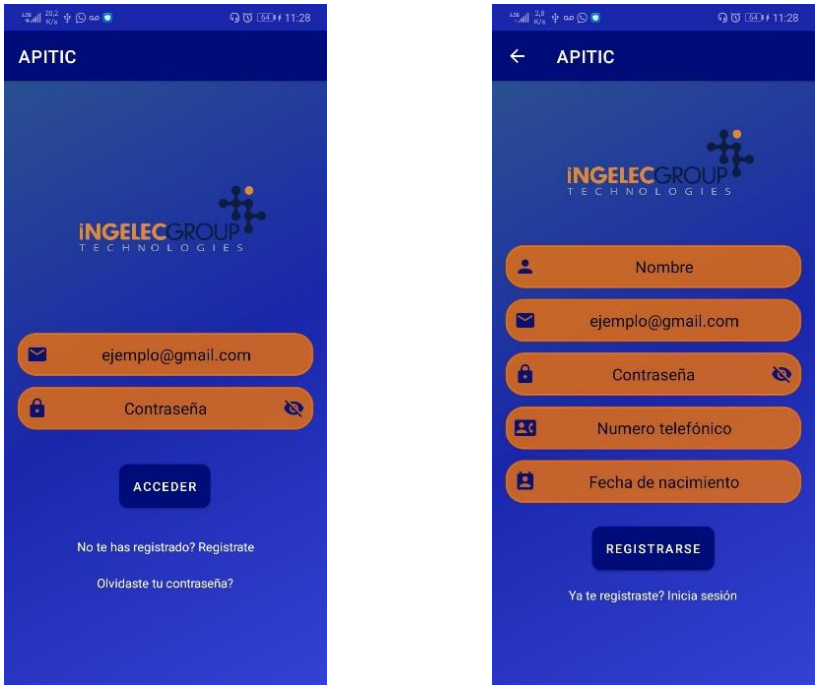


Figura 18. Android Studio: Interfaz registro e identificación de usuarios.

Para reestablecer la contraseña y evitar errores por caracteres, se aplicó la función *toLowerCase* de java, la cual convierte en minúsculas el dato tipo string de la caja de texto. Es importante tener en cuenta que Firebase no pone problema por la longitud de la contraseña, sin embargo cuando se va a entrar a la aplicación, aunque la contraseña sea la correcta, si no tiene 8 o más caracteres entonces no se podrá ingresar.

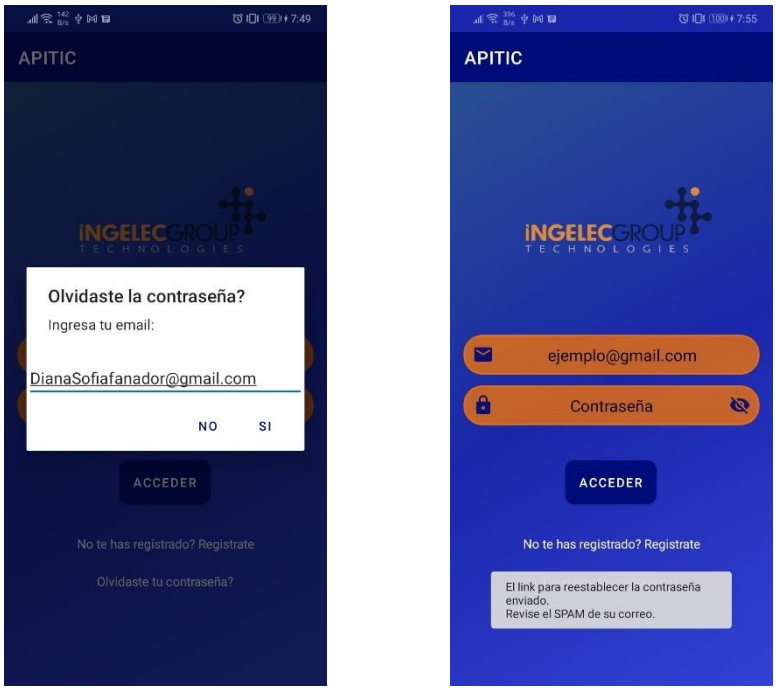


Figura 19. Interfaz: reestablecer contraseña.

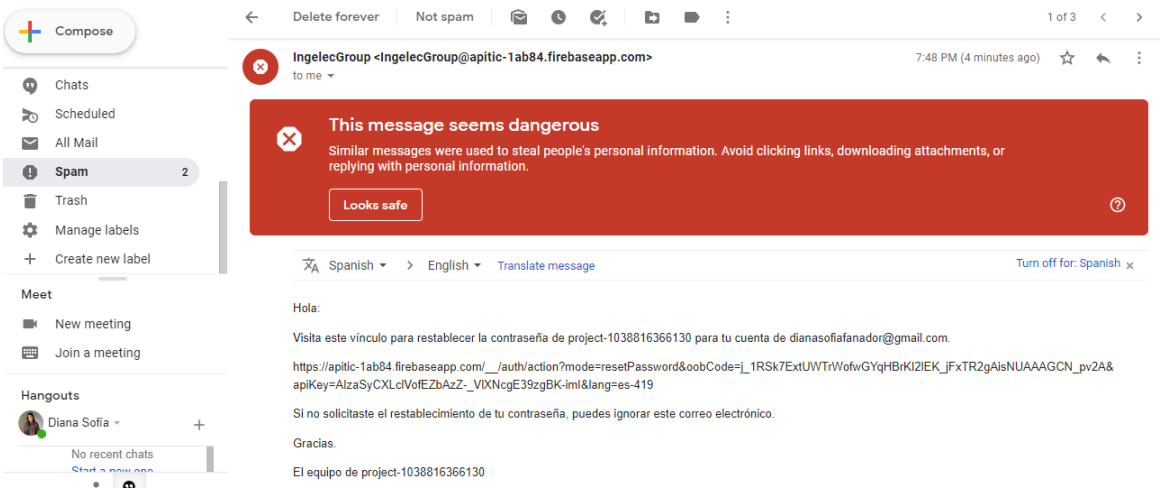


Figura 20. Reestablecer contraseña: correo en SPAM.

5.2. PERFIL DEL USUARIO

El propósito de este requisito es permitir al usuario acceder a la información que suministro al crear una cuenta en la aplicación, y que si lo desea, pueda modificar los datos y que estos automáticamente se actualicen en el Realtime Database.

Para empezar con el diseño e implementación de este requisito, fue necesario crear un *Navigation Drawer Activity,* que básicamente sería el cuerpo de todo el sistema, en donde un *activity* principal sería el que llamaría a la cadena de *fragments* necesarios para cumplir con el diseño del sistema.

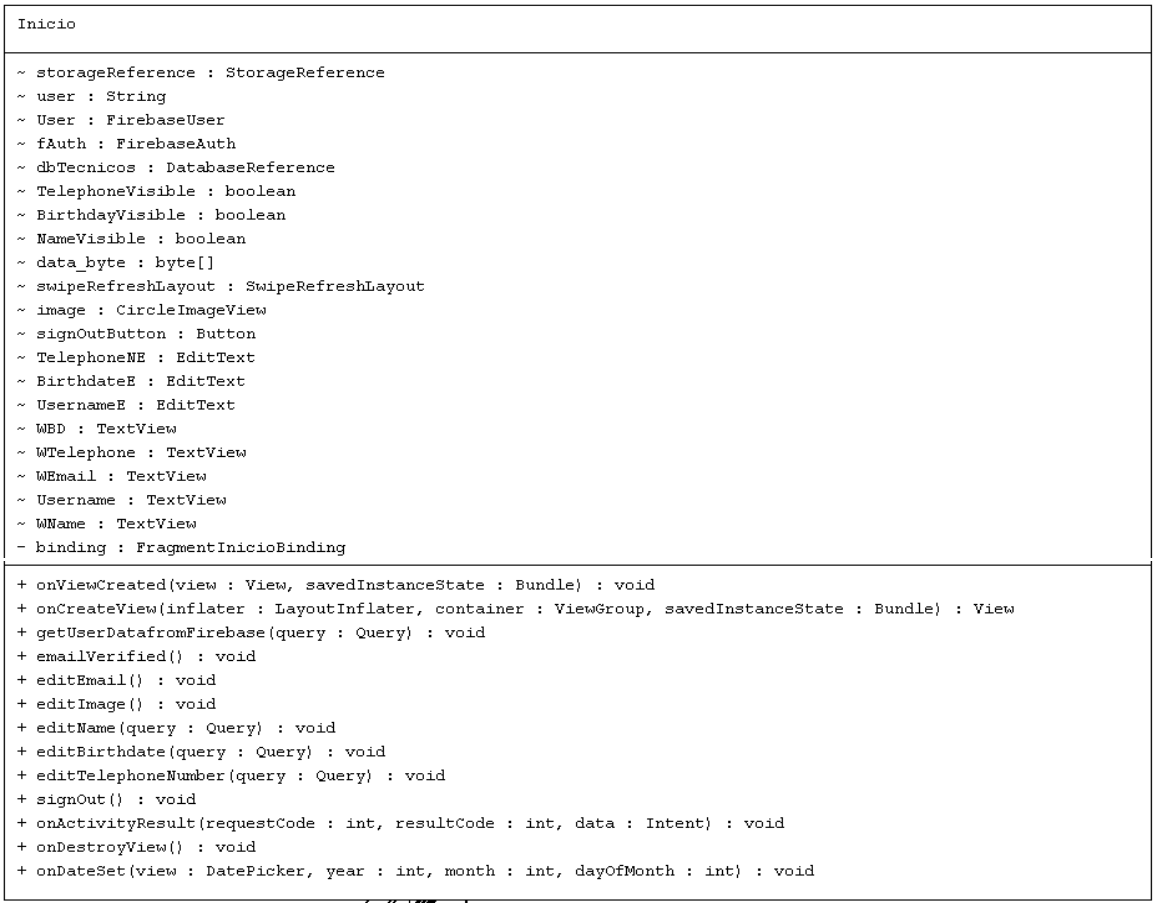


Figura 21. Diagrama UML del fragment Inicio

Para el inicio de sesión de un usuario nuevo que no ha verificado su correo, aparecerá un *Alert Dialog* que dificultará la experiencia dentro de la aplicación y adicionalmente no escribir datos en la base de datos. Cabe resaltar, que el *Alert Dialog* aparecerá solo en el fragment de inicio que es donde se reúne toda la información del usuario.

Para retornar la información desde Firebase, se utilizó la clase Query para realizar una consulta y leer los datos alojados en la referencia *“Tecnicos”,* ordenando el nodo por *“temail”* y buscando el email del usuario que ingresó.

En este fragment, el usuario tendrá la oportunidad de modificar el nombre, número de teléfono y fecha de nacimiento. A pesar de que, hay métodos de java y Firebase

que permiten actualizar el correo electrónico, se llegó a la conclusión que era mejor no hacerlo.

Adicionalmente, el usuario podrá añadir una imagen para personalizar su perfil. Esta imagen se almacenará en el *Cloud Storage* de Firebase y el nombre será el correo del usuario, de manera que siempre se escribirá la imagen en el mismo lugar para ahorrar espacio de almacenamiento, evitando que existan 2 fotos para el mismo usuario.

Durante esto, surgió la necesidad de comprimir el tamaño de la imagen para ahorrar espacio y que el proceso de lectura y escritura fuera *menos costosa*. Para esto, se usó el método *compress* de Bitmap, el cual escribe una versión comprimida del mapa de bits (imagen) en un flujo de salida especificado y este es enviado byte a byte al *Cloud Storage*. El proceso de lectura de la imagen se da al momento de retornar los datos del usuario que ha sido autenticado. Esto también se realiza con Bitmap, creando una archivo temporal para guardar ahí la imagen retornada y finalmente ponerla en el CircleImage del fragment.

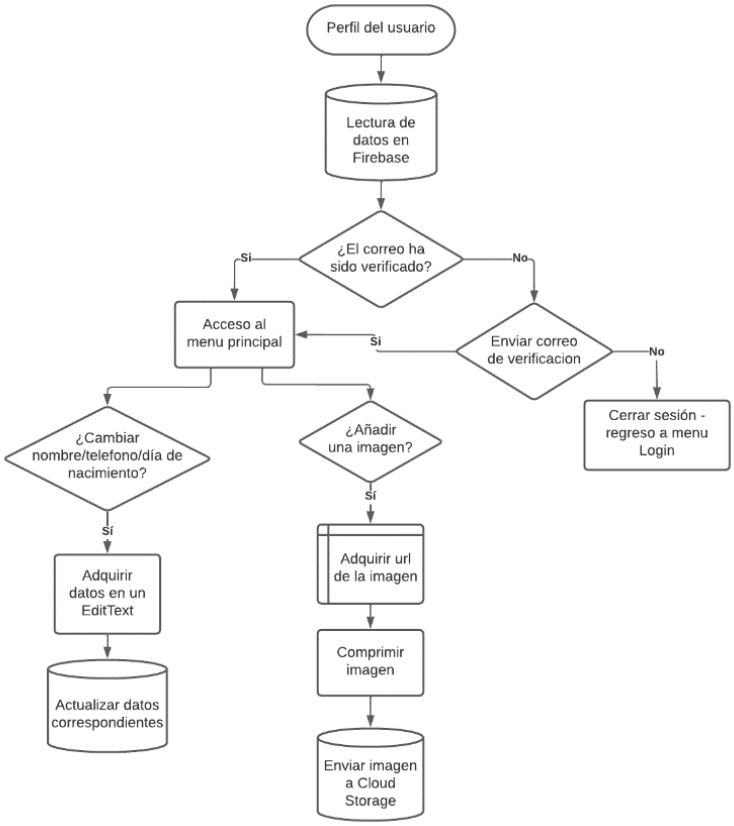


Figura 22. Diagrama de flujo perfil usuario.

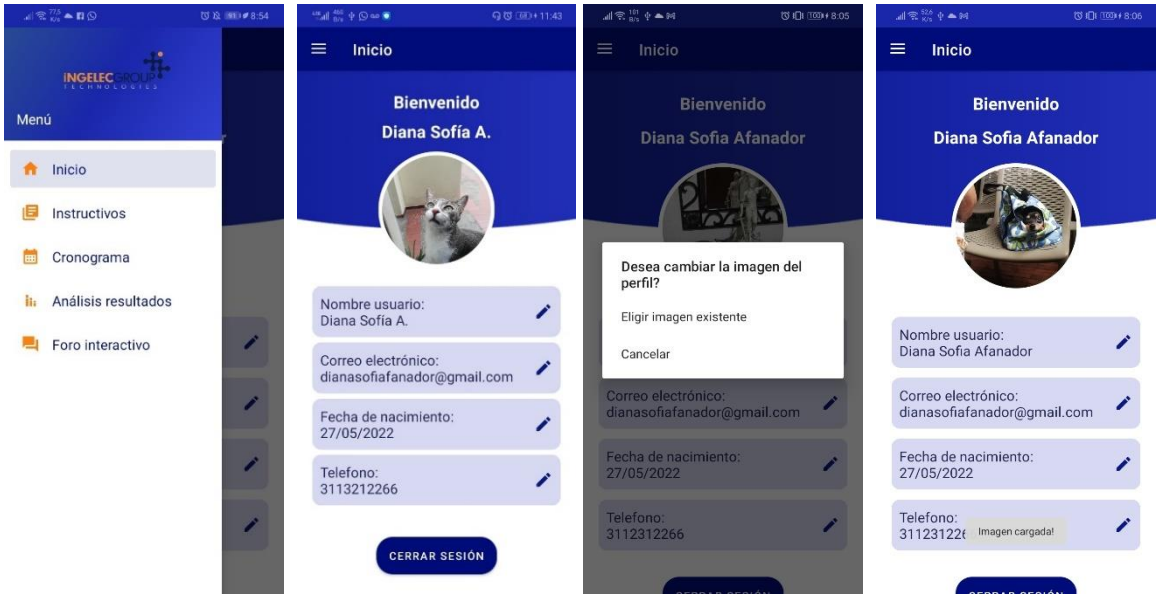


Figura 23. Interfaz: Perfil del usuario y Navigation Drawer Activity

5.3. PANTALLA DE BIENVENIDA

El propósito de este requisito es que al momento de entrar a la aplicación, aparezca un mensaje de bienvenida que en este caso es el logo de la empresa. Es un requisito que tienen la mayoría de las aplicaciones móviles y que a pesar de que no tiene ningún beneficio, eleva la aplicación a un nivel más profesional.

Figura 24. Interfaz: Pantalla de bienvenida.

5.4. ACCESO A LOS MANUALES DE CONFIGURACIÓN

El propósito de este requisito es permitir que el usuario tenga acceso a los manuales de configuración de los equipos utilizados durante el proyecto Claro Mintic 7K. Los manuales se dividen en 4 grupos: configuración de UPS, configuración modem y antena satelital, configuración de Access Points y configuración de radios. Cada grupo tiene más de 1 manual asociado.

Los manuales se encuentran almacenados en el *Cloud Storage*, aquí se hizo necesario que se crearan dos carpetas: una donde estuvieran los manuales en el tamaño original y otra en donde estuvieran los manuales comprimidos, con una menor calidad por supuesto.

Teniendo en cuenta lo anterior, para la implementación de este requisito fue necesario crear un fragment asociado al *activity* principal y además crear 4 *activitys* para cada grupo, de manera que desde el fragment de manuales, se pudieran acceder a los activitys y fuera posible enviar una bandera o variable adicional, que permitiera al activity conocer que manual debe desplegar dependiendo del valor de la bandera.

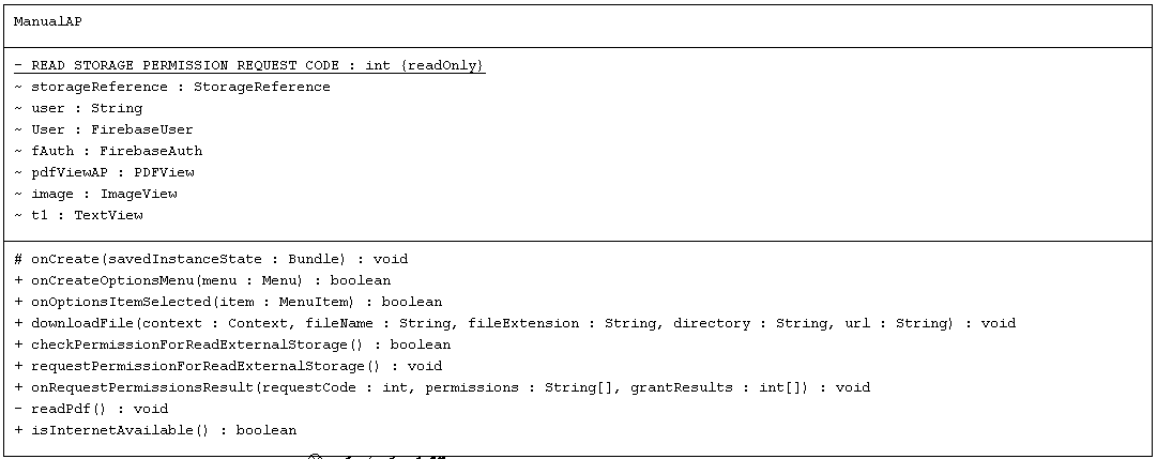


Figura 25. Diagrama UML del activity ManualAP.

Al momento que el usuario quiera acceder a un manual, el activity correspondiente recibe la bandera con un valor determinado. Si el usuario tiene conexión a internet, desplegará el manual comprimido y tendrá la opción de descargarlo, pero será el manual con la calidad original. Si el usuario no tiene conexión a internet y el manual esta descargado y guardado dentro de la ruta correspondiente, se desplegará el manual, de no ser así se mostrará un mensaje en pantalla haciendo saber que el manual no existe. Para esto es necesario que el usuario permita a la aplicación acceder al almacenamiento interno del dispositivo.

Figura 26. Cloud Storage: Manuales de configuración.

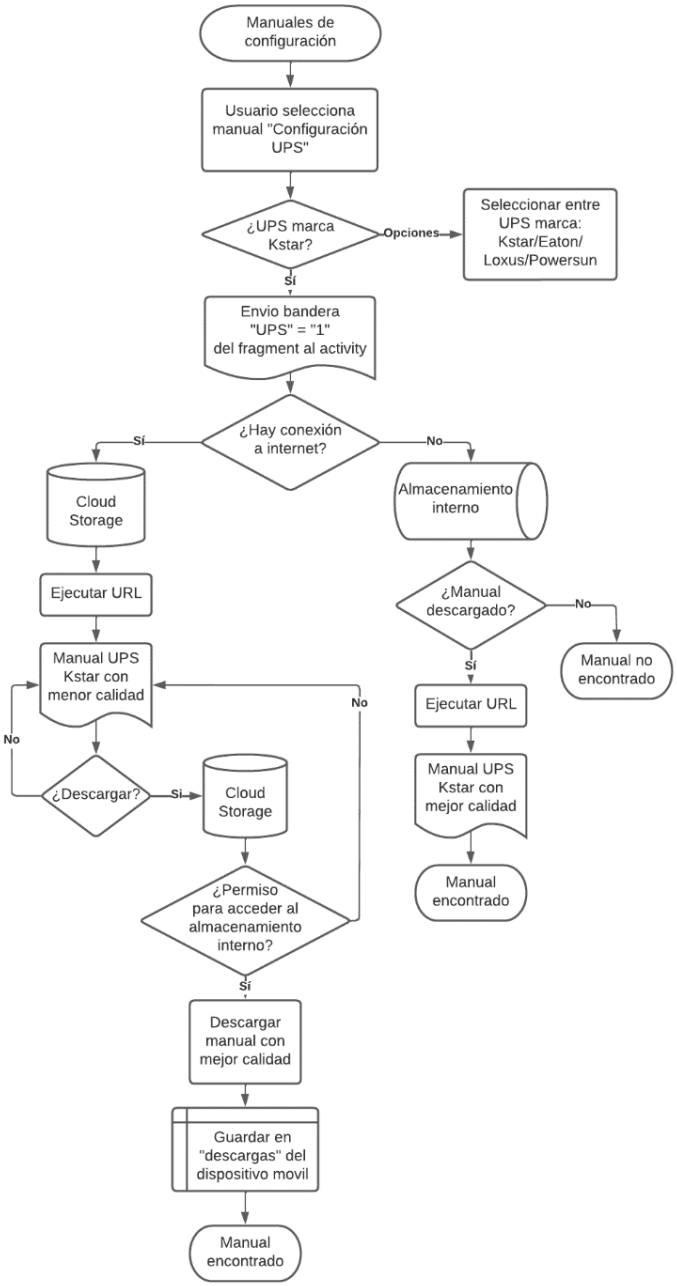


Figura 27. Diagrama de flujo manuales de configuración.

Para descargar y visualizar los manuales, se utiliza la librería *StorageReference* que representa una referencia a un objeto de Google Cloud Storage. La visualización en términos de programación significa que el documento se descarga temporalmente en el dispositivo y esto es lo que hace el método creado *VistaPDF().*

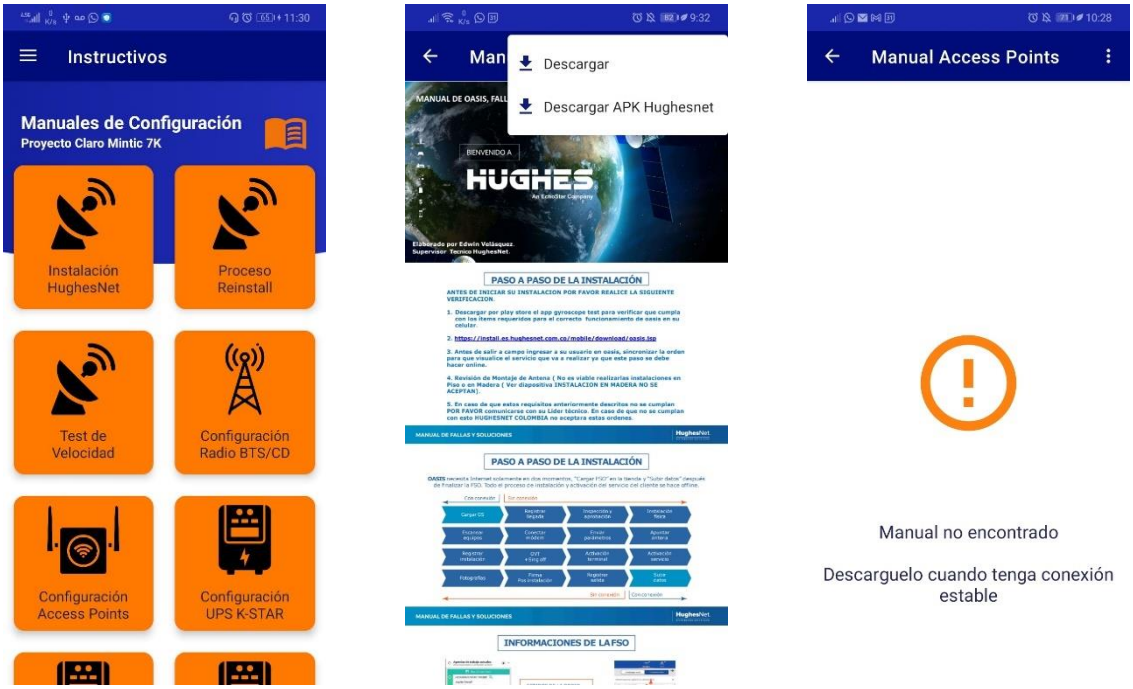


Figura 28. Interfaz: Acceso a los manuales de configuración

5.5. CRONOGRAMA DE ACTIVIDADES

El propósito de este requisito del sistema es que los usuarios lleven un control de las actividades que realizan diariamente, allí deberán indicar el centro de costos o ID beneficiario, el nombre del centro digital y el desarrollo de la actividad. Adicionalmente, se generará de manera automática la fecha y hora de inicio y última actualización de la actividad.

Para la implementación de este requisito, fue necesario crear un *fragment* Cronograma, el cual sería tomado como la vista principal, y otros tres *activitys* secundarios para editar, crear y mostrar cada actividad del cronograma. En este caso, se llegó a la conclusión que lo mejor era utilizar la base de datos *Cloud Firestore* en lugar de *Realtime Database*, ambas pertenecen a Firebase, debido a que permite crear colecciones de documentos y acceder a ellos más fácilmente.

Teniendo en cuenta lo anterior, se creó la clase pública Actividad, la cual tiene 6 atributos, 5 de ellos tipo *String* y 1 tipo *Date*: id actividad (*id*), nombre centro digital (*title*), estado actividad (*state*), fecha de creación (*initialDate*), fecha de última actualización (*finalDate*).

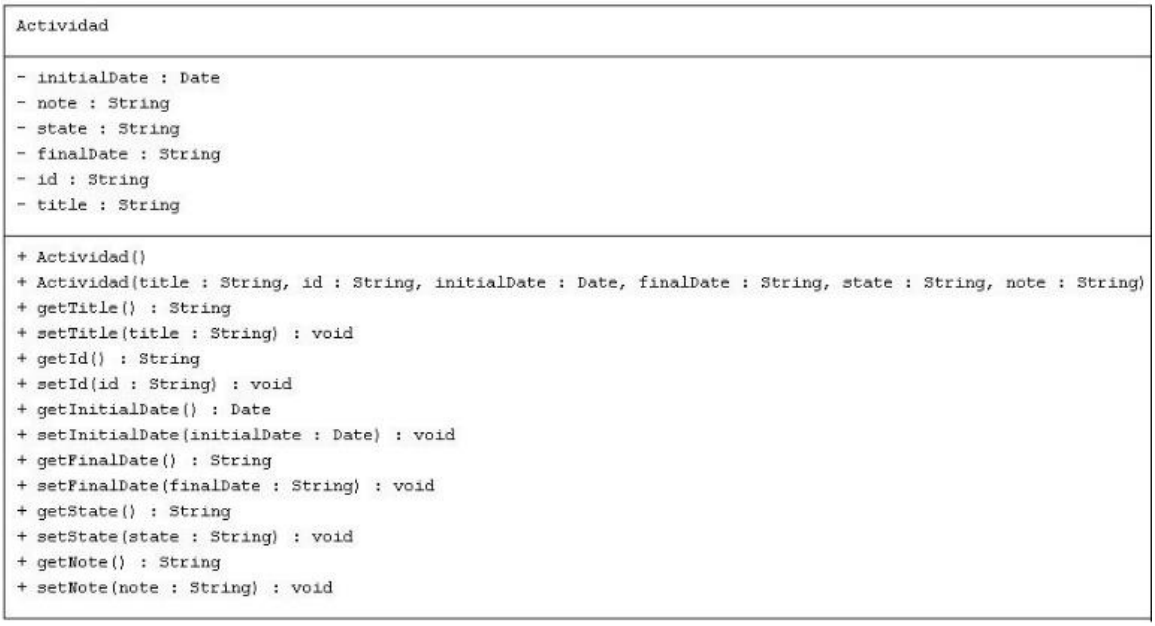


Figura 29. Diagrama UML de la clase Actividad.

El dato *initialDate* se creó como un tipo *Date,* es decir fecha, para que la base de datos lo tomara como una marca de tiempo y que cuando se realizara la consulta, se pudiera organizar tomando el orden de este dato de forma descendente por fecha y no por letras.

Como lo que se deseaba era crear una vista dinámica para cada objeto, es decir cada actividad, el elemento clave para implementar esto fue el *RecyclerView*, el cual se pobla a partir de los datos que recibe de Firestore con ayuda de un adaptador.

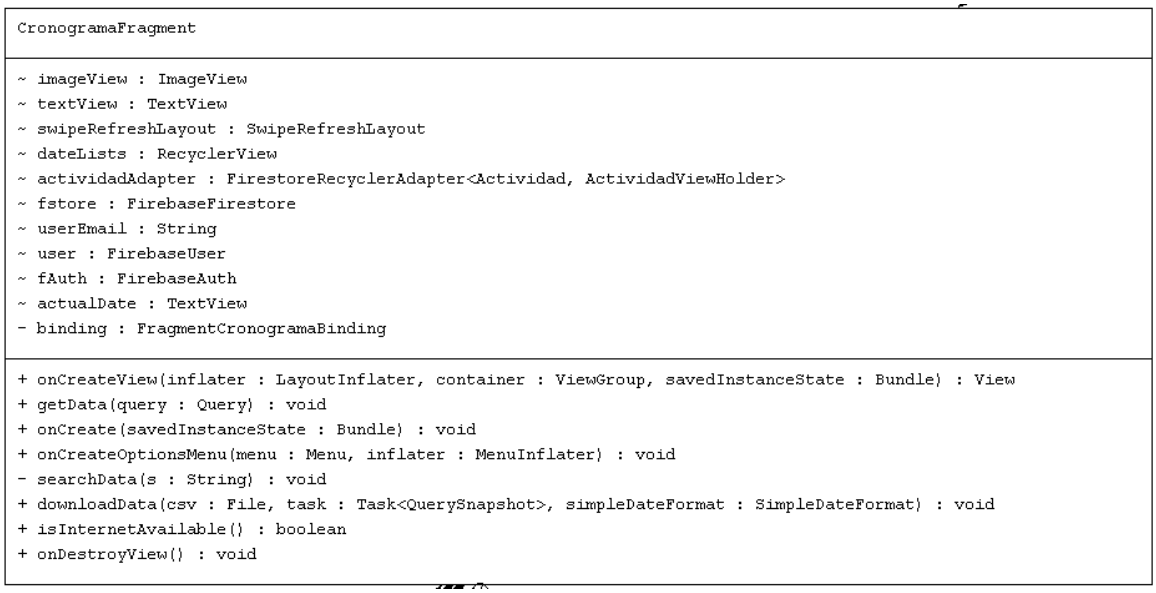


Figura 30. Diagrama UML del Fragment Cronograma.

Por otra parte, se implementó nuevamente el método creado en el *fragment* inicio, *isInternetAvailable()*, para que se mostrará un mensaje y una imagen de alerta cuando el usuario no tuviera acceso a internet y de tenerlo, que hiciera la consulta de los datos a la base de datos. Además, se implementó un *Swipe Refresh Layout* para que se pudiera refrescar la página según se requiera.

5.5.1. ADQUISICIÓN Y MUESTRA DE DATOS

Los datos se reciben en el *activity addCrono* cuando el usuario los suministra y son enviados a la base de datos, en donde la llave primaria es una clave que se le proporciona a cada usuario cuando se registra en la aplicación. De esta manera, el usuario solo tendrá acceso a sus actividades y no influirá en las actividades de otros usuarios.

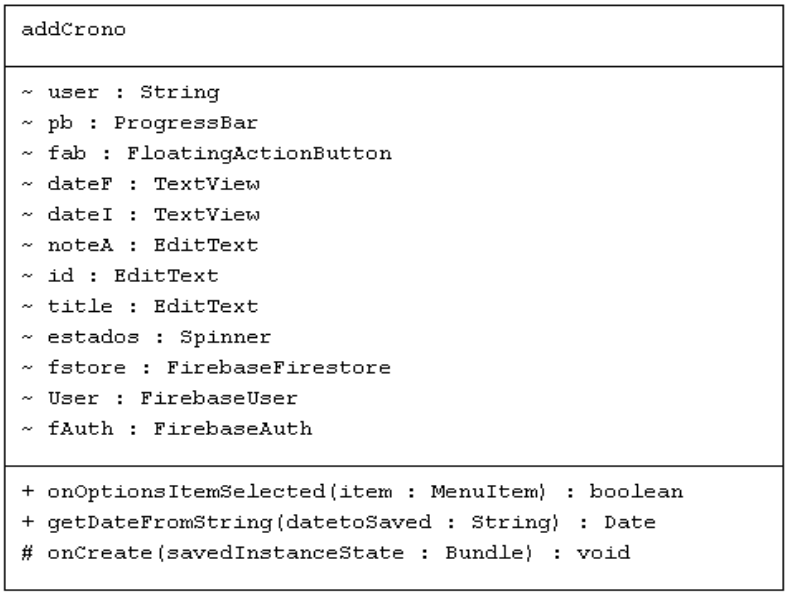


Figura 31. Diagrama UML del activity addCrono.

El *activity detallesCrono* fue creado para que usuario pudiera observar los detalles de cada actividad, además se incorporó un *Floating Action Button* para que el usuario pudiera editar la actividad de ser necesario. Los datos que recibe este *activity* son enviados desde el *fragment* principal.

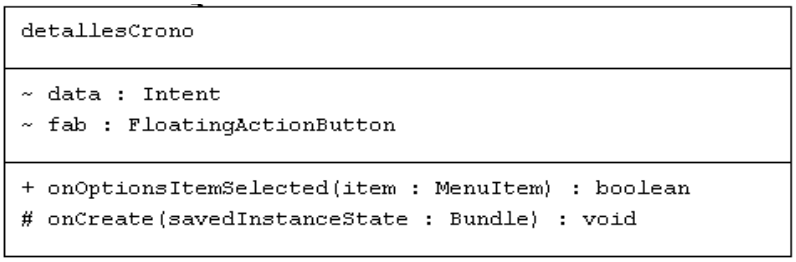


Figura 32. Diagrama UML del activity detallesCrono.

En el *activity editCrono*, el usuario puede editar los datos que añadió en *addCrono* exceptuando las fechas de inicio y última actualización, ya que la fecha de inicio se crea al momento de añadir los datos y, la fecha de la última actualización se crea por defecto al momento de abrir el *activity* en cuestión. Los datos pueden llegar al fragment de dos destinos distintos: desde el *fragment* principal o desde el *activity detallesCrono*.

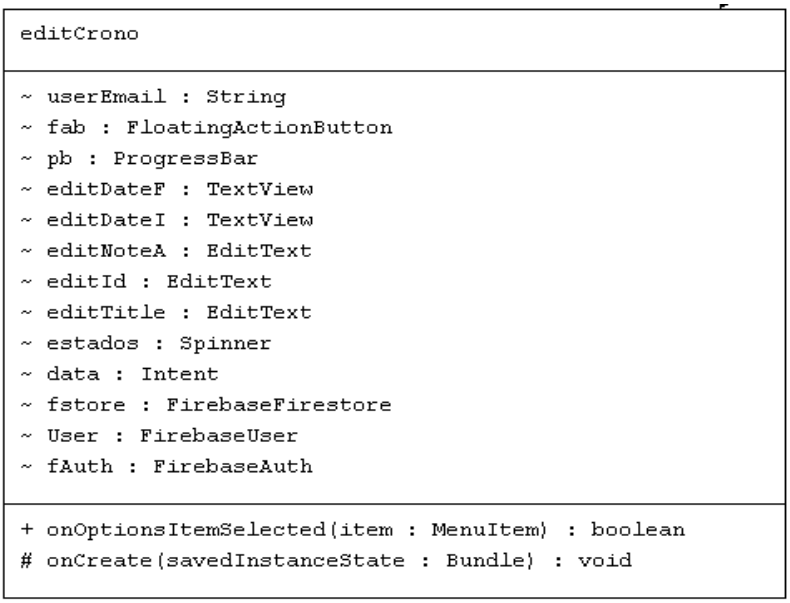


Figura 33. Diagrama UML del activity editCrono.

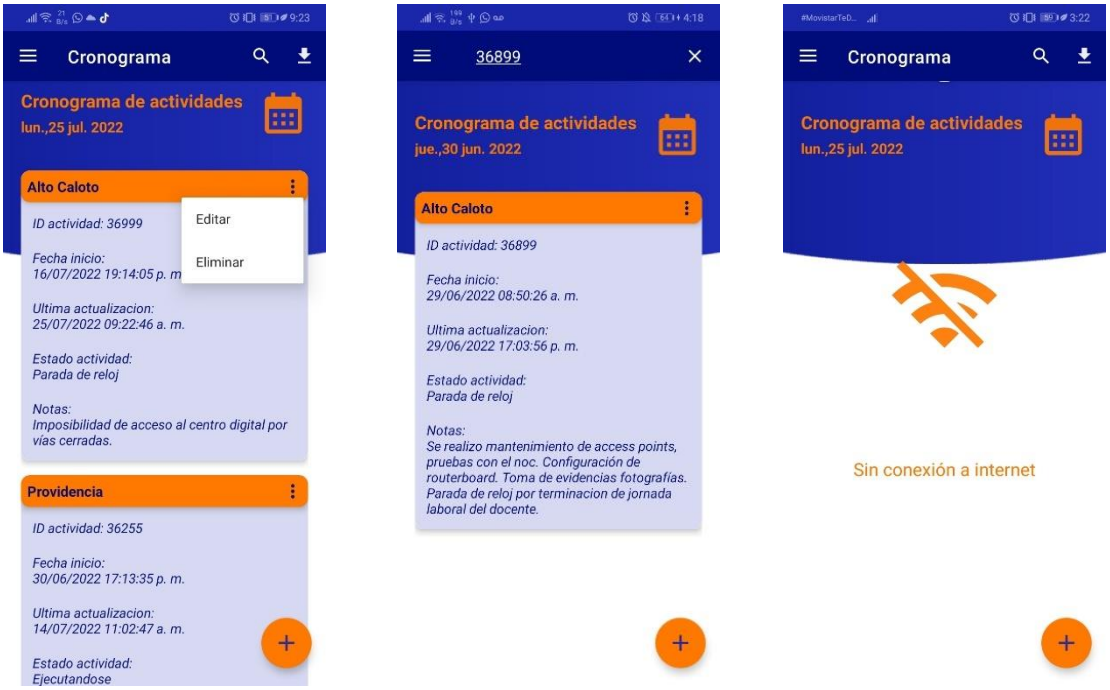


Figura 34. Interfaz: Fragment Cronograma.

Figura 35. Interfaz: Activity addCrono, detallesCrono y editCrono.

Con el fin de optimizar la vistas, en el fragment inicial únicamente se pueden ver las ultimas 30 actividades ingresadas por el usuario. Sin embargo, para poder visualizar actividades más antiguas que eso, se implementó el método *searchData(String s)* para buscar a partir del ID. Esta opción busca en toda la base de datos del usuario y entrega resultados. Además, para complementar lo anterior, se creó el método *downloadData(File csv, Task<QuerySnapshot> task, SimpleDateFormat simpleDateFormat)* para descargar todas las actividades guardándolas en un archivo *csv*, el cual tiene por nombre *cronograma-dd-mm-aaa.csv*

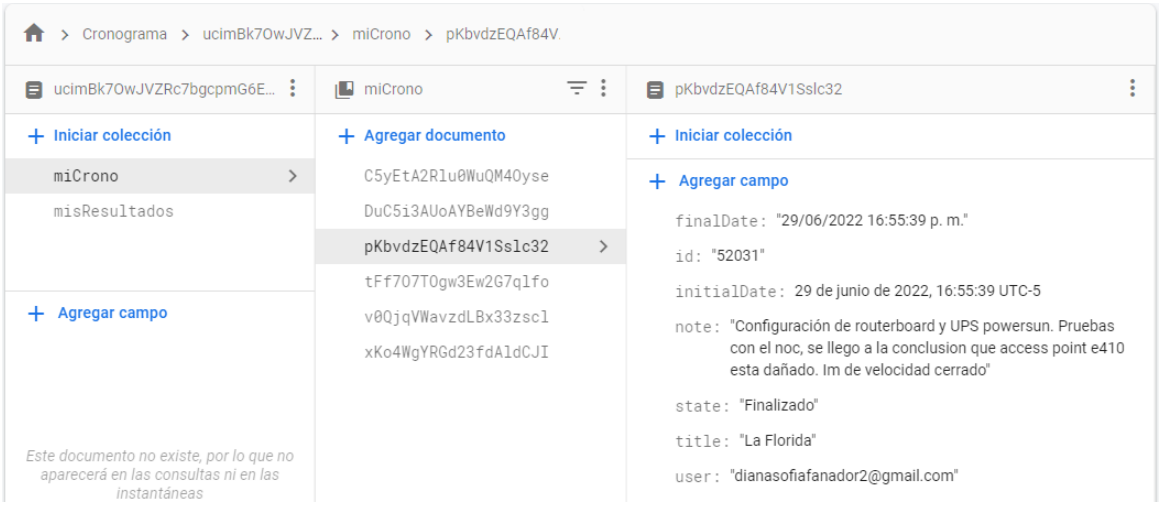


Figura 36. Cloud Firestore: colección cronograma: miCrono.

5.6. ANALISIS DE PRODUCTIVIDAD

El propósito de este requisito es que los usuarios tengan acceso a los resultados de su proceso en la empresa y que esto los incentive a mejorar. Para esto es importante tener en cuenta que diariamente las actividades que se realizan en campo tienen 3 estados: ejecutándose, que se refiere a la actividad que se encuentra realizando en el día; paradas de reloj, es decir, que el usuario debe detener la actividad por una fuerza mayor y el segundo estado es finalizado, significando que la actividad ha sido terminada con éxito.

Para lo anterior, se implementó un diagrama de Pie usando la librería *MPAndroidChart*, el cual reúne los datos de la colección de cronogramas de cada usuario y sacará el porcentaje de cada estado para finalmente mostrarlo. Para obtener los datos, se realiza una consulta de la base de datos y los almacena en 3 listas diferentes dependiente del estado: *dataListE*, *dataListP* y *dataListF*. En este diagrama únicamente se muestra lo que el usuario ha realizado durante el mes corriente, si el usuario no ha ejecutado ninguna tarea entonces no se mostrará diagrama.

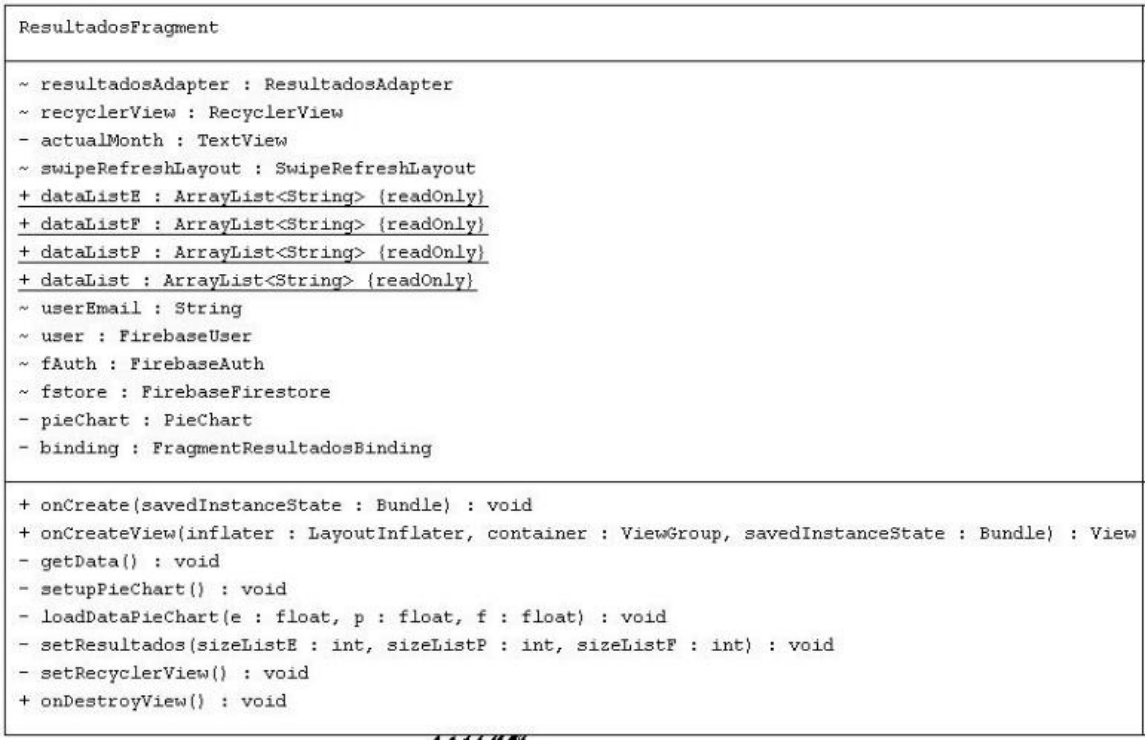


Figura 37. Diagrama UML del Fragment Resultados.

Por consiguiente, surgió la necesidad de implementar una manera en que el usuario pudiera acceder a los resultados de los meses anteriores y pudiera ver su progreso. Por lo que fue necesario crear una clase publica *Resultado*, la cual tiene

5 atributos tipo String: cantidad de actividades en estado ejecutándose (*stateE*), cantidad de actividades en estado paradas de reloj *(stateP*), cantidad de actividades en estado finalizado (*stateF*), mes actual (*month*) y año actual (*year*).

La relación anterior se almacena en una nueva subcoleccion llamada *misResultados*, el ultimo día del mes corriente y se puede observar en la aplicación a partir de ese día. Para evitar que hubiera dos resultados del mismo mes, se realiza una consulta del mes y año actual en la subcolección en los campos *month* y *year,* si no hay valores en la consulta entonces se envían los tamaños de las listas *dataListE*, *dataListP* y *dataListF,* junto con el mes y año actual.

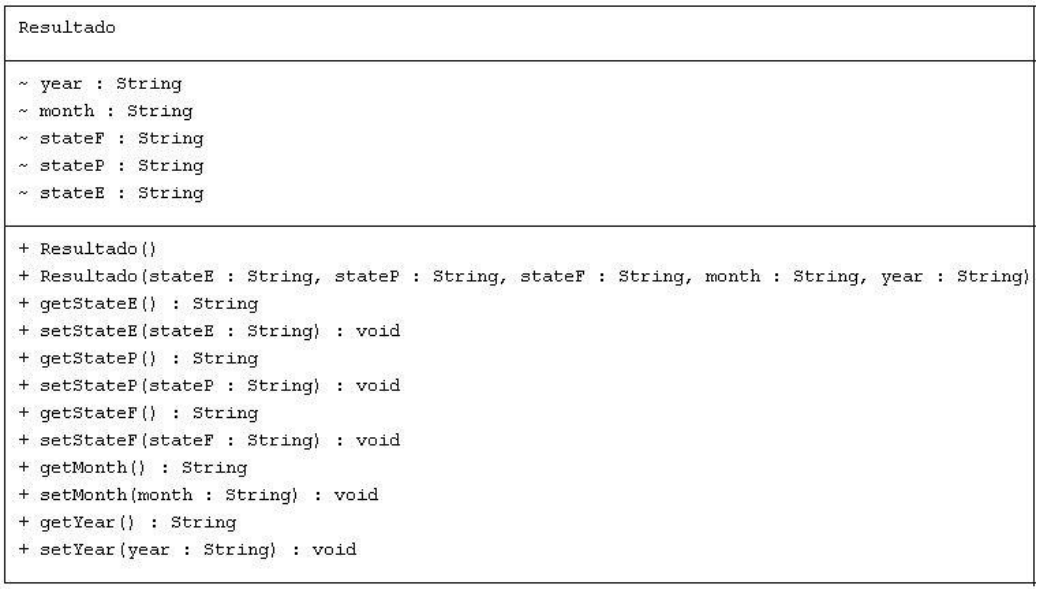


Figura 38. Diagrama UML de la clase publica Resultado.

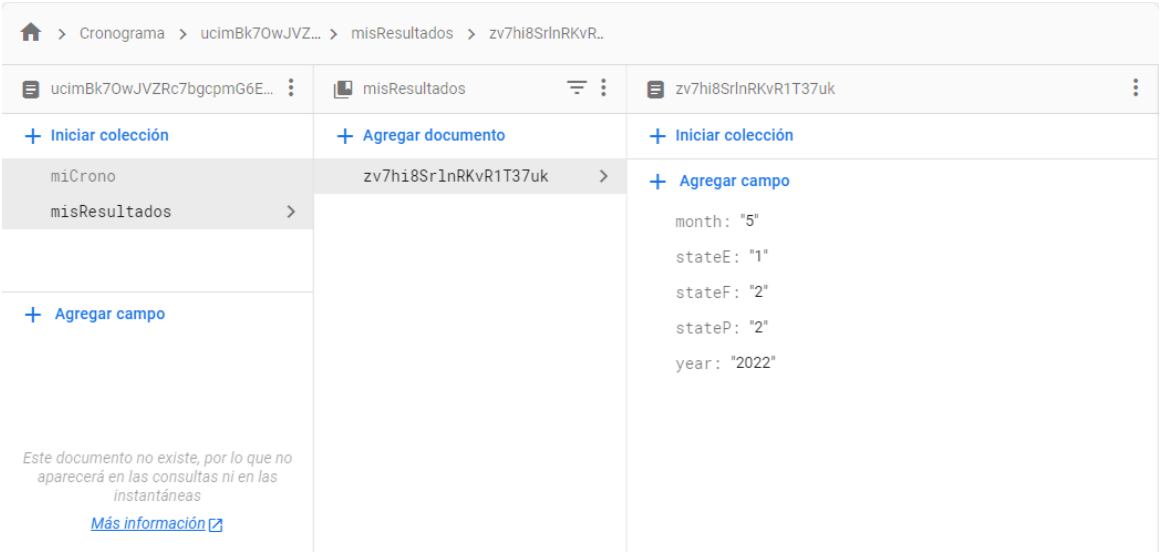


Figura 39. Cloud Firestore: colección cronograma: misResultados.

Como se trataba de implementar una vista dinámica para el objeto *Resultado*, se utilizó nuevamente un RecyclerView y se implementó un adaptador para poblarlo teniendo en cuenta los datos guardados en Firestore.

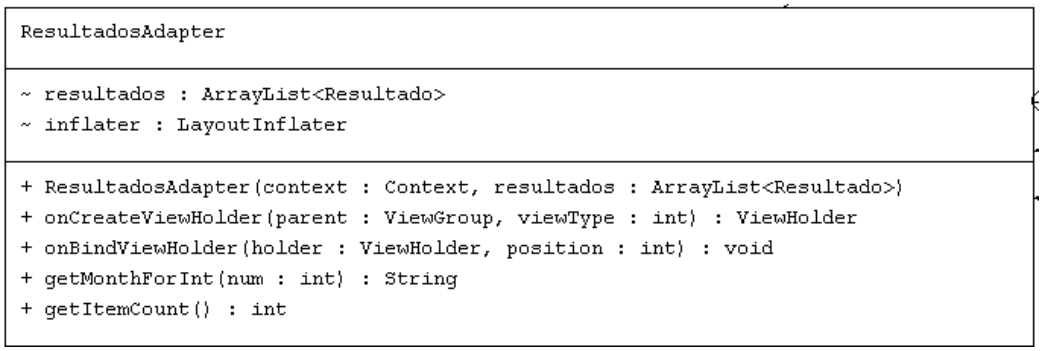


Figura 40. Diagrama UML del adaptador del recycler view de los resultados.

Figura 41. Interfaz: Fragment Resultados.

5.7. FORO INTERACTIVO

El propósito de este requisito es permitir que los usuarios puedan plantear preguntas en torno a la configuración de los equipos e intercambiar ideas al respecto, incluso que los usuarios puedan compartir sus habilidades y experiencias, de manera que todos estén informados y que no siempre se recurra al técnico de soporte, para solucionar un problema que es muy probable, que un usuario sepa como solucionarlo.

Para implementar este requisito, fue necesario crear un *fragment Foro*, que sería la vista principal y estaría unida a *Nativigation Drawer Activity*, y otros tres *activitys* secundarios para editar, crear y mostrar cada pregunta del foro. Aquí nuevamente, se utilizó la base de datos *Cloud Firestore* y se almaceno la información en la colección *Foro*.

Teniendo en cuenta lo anterior, se crearon 2 clases públicas Pregunta y Respuesta, igualmente se implementaron 2 *RecyclerView* junto con sus adaptadores, para poblarlos con las preguntas y sus respectivas respuestas.

La clase Pregunta tiene 6 atributos, 5 de ellos tipo *String* y 1 tipo *Date:* fecha de creación de la pregunta (*dateP*), fecha de edición de la pregunta (*dateE*), titulo de la pregunta (*title*), descripción de la pegunta (*description*), usuario que realizó la pregunta (*userP*) y categoría de la pregunta (*categoría*). El campo *dateP* fue el que se tomo como tipo *Date* para organizar las preguntas por marca de tiempo en orden descendente.

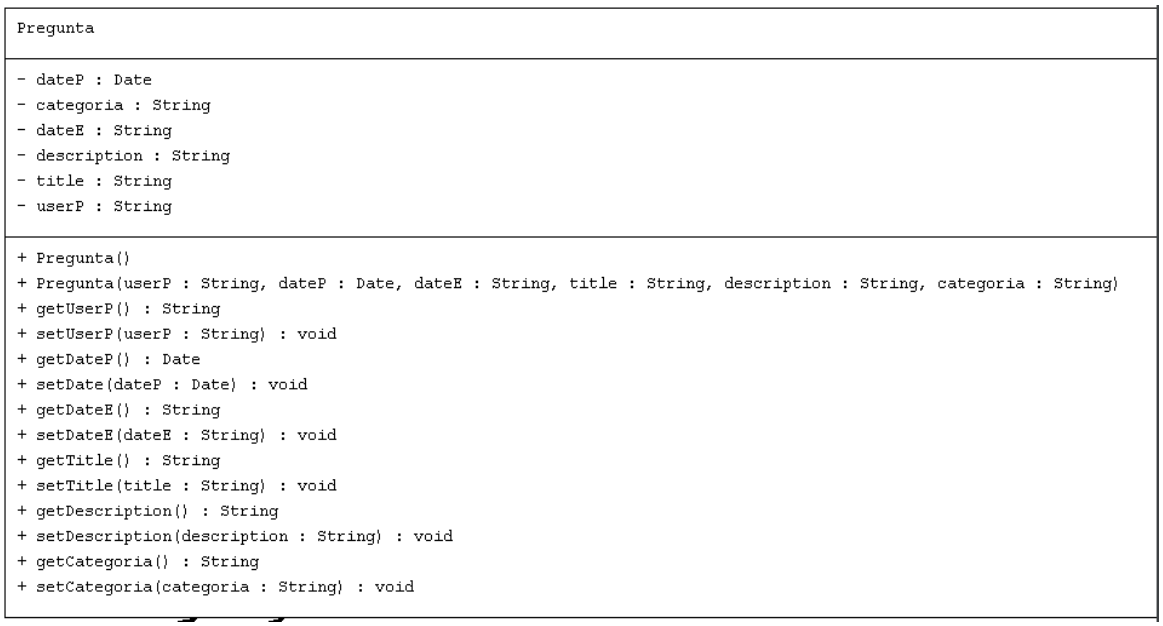


Figura 42. Diagrama UML de la clase pública Pregunta.

La clase Respuesta tiene 3 atributos, 2 de ellos tipo *String* y 1 tipo *Date:* fecha de creación de la respuesta (*dateR*), descripción de la pegunta (*descriptionR*) y usuario que realizó la pregunta (*userR*). El campo *dateR* fue el que se tomó como tipo *Date* para organizar las preguntas por marca de tiempo en orden descendente.

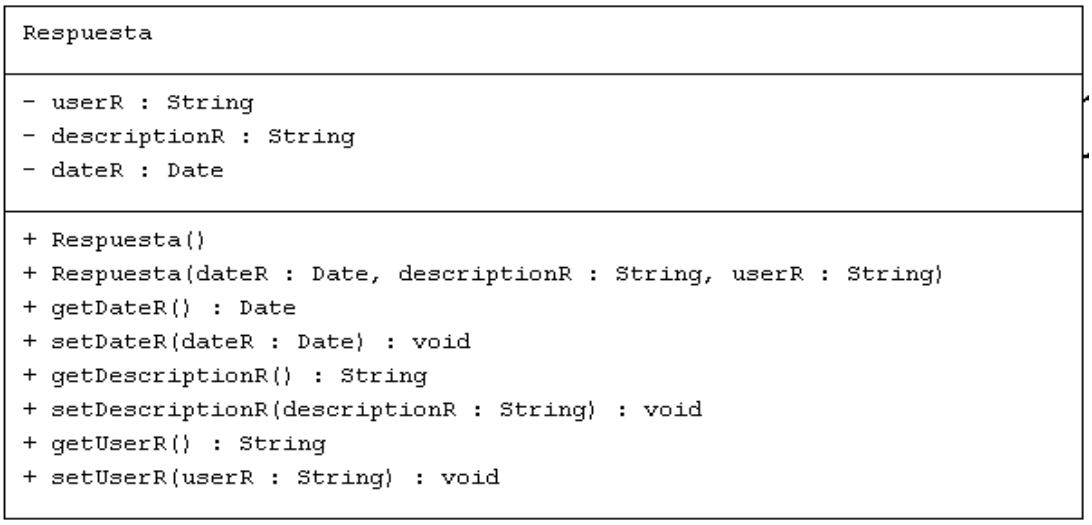


Figura 43. Diagrama UML de la clase pública Respuesta.

Con el fin de optimizar la vistas, en el fragment principal únicamente se pueden ver las ultimas 50 preguntas ingresadas por el usuario. Para facilitar la búsqueda de las preguntas, se implementó el método *setFilter(String categoria)* el cual se encargaría de realizar una consulta en la base de datos y buscaría las preguntas según categoría. Las preguntas, entonces, tendrían 6 tipos de categorías: configuración antena satelital, radio, router, access point, UPS y otro.

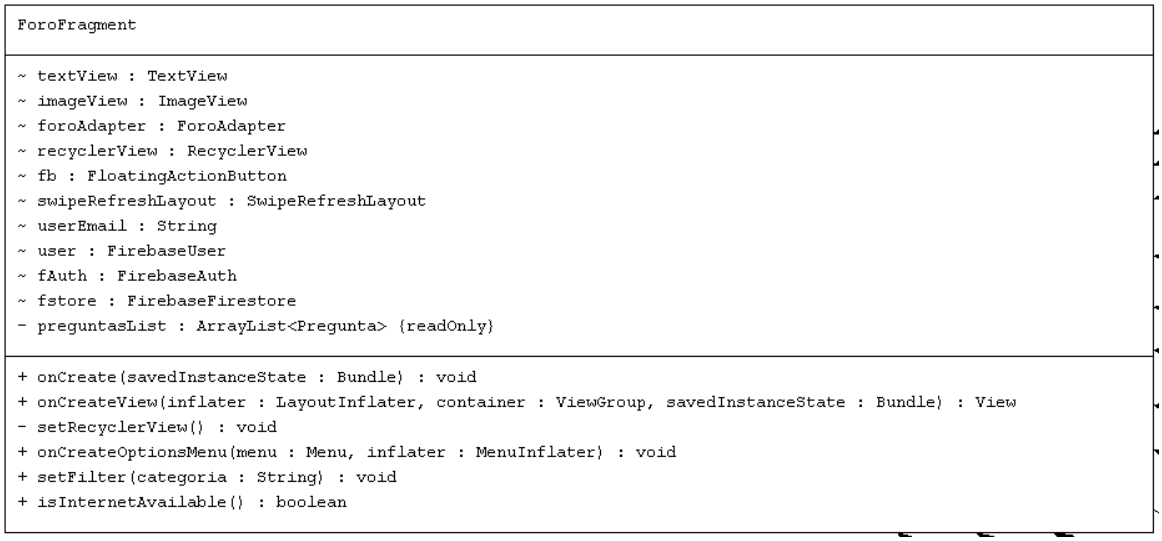


Figura 44. Diagrama UML del fragment Foro.

Cuando el usuario es quien realizó la pregunta, el titulo dentro de la vista en el fragment principal, aparecerá en color naranja y tendrá la oportunidad de editar o eliminar la pregunta si lo ve necesario. Los datos como las fechas no se pueden editar ya que se manejan automáticamente a través del código fuente.

5.7.1. ADQUISICIÓN Y MUESTRA DE DATOS

Los datos se reciben en el *activity addForoP* cuando el usuario los suministra y son enviados a la base de datos, para que todos los usuarios tengan acceso a las preguntas y respuestas.

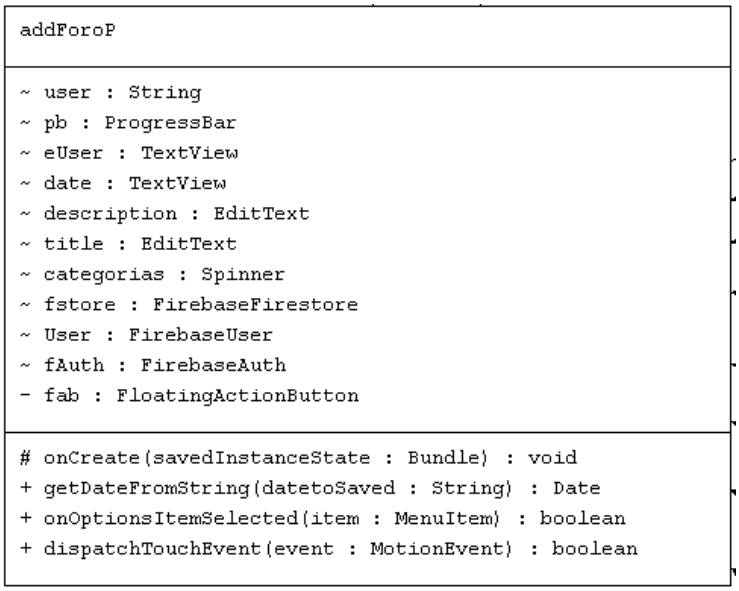


Figura 45. Diagrama UML del activity addForoP.

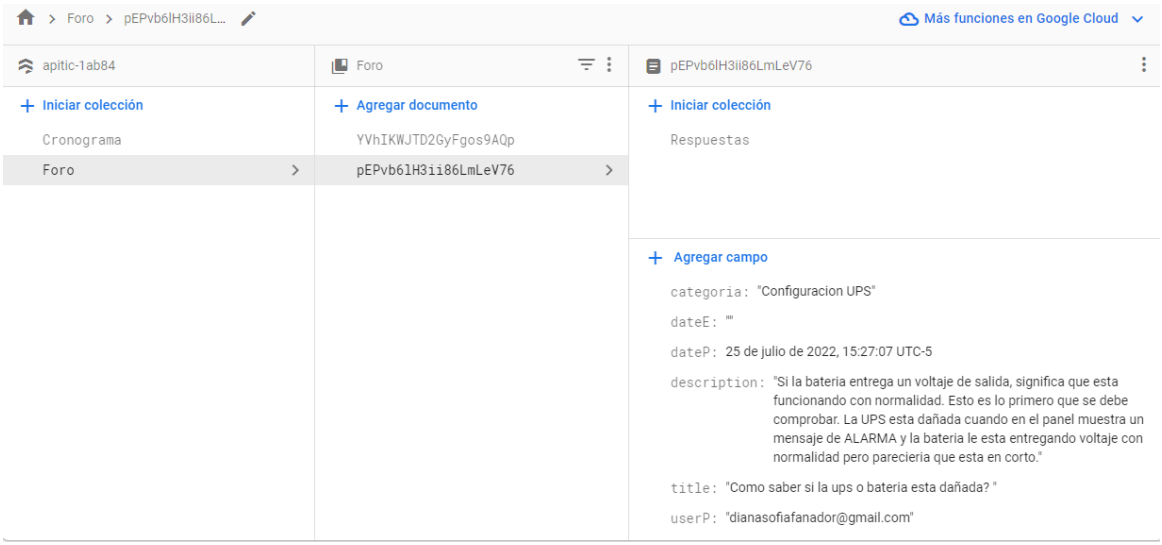


Figura 46. Cloud Firestore: colección Foro.

En el *activity detallesForo* se reciben los datos que son enviados desde el fragment principal al momento de iniciar el activity para que se muestren al usuario, además es donde se encuentra ubicado el *RecyclerView* de respuestas. En este activity, el usuario puede responder la pregunta, la cual se almacenará dentro de la ubicación de la pregunta creando una subcolección *Respuestas.*

Cuando el usuario es quien realizó la respuesta, el nombre del usuario aparecerá en color naranja y podrá editarla o eliminarla. Para la edición de la respuesta, no se hizo necesario implementar otro fragment porque era un desgaste de recursos, por lo que en la caja del *EditText* donde se escriben las respuestas, se retornará la descripción desde la base de datos y el usuario podrá editarla. Por otra parte, el dato de fecha de edición de la pregunta solo aparecerá en este activity cuando ésta sea editada.

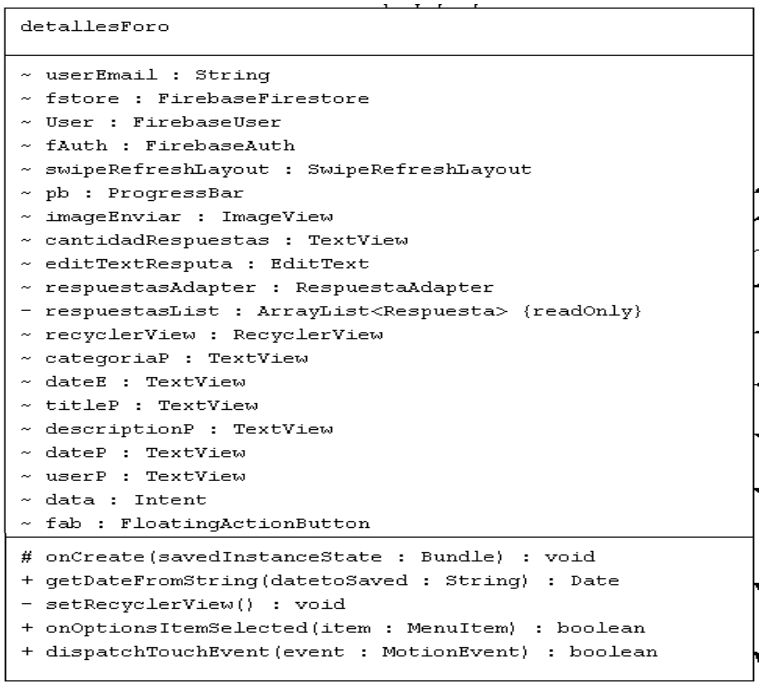


Figura 47. Diagrama UML del activity detallesForo.

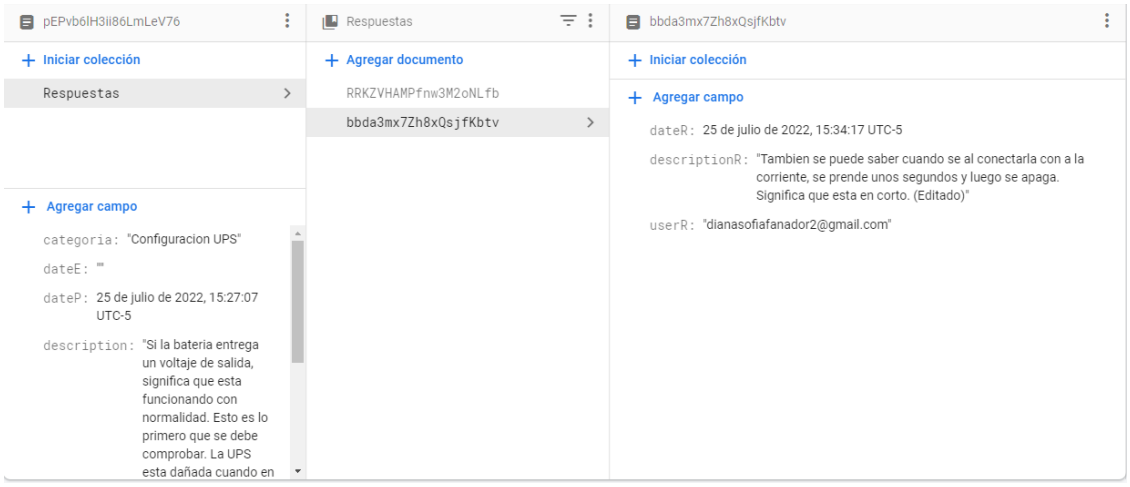


Figura 48. Cloud Firestore: colección Foro: Respuestas.

En el *activity editForoP* se reciben los datos que se envían desde el fragment Foro principal, aquí se pueden editar los datos como el título, descripción y categoria

de la pregunta. Así mismo, al momento de guardar la información, se actualizará la fecha de edición. En caso de que el usuario quiera cancelar la edición, se enviarán los datos de la pregunta al activity *detallesForo* y se ejecutará ese activity.

Las respuestas de la pregunta que está siendo editada, también se mostrarán en la vista de este activity pero no podrán editarse ni eliminarse, tampoco se tendrá la opción de responder.

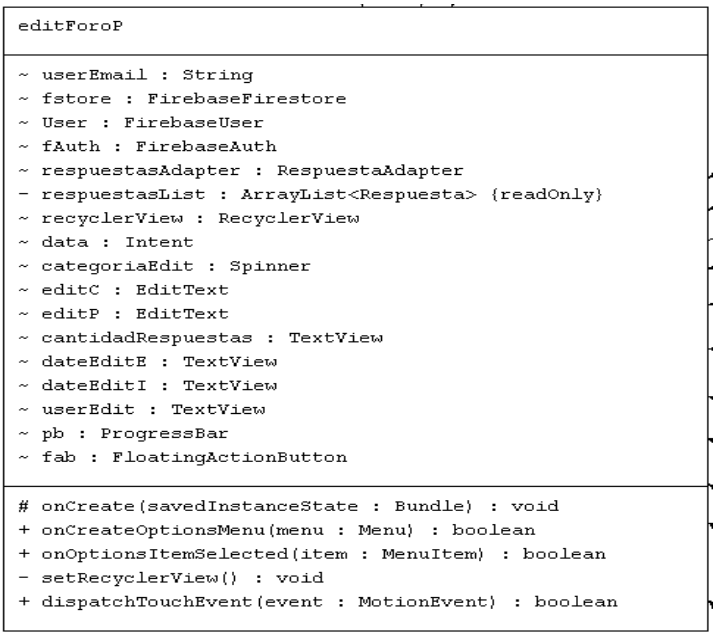


Figura 49. Diagrama UML del activity editForoP.

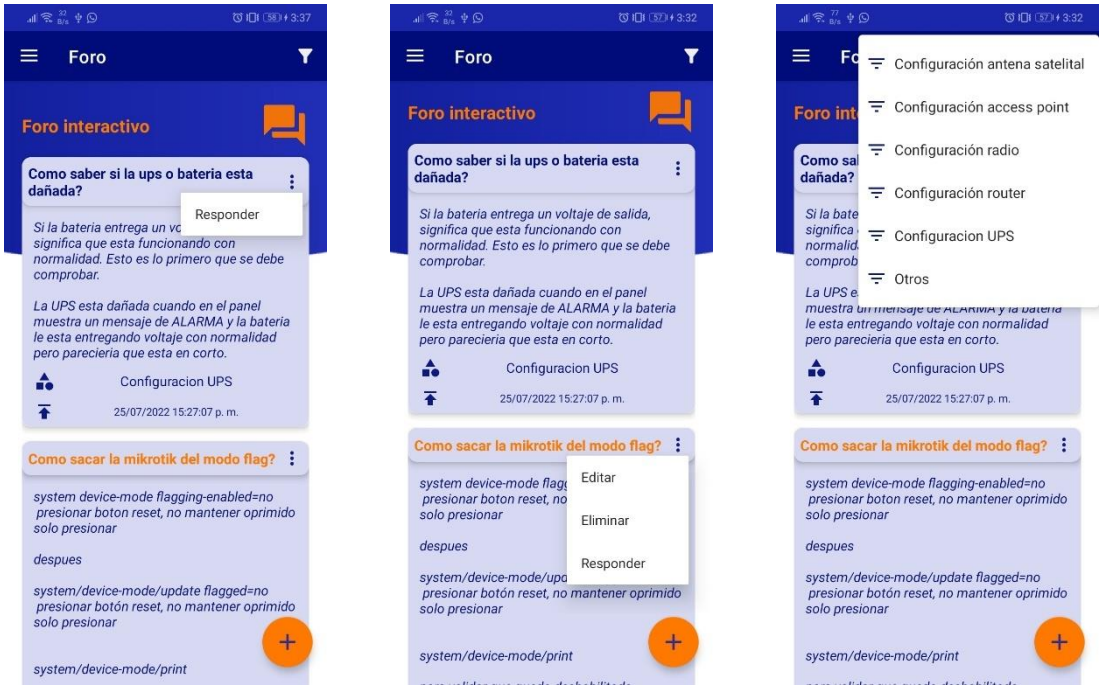


Figura 50. Interfaz: fragment Foro.

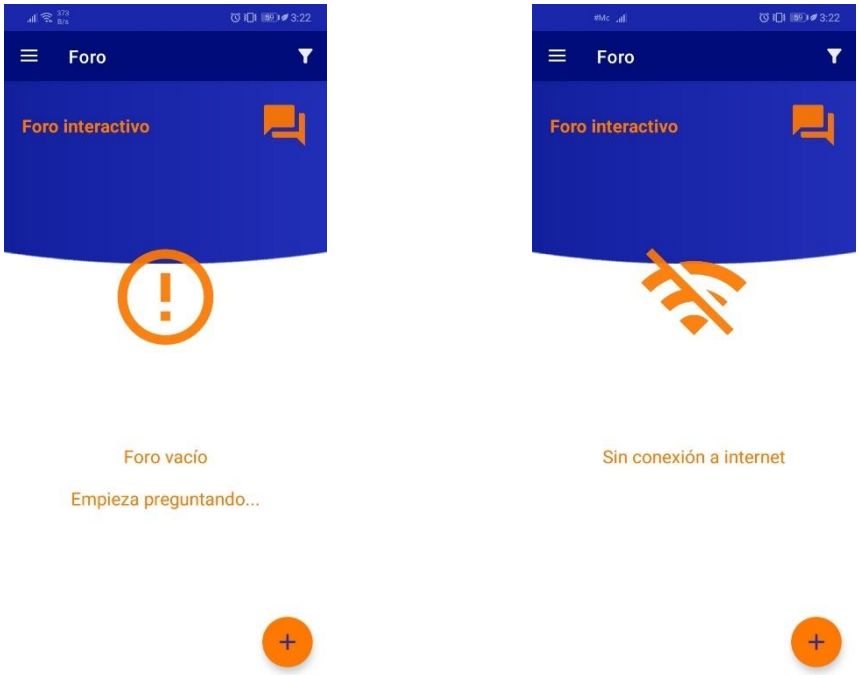


Figura 51. Interfaz: fragment Foro en estado vacío y sin conexión.

Figura 52. Interfaz: activity addForoP y editForoP.

En los fragment principales como Cronograma, Resultados y Foro se implementó el *Swipe Refresh Layout* para actualizar los datos y el método *isInternetAvalaible()* para que anunciar al usuario cuando la aplicación no detecta que haya una conexión a internet. En el activity detallesForo, también se puede refrescar la página.

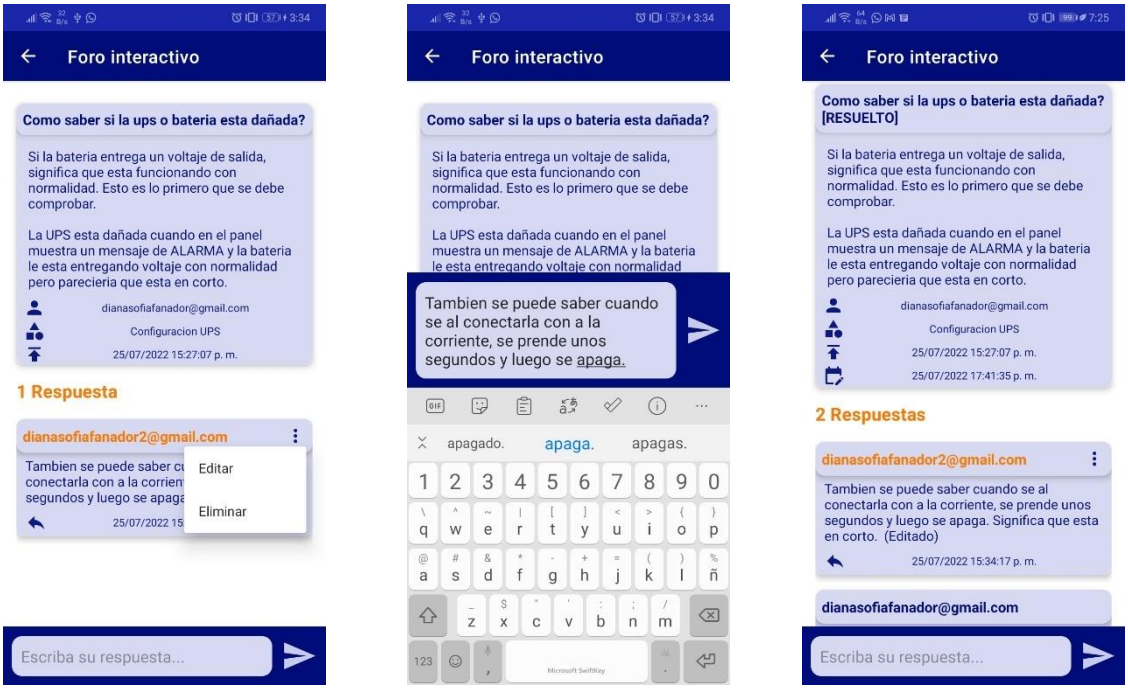


Figura 53. Interfaz: activity detallesForo edición de respuesta.

6. CONCLUSIONES

La herramienta de desarrollo móvil APITIC logró cumplir con los requerimientos planteados, englobando los servicios que el usuario necesita para brindar soluciones a reprocesos que impactan negativamente la productividad del ciclo de trabajo. Por otro lado, esta herramienta permite considerar nuevas rutas de aprendizaje, no solo en proyectos de ingeniería, sino en cualquier tipo de proyecto.

El desarrollo frontend, es decir la interfaz del usuario desde el punto de vista del código, fue implementado en Android Studio y utilizó el lenguaje de programación Java. Se garantizó que el usuario tuviera una experiencia final amigable y que el diseño fuera visualmente atractivo, de manera que los usuarios pudieran disfrutar del uso de la aplicación.

Para el desarrollo backend, que se refiere a la programación que no es visible para el usuario y se ejecuta en el servidor, se utilizaron los servicios de la herramienta Firebase, por lo que no fue necesario establecer una lógica del lado del servidor ya que todo se manejó desde Java, es decir el dispositivo del cliente, brindando una alternativa bastante llamativa al desarrollo backend tradicional.

Con el fin de proporcionar un apoyo al proyecto Claro Mintic 7K, se diseñaron e implementaron cuatro requisitos del sistema de la aplicación móvil, guiados a suplir esta necesidad: el primero es el acceso a los manuales de configuración de los equipos que se instalan en los centros digitales; el segundo es la creación de un cronograma de actividades, que debe alimentar diariamente el usuario final y que es posible descargarlo en un archivo *csv*, para que el personal en oficina haga sus respectivas verificaciones y utilice la información para evitar reprocesos; el tercero es un control de la productividad de las actividades realizadas, para que el usuario pueda autoevaluarse; y por último, es la implementación de un foro interactivo, para que los usuarios compartan sus ideas y soluciones, a las preguntas más frecuentes sobre la configuración de los equipos.

La experiencia de aprendizaje se desenvolvió en dos áreas muy diferentes de la ingeniería, pero que se complementan, como lo son las telecomunicaciones y el desarrollo de software. Durante este proceso, se identificaron las necesidades que surgen para instalar y mantener el servicio de internet en zonas tan remotas, como por ejemplo los resguardos indígenas. Una de las dificultades más frecuentes es que ocurren quema de equipos, debido a la inestabilidad de la fluido eléctrico y las descargas eléctricas naturales. En muchos de los casos, la tierra no es la más propicia para implementar un sistema de puesta a tierra seguro y potente.

7. RECOMENDACIONES

Para el uso de la aplicación es importante tener en cuenta que el plan escogido para Firebase, es el plan Spark, el cual tiene unos límites de transferencia y operaciones. Por ejemplo, Cloud Firestore que es el servicio de base de datos usado para el cronograma, resultados y foro, permite 50k operaciones de lectura, 20k operaciones de escritura y 20k eliminaciones de documentos al día.

Por otra parte, Realtime Database que es el otro servicio de base de datos usado para guardar la información del usuario, permite 100 conexiones simultaneas al día, 1GB de almacenamiento y 10 GB de descarga al mes.

Finalmente, para el Cloud Storage, que es donde se almacenan las fotos de los usuarios y los manuales de configuración, permite 5 GB de almacenamiento, 20k operaciones de carga y 50k operaciones de descarga al día. Lo anterior son casi 2.5k fotos de alta resolución.

El área TIC de la empresa Ingelec Group S.A.S. cuenta con 38 técnicos operativos, de cuales 22 se dedican a realizar mantenimientos en los centros digitales y son el público al que va dirigido la aplicación principalmente. Inicialmente, no habrá ningún problema para el almacenamiento pero es probable que después de un tiempo toque cambiarse al plan Blaze de Firebase, el cual cobra un pequeña cantidad en dólares, después de que se pasa la cantidad de operaciones y transferencias mencionadas con anterioridad.

Por último, todos los correos que envía Firebase para la autenticación del usuario y cambio de contraseña, llegan al correo SPAM.

8. BIBLIOGRAFÍA

Academy, S. (15 de enero de 2020). *YouTube.* Recuperado el 17 de marzo de 2022, de https://www.youtube.com/watch?v=Wf7DDIaRYjk&list=RDCMUCR1t5eSmLxLUdBnK2XwZOuw&index=6&ab_channel=SmallAcademy

Academy, S. (20 de febrero de 2022). *YouTube.* Recuperado el 28 de marzo de 2022, de https://www.youtube.com/watch?v=UzjXf1ehjN8&ab_channel=SmallAcademy

Codebun. (2022). *Codebun.* Recuperado el 28 de marzo de 2022, de https://codebun.com/android-table-layout-with-recycler-view/

Coding), B. K. (1 de marzo de 2017). *YouTube*. Recuperado el 12 de febrero de 2022, de https://www.youtube.com/watch?v=EM2x33g4syY&list=RDCMUC9YTuDeKzDoyOphWHtdK0jA&index=5&ab_channel=SimplifiedCoding

Coding), B. K. (1 de marzo de 2017). *YouTube*. Recuperado el 23 de febrero de 2022, de https://www.youtube.com/watch?v=jEmq1B1gveM&list=RDCMUC9YTuDeKzDoyOphWHtdK0jA&index=5&ab_channel=SimplifiedCoding

Coding), B. K. (17 de abril de 2018). *YouTube*. Recuperado el 26 de febrero de 2022, de https://www.youtube.com/watch?v=WeoryL3XyA4&ab_channel=SimplifiedCoding

Firebase. (2022). *Firebase.* Recuperado el 5 de marzo de 2022, de https://firebase.google.com/docs/storage/android/download-files?hl=es-419

Firebase. (2022). *Firebase.* Recuperado el 25 de marzo de 2022, de https://firebaseopensource.com/projects/firebase/firebaseui-android/firestore/readme/

Gonzalez, M. S. (5 de noviembre de 2012). *Redes Telematicas.* Recuperado el 20 de julio de 2022, de https://redestelematicas.com/la-ultima-milla/

HughesNet. (2020). *HughesNet*. Recuperado el 20 de julio de 2022, de https://www.hughes.com/sites/hughes.com/files/2022-03/HT2010-Router.pdf

Ingenieros, A. (2018). *AC+CC Ingenieros*. Recuperado el 19 de julio de 2022, de https://www.ac-cc.com/blog/en-que-consiste-un-sistema-de-energia-de-respaldo-o-de-emergencia

Kracik, P. (9 de junio de 2020). *Medium.* Recuperado el 5 de marzo de 2022, de https://medium.com/firebase-developers/firebase-how-to-access-and-download-files-in-cloud-storage-b8f2cf49aa13

Martinez, J. L. (15 de diciembre de 2017). *PRORED*. Recuperado el 20 de julio de 2022, de https://www.prored.es/que-es-un-radioenlace/

Matur, N. (30 de mayo de 2018). *Packt.* Recuperado el 18 de julio de 2022, de https://hub.packtpub.com/android-studio-how-does-it-differ-from-other-ides/

Munje, C. (21 de diciembre de 2020). *GeeksforGeeks.* Recuperado el 16 de marzo de 2022, de https://www.geeksforgeeks.org/how-to-load-pdf-from-url-in-android/

Networks, C. (2018). *Cambium Networks.* Recuperado el 20 de julio de 2022, de https://www.cambiumnetworks.com/wp-content/uploads/2018/10/SS_ePMP3000_SectorAntenna_10032018_bleed.pdf

Networks, C. (2022). *Cambium Networks*. Recuperado el 19 de julio de 2022, de https://www.cambiumnetworks.com/products/wifi/cnpilot-e510-wifi-access-point/

Networks, C. (2022). *Cambium Networks*. Recuperado el 19 de julio de 2022, de https://www.cambiumnetworks.com/products/wifi/cnpilot-e410-wifi-access-point/

Pervaiz, A. (11 de diciembre de 2018). *YouTube*. Recuperado el 06 de abril de 2022, de https://www.youtube.com/watch?v=GkcQWsojLFE&ab_channel=AtifPervaiz

Prabhu, R. (11 de julio de 2022). *Geeks for Geeks*. Recuperado el 18 de julio de 2022, de https://www.geeksforgeeks.org/object-oriented-programming-oops-concept-in-java/

Shah, K. (5 de julio de 2012). *Stackoverflow.* Recuperado el 12 de junio de 2022, de https://stackoverflow.com/questions/11341931/how-to-create-a-csv-on-android

Stevenson, D. (24 de septiembre de 2018). *Medium*. Recuperado el 18 de julio de 2022, de https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0

Trebilcox-Ruiz, P. (21 de novimebre de 2016). *Envatotuts+.* Recuperado el 3 de marzo de 2022, de https://code.tutsplus.com/es/tutorials/firebase-for-android-file-storage--cms-27376

Uzelac, V. (1 de junio de 2018). *Stackoverflow.* Recuperado el 11 de marzo de 2022, de https://stackoverflow.com/questions/50649460/android-studio-firebase-query-search-returning-object

Wick, J. (3 de julio de 2015). *Stackoverflow.* Recuperado el 18 de junio de 2022 , de https://stackoverflow.com/questions/31204320/how-can-i-change-the-navigationviews-item-text-size

Y, D. (23 de abril de 2013). *Stackoverflow.* Recuperado el 12 de marzo de 2022, de https://stackoverflow.com/questions/16177191/bitmap-compressbitmap-compressformat-png-0-fout-making-image-size-bigger-tha

Yeung, W. (9 de septiembre de 2020). *Medium.* Recuperado el 19 de abril de 2022, de https://medium.com/@clyeung0714/using-mpandroidchart-for-android-application-piechart-123d62d4ddc0