# ESCUELA POLITÉCNICA NACIONAL

## FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

### APLICACIONES IOT (INTERNET OF THINGS)

### DESARROLLO DE UNA APLICACIÓN MÓVIL QUE PERMITA EL MONITOREO DEL COMPORTAMIENTO DE UN CIUDADANO EN SUS ACTIVIDADES DIARIAS, DENTRO DE UN ÁREA DE LA CIUDAD DE QUITO, UTILIZANDO APLICACIONES IOT

**TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCION DEL TITULO DE INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN**

**JORGE DANIEL VILLAVICENCIO SANDOVAL**

**DIRECTOR: SORAYA LUCÍA SINCHE MAITA, PhD.**

**DMQ, Agosto 2023**

# CERTIFICACIONES

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

**Jorge Daniel Villavicencio Sandoval**

Certifico que el presente trabajo de integración curricular fue desarrollado por Jorge Daniel Villavicencio Sandoval, bajo mi supervisión.

**Soraya Lucía Sinche Maita, PhD.**
**DIRECTOR**

# DECLARACIÓN DE AUTORÍA

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

JORGE DANIEL VILLAVICENCIO SANDOVAL

SORAYA LUCIA SINCHE MAITA, PhD

# DEDICATORIA

A mi madre y a todas las personas que confiaron en mí.

***Jorge Daniel Villavicencio Sandoval***

# AGRADECIMIENTO

Agradezco muy profundamente a mi pilar fundamental, mi madre Florita sin su bendición y duro trabajo no sería posible esta parte de mi vida. Gracias por su infinito cariño, incansable fortaleza y por ser mi motivación para mejorar día a día. Este y mis futuros logros son el producto de todo su esfuerzo, dedicación y la confianza que supo brindarme.

A mis hermanas Lucia, Tania, mi hermano Roberto y su esposa Andrea por acompañarme en parte de este proceso, a mis hermosas sobrinas, mis motivaciones para no apartarme del camino.

A mis tías Gloria, Rosita y mi primo Gustavo por apoyarme e incentivarme en mi proceso personal y educativo. A mis tíos Pedrito y Yolita gracias por acogerme en su hogar, escuchar mis problemas y aconsejarme en mis momentos más complicados.

A Santiago, Jessenia, Doménica, Julio y Alexandra, la maravillosa familia que conocí al principio de mi vida educativa con los cuales hemos formado un vínculo fraternal y de los cuales me llevo las más gratas experiencias, sentimientos y recuerdos. Gracias a ellos y a sus familias por abrirme las puertas de sus hogares y recibirme como un miembro más.

A mis amigos Joel, Kevin y Erika por permitirme compartir grandes experiencias en tan poco tiempo.

A la Escuela Politécnica Nacional, lugar en el cual aprendí a superar desafíos y proponerme nuevas metas, a su personal docente en especial a la PhD Soraya Sinche por depositar en mi la confianza y paciencia no solo en el desarrollo de este trabajo si no también en el transcurso de mi carrera.

Al camino que me trajo hasta aquí ya que a veces fue largo y difícil, pero me alegra que haya sido así, porque si no hubiese sido complicado llegar hasta aquí, la lección no hubiese sido tan clara.

# ÍNDICE DE CONTENIDOS

# ÍNDICE DE FIGURAS

# ÍNDICE DE TABLAS

# RESUMEN

En el contexto de una ciudad inteligente y sustentable la información proporcionada por los ciudadanos juega un papel fundamental para determinar los servicios, infraestructura y actividades de interés más importantes para cada uno de ellos. Además, esta información recolectada puede resultar útil tanto para el usuario que desea conocer sus actividades diarias, así como él se relaciona con los espacios públicos a su disposición.

Debido a esto el siguiente trabajo de integración curricular presenta el desarrollo de un prototipo de aplicación móvil, para sistema operativo Android, encargado de recolectar información de interés para el monitoreo ciudadano. Este prototipo adquiere y almacena información de las actividades diarias del ciudadano tales como número de pasos, tiempo de sueño y tiempo que el ciudadano pasa en áreas verdes.

En el primer capítulo se expone los conceptos relacionados a ciudades inteligentes, zonas caminables y verdes, así como el lenguaje de programación Dart, el framework Flutter, el entorno de desarrollo *Visual Studio Code*, y los servicios requeridos para obtención y almacenamiento de las variables de actividad física. En el segundo capítulo se presenta la metodología de desarrollo utilizada para la codificación del prototipo, los requerimientos y modelado, así como la implementación de los módulos y funcionalidades que conforman el sistema. En el tercer capítulo se presentan los resultados de las pruebas de funcionalidad realizadas al prototipo de aplicación, la información obtenida de los usuarios registrados, conclusiones y recomendaciones.



**PALABRAS CLAVE:** Aplicación móvil, monitoreo ciudadano, actividad diaria, Flutter

# ABSTRACT

In the context of a smart and sustainable city, the information provided by citizens plays a fundamental role in determining the most important services, infrastructure, and activities of interest to each one of them. Furthermore, this collected information can be useful both for the user who wants to know about their daily activities, and their interaction with the available public spaces.

Due to this, the following Curricular Integration Work presents the development of a prototype mobile application for the Android operating system. This application is designed to collect information of interest for citizen monitoring. The prototype acquires and stores information about the citizen's daily activities, such as number of steps, sleep duration and time spent in green areas.

First Chapter introduces concepts related to smart cities, walkable and green zones, as well as the Dart programming language, Flutter framework, Visual Studio Code development environment, and the required services for obtaining and storing variables related to physical activity. Second Chapter presents the development methodology used for coding the prototype, including requirements, modeling, as well as the implementation of the modules and functionalities that constitute the system. Finally, Third Chapter exposes the results of functionality tests carried out on the application prototype, the information obtained from registered users, conclusions, and recommendations.

**KEYWORDS:** Mobile application, citizen monitoring, daily activity, Flutter

# 1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

Actualmente los sistemas de Internet de las Cosas (*Internet of Things*) representan la nueva tendencia para estructurar una red global de dispositivos conectados, que interactúan entre sí y con el medio ambiente a través de Internet. [1]

Por otro lado, el concepto de Ciudades Inteligentes (*Smart Cities*) abarca un entorno urbano moderno, que prioriza la disponibilidad de recursos y servicios que puede ofrecer una ciudad inteligente, los cuales permiten mejorar la calidad de vida de los ciudadanos mediante un modelo basado en información útil y procesamiento de datos. [2] En una ciudad inteligente, datos sobre el comportamiento de los ciudadanos; por ejemplo, datos relacionados con movilidad, actividad física y su interacción con el medio ambiente, pueden ser recopilados y almacenados mediante dispositivos móviles para dar soporte a ciudades inteligentes y sustentables. [3]

En base de lo mencionado, el siguiente proyecto de integración curricular tiene por objetivo el desarrollo de un prototipo de aplicación móvil para el sistema operativo Android, que permita el monitoreo del comportamiento de un ciudadano en sus actividades diarias y el uso de espacios verdes públicos mediante el prototipo de aplicación IoT dentro de un área de la ciudad de Quito.

## 1.1 Objetivo general

Desarrollar una aplicación móvil que permita el monitoreo del comportamiento de un ciudadano en sus actividades diarias, dentro de un área de la ciudad de Quito, utilizando aplicaciones IoT.

## 1.2 Objetivos específicos

1. Analizar la herramienta utilizadas para el desarrollo de aplicaciones en dispositivos móviles con sistema operativo *Android*.
2. Diseñar los diferentes componentes del prototipo de aplicación móvil para monitorear el comportamiento ciudadano en el contexto de una ciudad inteligente.
3. Implementar un prototipo de aplicación móvil para el monitoreo del comportamiento ciudadano en el contexto de una ciudad inteligente.
4. Realizar pruebas de funcionamiento del prototipo de aplicación móvil.

## 1.3 Alcance

Para el desarrollo del prototipo de aplicación móvil se toma como base la metodología ágil Kanban.

Se recopila información del *framework Flutter* utilizado para el desarrollo de aplicaciones móviles en sistemas operativos Android. Se estudia la información respecto al lenguaje de programación DART utilizado en el framework de desarrollo Flutter, el manejo de servicios móviles necesarios para el prototipo y el funcionamiento de Firebase para el almacenamiento de información en la nube.

Posteriormente se revisan las características de las aplicaciones disponibles para dispositivos móviles orientadas a la recopilación de datos, que permitan obtener información acerca de parámetros tales como: actividad física y tiempo de permanencia en puntos de interés como parques y áreas verdes.

A continuación, se obtienen los requerimientos funcionales y no funcionales a través de una entrevista realizada al menos a 5 potenciales usuarios del sistema. Se diseñan los componentes del prototipo de aplicación móvil tales como diagramas actividades, casos de uso y el diseño de las interfaces visuales del prototipo de la aplicación móvil, planteados según los requerimientos obtenidos.

El prototipo de aplicación móvil contiene los siguientes roles:

- **Administrador:** Permite la visualización de la cantidad de actividad física promedio de los usuarios registrados.
- **Usuario:** Permite el registro y autenticación, así como también observar la información de la ciudad, la cantidad de actividad física, la ubicación en tiempo real en el mapa y el tiempo de estadía promedio en áreas verdes.

Finalmente se realizan pruebas de funcionamiento del prototipo de aplicación móvil con al menos 5 usuarios para cada rol, luego de lo cual se analizan los resultados obtenidos y los ajustes pertinentes de ser el caso.

En Figura *1.1* se detalla un esquema general del prototipo de aplicación móvil con los servicios correspondientes.

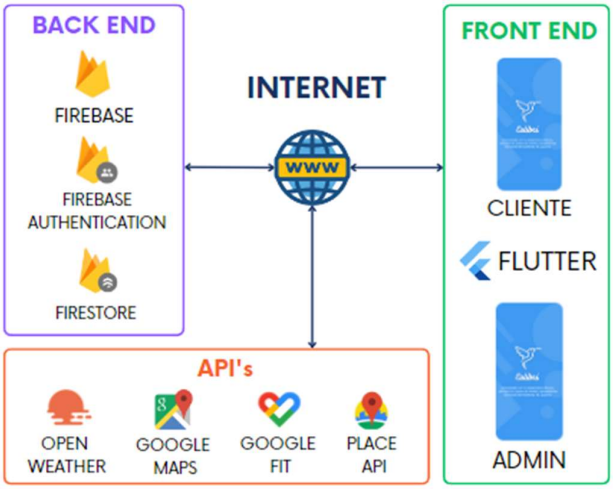



Figura 1.1. Arquitectura del prototipo de aplicación móvil.

## 1.4 Marco teórico

En esta sección se presentan conceptos importantes para la codificación de un prototipo de aplicación móvil, orientada a monitorear el comportamiento ciudadano en un área verde, en el contexto de una ciudad inteligente; así como, las características de las herramientas y tecnologías utilizadas para el desarrollo del prototipo de aplicación móvil utilizando tecnologías *IoT*.

### 1.4.1 Ciudades Inteligentes

Una ciudad inteligente (*Smart City)* se refiere a un espacio urbano moderno, que integra Tecnologías de la Información y Comunicación (TIC) para la administración de servicios básicos, tales como energía, suministro de agua, transporte, seguridad, educación, salud, entre otros. Con el fin de mejorar la eficiencia de estos servicios y la calidad de vida de sus habitantes, así como también influir en el impacto ambiental en una ciudad.[3]

La recopilación de datos para su posterior procesamiento permite obtener información relevante, para convertir ciudades en áreas urbanas inteligentes y autosustentables. Por ejemplo, datos relacionados con la movilidad, consumo de energía, el uso e interacción con áreas verdes de parte de los ciudadanos, así como su actividad física, recopilados mediante dispositivos móviles, o por dispositivos fijos estratégicamente ubicados conectados a Internet son sumamente importantes dentro de la conversión de ciudades convencionales a inteligentes. [4]

### 1.4.2 Movilidad sostenible

La movilidad sostenible se plantea como uno de los ejes fundamentales dentro de una ciudad inteligente, y se define como el desplazamiento dentro de un entorno urbano, priorizando el mínimo impacto con el medio ambiente y reducción de costos. [4]

En el Ecuador según la Comisión Económica para América Latina y el Caribe (CEPAL) se producen alrededor de 1.9 toneladas métricas de $CO_2$ por habitante. Lo que representa un 0.1% de emisiones a nivel mundial.[5]

Según INRIX, Quito fue para el año 2022 la ciudad más congestionada del Ecuador, debido a que aproximadamente el 70% de los ciudadanos se movilizan mediante transporte público o vehículo particular, a diferencia de la movilidad peatonal que ocupa el 14.8% de la movilidad en la ciudad. [6]

En la Figura *1.2* se muestra la distribución de movilidad para la ciudad de Quito durante el año 2022.

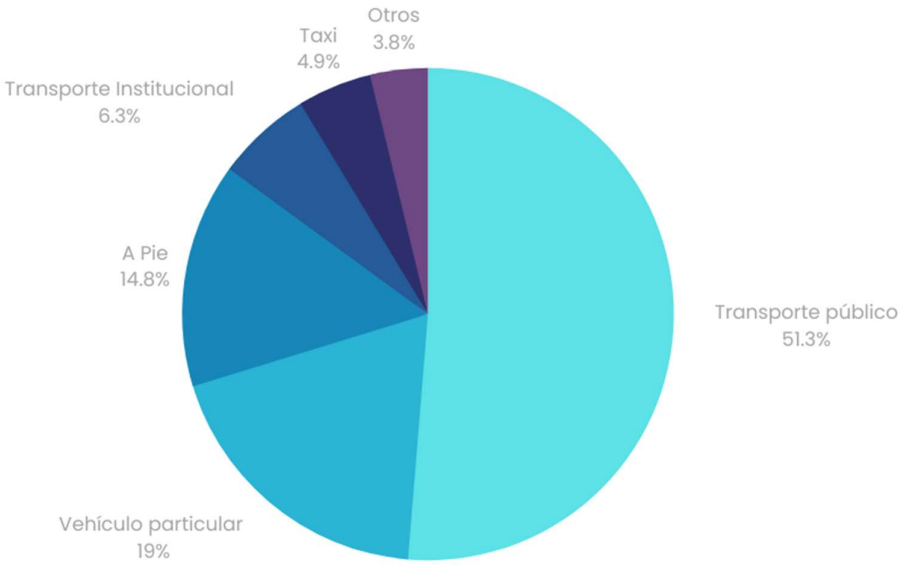



Figura 1.2. Distribución modal transporte en Quito[6]

Según el Plan de Movilidad Urbana Sostenible (PMUS) [7] implementado en algunas ciudades europeas tales como Copenhague (Dinamarca), Ámsterdam (Países Bajos) y Oslo (Noruega), una de las medidas a tomar es fomentar la movilidad a pie y en bicicleta, mediante la construcción de espacios y/o carriles exclusivos para este tipo de movilidad; además de eliminar barreras arquitectónicas que limiten la utilización de sistemas de movilidad alternativos a los transportes públicos y particulares.

### 1.4.3 Zonas caminables

Actualmente, la inactividad física se presenta como un factor de riesgo en el aumento de enfermedades tales como, la diabetes, hipertensión, enfermedades cardiovasculares, entre otras. Algunos estudios han reflejado que la población adulta sana camina entre 4000 a 18000 pasos diarios.[8]

Algunos de los principios de los cuales se basan los espacios caminables son:

- **Movilidad de base peatonal:** Las ciudades y zonas verdes caminables son un conjunto de espacios urbanos, que permiten y priorizan la actividad peatonal mediante la planificación urbanística y la regulación del espacio público.
- **Derecho al espacio público:** El espacio público es un espacio de socialización e inclusión debido a la diversidad de personas que lo utilizan.
- **Planificación urbanística caminable:** Diseñar el espacio público de manera que posibilite al peatón movilizarse a pie es un eje fundamental dentro de la planificación urbanística, por lo cual minimizar las distancias entre lugares de interés, servicios públicos o sitios de comercio se presenta como prioritario para reducir la atracción al uso de vehículos privados o transporte público
- **Seguridad vial:** Es uno de los principales enfoques a tomar en cuenta cuando se diseñan espacios caminables. Esta dimensión está diseñada para verificar el cumplimiento de las leyes de tránsito, tanto para la circulación vehicular como también para la peatonal.[9]

### 1.4.4 Zonas verdes

En el desarrollo de ciudades inteligentes y sostenibles, el concepto de áreas o zonas verdes juegan un papel fundamental. La importancia de espacios verdes en zonas urbanas se basa en los efectos positivos para la salud física y mental de los ciudadanos, así como la disminución en impactos sanitarios, económicos, ambientales y sociales.[10]

La ciudad de Quito para el año 2018 presentó un alto nivel de espacios verdes tal y como lo refleja el Índice Urbano Verde (IUV), que para ese año en Quito era de aproximadamente $21.6\ m^2$ por habitante, lo que la situaba como unas de las ciudades con más espacios verdes en Ecuador, incluso superando el índice internacional recomendado por la Organización Mundial de la Salud que marca el IUV en $9\ m^2$ por habitante [11]

La Figura *1.3* muestra la distribución del índice verde urbano (IVU) dentro del distrito metropolitano y zonas rurales aledañas de Quito.

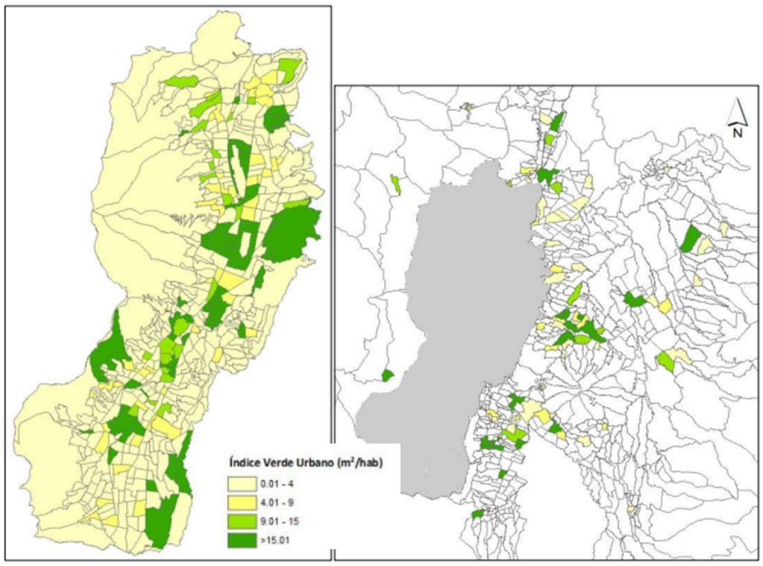



Figura 1.3. Distribución del índice verde urbano en Quito[11]

### *1.4.5 Dart*

Dart es un lenguaje de programación optimizado de código abierto creado por Google para el desarrollo de aplicaciones móviles, web y de escritorio; este lenguaje es de alto nivel, orientado a objetos y controlados por eventos.

DART proporciona un marco de referencia para ejecutar código para diferentes plataformas mediante las siguientes tecnologías de compilación.

- **Plataforma nativa**: permite el desarrollo de aplicaciones dirigidas a dispositivos móviles y de escritorio, mediante una máquina virtual con un compilador en tiempo de ejecución; además de un compilador de código nativo rápido.
- **Plataforma web:** permite desarrollar aplicaciones destinadas a navegadores web. Dart permite compilar aplicaciones web en tiempo de desarrollo o en producción mediante la traducción de código *Dart* a *JavaScript*.[12]

### 1.4.6 Flutter

Flutter es un *framework* o SDK (*Software development Kit*) creado por Google para el desarrollo de aplicaciones móviles nativas mediante el lenguaje de programación Dart. Este *framework* permite construir interfaces personalizadas para dispositivos móviles con diferentes sistemas operativos tales como Android, iOS, web y escritorio.

Flutter trabaja bajo la premisa de desarrollo rápido mediante el uso de *Hot Reload* y un conjunto de componentes personalizables llamadas *Widgets.*[13]

### 1.4.7 *Hot Reload*

Hot Reload permite la fácil y rápida construcción de interfaces gráficas (UI), agregar funcionalidades y corregir errores en tiempo de ejecución sin detener o reiniciar la aplicación. Esta función carga los cambios de código en la Máquina Virtual de Dart, reconstruyendo el árbol de Widgets sin detener o reiniciar el estado actual del emulador de dispositivo.[13]

### 1.4.8 Widgets

Los Widgets son un conjunto de componentes en *Flutter* que describen como debería verse estructurada visualmente una aplicación móvil, *web* o de escritorio; además de cómo se deben comportar los elementos dispuestos en la pantalla tales como botones, textos, imágenes, etc. [13]

Algunos de los Widgets más utilizados son los siguientes:

- ***Text*:** permite la inserción de cadenas de texto dentro de los componentes visuales de la aplicación.
- ***Row*, *Colum*:** permite organizar y posicionar otros widgets horizontalmente (*Row*) como verticalmente (*Column*).
- ***Stack*:** permite colocar o superponer widgets primarios por encima de otros widgets secundarios.
- ***Container*:** permite crear un elemento rectangular visual en la pantalla de la aplicación en el cual se pueden colocar otros widgets. Se pueden modificar sus propiedades tales como: margen, relleno y restricciones de tamaño. [13]

En la Figura 1.4 se muestra la distribución de los widgets *Text*, *Column* y *Row* dentro de una aplicación de *Flutter*.

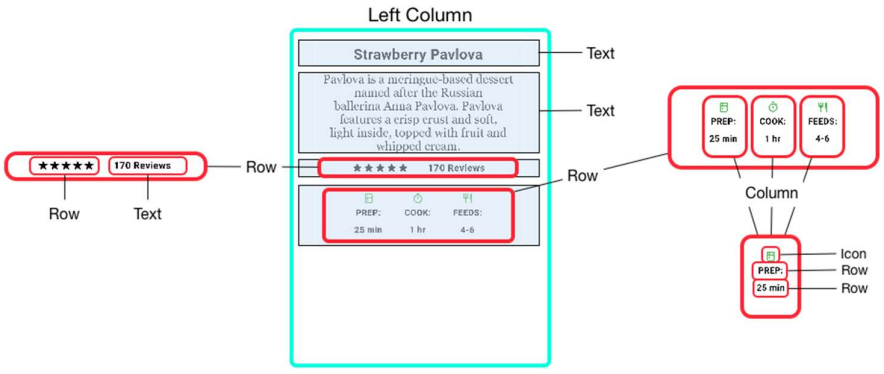



Figura 1.4. Distribución de widgets Flutter[13]

## 1.5 Firebase

Firebase es una plataforma alojada en nube creada por *Google*, que facilita el desarrollo, mantenimiento y despliegue de aplicaciones web y móviles a través de la implementación de los distintos módulos que la conforman. Además, Firebase cuenta con varias funcionalidades que permite obtener aplicaciones escalables de forma sencilla. [14]

Algunas de las funcionalidades de *Google Firebase* para el desarrollo del prototipo de aplicación son los siguientes:

- ***Firebase Autenticación*:** permite la autenticación de los usuarios en aplicaciones mediante números de teléfonos, números de identificación, correos, etc.[15]
- ***Firebase Storage*:** para almacenar contenido generado por aplicaciones móviles y webs tales como fotos, videos, archivos etc.[16]

  ***Firebase RealTime Database*:** permite el almacenamiento de la base datos *NoSQL* sincronizando a los clientes en tiempo real. Este servicio trabaja a través de *WebSockets*, lo que ayuda que los datos estén disponibles al instante.[17]

En la Figura 1.5 se muestra la consola de *Firebase* donde se pueden agregar, eliminar y configurar los servicios requeridos para un proyecto con *Flutter*. En la parte izquierda se muestran los accesos directos a los servicios utilizados.

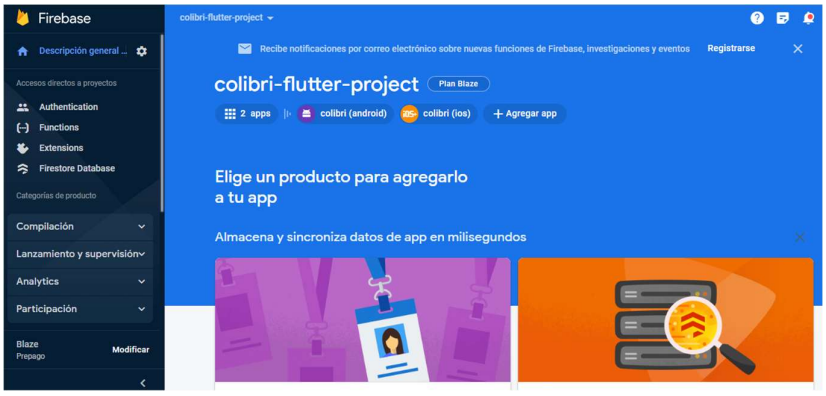



Figura 1.5. Consola *Firebase*

## 1.6 Herramienta de desarrollo

### *1.6.1 Visual Studio Code*

*Visual Studio Code* es un editor de código fuente gratuito creado por Microsoft, disponible para diferentes sistemas operativos como *Windows, Linux y MacOS. VS Code* es

principalmente utilizado para desarrolladores *frontend* en diferentes lenguajes de desarrollo tales como: JavaScript, Dart, Python, entre otros.[18]

Algunas características de *Visual Studio Code* son los siguientes:

- ***Intelli-Sense*:** Detecta pequeños bloques de códigos que se pueden encontrar incompletos; además de sintaxis incorrectas.
- **Extensiones*:*** *VS Code* cuenta con un sin número de extensiones para cada lenguaje de programación los cuales no ralentizan al editor.
- **Repositorio:** *VS Code* cuenta con soporte directo para gestión de almacenamiento en repositorios externos conectados en línea como GITHUB.[19]

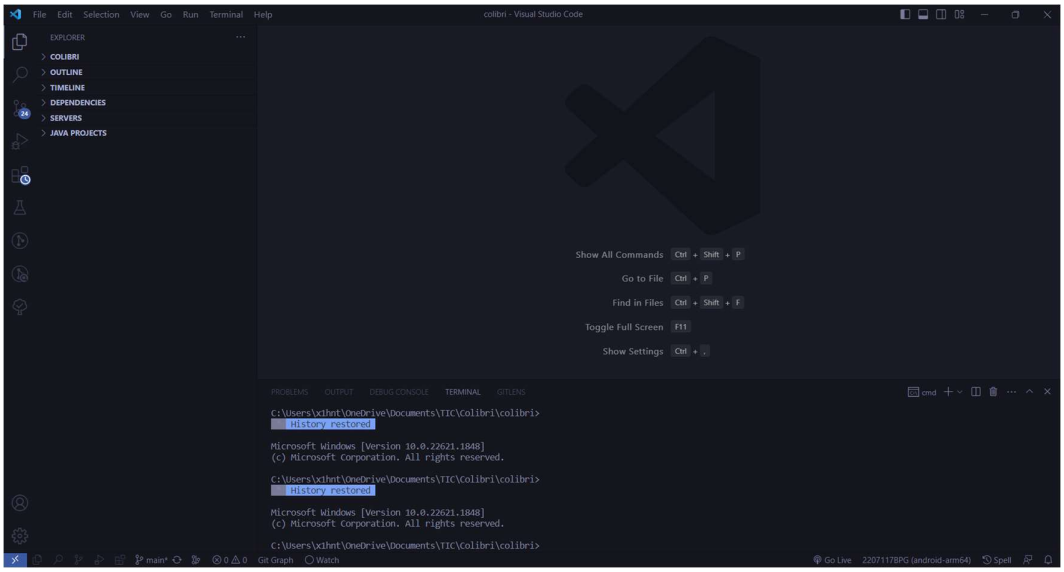



Figura 1.6. Entorno de trabajo *Visual Studio Code*

### 1.6.2 SCRCPY

SCRCPY es una aplicación disponible para sistemas operativos *Linux, Windows y macOS,* que refleja dispositivos Android (audio y video) conectados mediante puerto USB y que trabaja bajo el protocolo TCP/IP. Permite controlar el dispositivo móvil mediante el teclado y el mouse de la computadora.[20]

En la Figura *1.7* se muestra la ejecución de SCRCPY y el reflejo de la pantalla del dispositivo Android

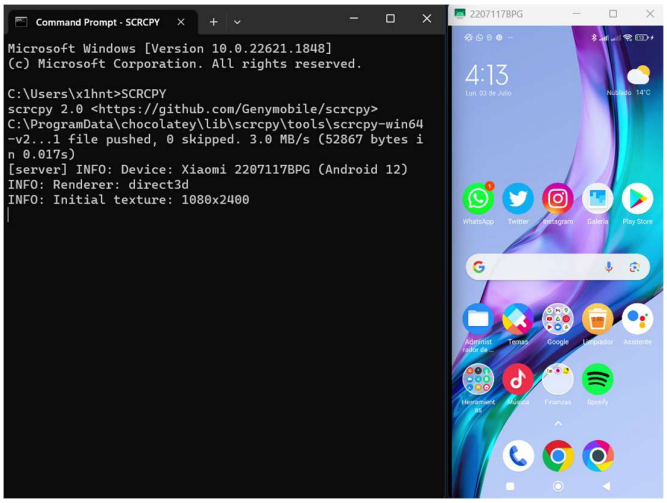



Figura 1.7. SCRCPY

### *1.6.3 Google Fit*

*Google Fit* es una aplicación móvil disponible para el sistema operativo Android e IOS desarrollada por Google que permite el registro, almacenamiento y seguimiento de la actividad física como: caminatas, entrenamientos, tiempo de sueño del usuario entre otros usando los sensores embebidos dentro del dispositivo móvil. Los parámetros calculados en *Google Fit* son recuperados mediante *API Fitness* disponible en *Google Cloud Platform*.[21]

### *1.6.4 Google Maps Platform*

*Google Maps Platform* es un conjunto de API's de Google que permite que los desarrolladores implementen soluciones de ubicación y geolocalización mediante *Google Maps* para aplicaciones móviles y web. Los servicios que ofrece son variados, desde la ubicación en tiempo real de los dispositivos conectados y localizados mediante el GPS embebido en el dispositivo móvil hasta el estado de servicio de lugares o puntos de interés.

La implementación de los servicios de localización y ubicación, así como el despliegue visual del mapa en los diferentes tipos de aplicaciones, se los hace a través del SDK de *Maps* disponibles en *Google Cloud Platform*.[22]

En la Figura *1.8* se muestra la implementación de un proyecto que hace uso de *Google Maps* dentro de un aplicativo móvil.

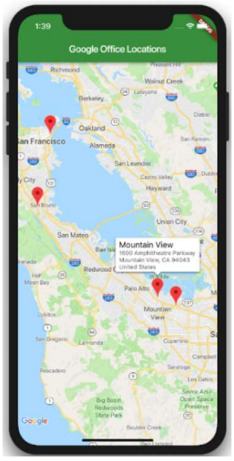



Figura 1.8. Despliegue *Google Maps* en dispositivo móvil

### *1.6.5 Places API*

*Places API* es un servicio disponible en *Google Cloud Platform*, que permite las solicitudes HTTP para obtener información en general de puntos de interés en relación con un punto geográfico suministrado por los aplicativos.

Adicionalmente, acepta solicitudes HTTP mediante una URL estándar construida por el tipo de lugar de interés, la llave de la API y la localización del dispositivo o la ubicación de interés cercano al punto y este retorna un documento en formato JSON o XML. [23]

En la Figura 1.9 se muestra un ejemplo de solicitud HTTP en la que se solicita los puntos de interés, los cuales estén catalogados como parques en un radio determinado por el usuario.

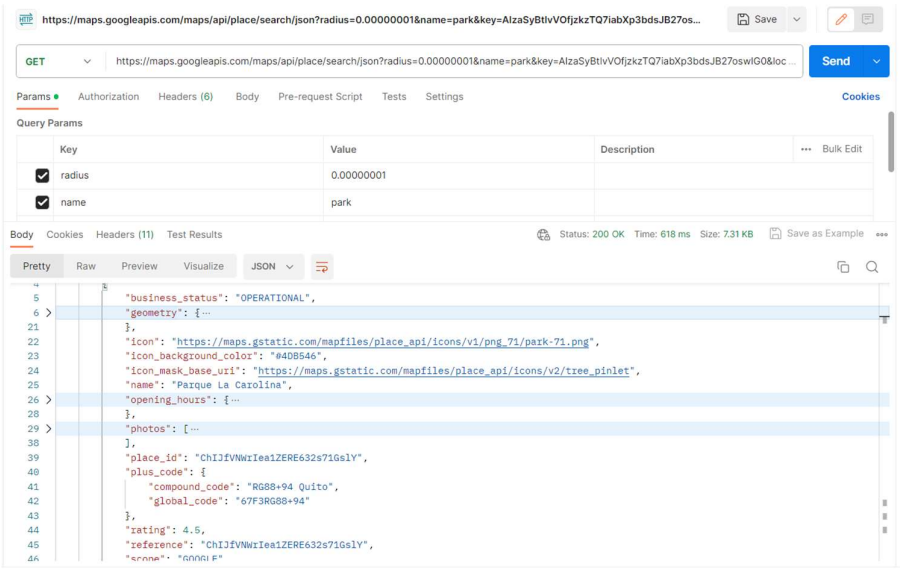



*Figura 1.9.* Solicitud a la *Places API* mediante *Postman*

# 2 METODOLOGÍA

En este capítulo se presenta el diseño y análisis de requerimientos funcionales y no funcionales, así como también el modelado e implementación del prototipo de aplicación móvil.

## 2.1 Metodología Ágil Kanban

La metodología Kanban es un marco de trabajo cuyo objetivo es controlar el desarrollo de software de forma eficiente, dividiendo los trabajos que se requieren hacer con la disponibilidad de cada miembro del equipo de trabajo. Kanban permite visualizar el trabajo y la carga para cada uno de los miembros del equipo de desarrollo por medio de tableros organizados por columnas divididos en “Por hacer”, “En progreso” y “Realizado”.

En la Figura *2.1* se visualiza el tablero Kanban inicial del prototipo de aplicación con las respectivas tareas por hacer, en progreso y terminadas.


Figura 2.1. Tablero Kanban inicial del prototipo

## 2.2 Diseño del prototipo

A continuación, se describe cómo se obtienen las características del sistema mediante el análisis de requerimientos funcionales y no funcionales. Así como el diseño de los diagramas de actividades, casos de usos y las interfaces graficas para cada uno de los módulos que conforman el prototipo de aplicación móvil.

### 2.2.1 Análisis de requerimientos

Las características del sistema fueron determinados por medio de una entrevista con Moisés Días estudiante de maestría en Ingeniería bajo la tutoría del PhD Jorge Sá Silva, miembros del equipo de trabajo del proyecto *UrsoVerde* de la Universidad de Coímbra.[24]

En base a la entrevista realizada y en el contexto de la ciudad de Quito se propone implementar una aplicación móvil, con la que se pueda monitorear el comportamiento de los ciudadanos, registrando variables relacionadas a la cantidad de actividad física como el número de pasos diarios, la cantidad de sueño y el tiempo de estadía en espacios verdes dentro de la ciudad basándose en las principales funcionalidades de *UrsoVerde*.

Tabla 2.1. Resumen resultado de la entrevista

| N° | PREGUNTA | SI % | NO% |
|---|---|---|---|
| 1 | ¿Es útil una aplicación que permita visualizar las variables de actividades diarias de usuarios? | 100 | 0 |
| 2 | ¿Considera útil que en la interfaz del usuario se pueda visualizar la información personal de este? | 100 | 0 |
| 3 | ¿Considera necesario que el usuario pueda visualizar la información de la ciudad en la cual se encuentra? | 100 | 0 |
| 4 | ¿Considera útil que la aplicación se encuentre distribuida en diferentes pantallas divididas en Inicio, Mapa, Actividad y Configuración? | 100 | 0 |
| 5 | ¿Considera adecuado que la información diaria del usuario se encuentre en la pantalla de Inicio? | 100 | 0 |
| 6 | ¿Considera útil llevar un registro del tiempo en la que el usuario se encuentra en un área verde? | 100 | 0 |
| 7 | ¿Considera útil el registro de la cantidad de pasos que el usuario realiza en un lapso de 24 horas? | 100 | 0 |
| 8 | ¿Considera útil el registro de la cantidad de tiempo que el usuario durmió en las últimas 24 horas? | 100 | 0 |

Con respecto a las preguntas realizadas dentro de la entrevista se define los módulos y funcionalidades con las cuales debe contar el prototipo de aplicación móvil. Además, es

indispensable plantear los requerimientos funcionales para los usuarios Cliente y Administrador.

### 2.2.2 Requerimientos funcionales para el usuario Cliente

Los requerimientos funcionales para el usuario Cliente se plantean con respecto a la entrevista realizada cuyas respuestas se exponen en la Tabla *2.1*. Además de las principales funcionalidades con las cuales cuenta el prototipo de *UrsoVerde.*

1. El sistema debe permitir el registro de nuevos usuarios a través de la interfaz gráfica, ingresando una dirección de correo electrónico y la contraseña suministrada por el usuario.
2. El sistema debe permitir la autenticación del usuario mediante el correo electrónico y contraseña.
3. El sistema debe contar con los botones y las rutas necesarias que permitan navegar a través de las diferentes pantallas que conforman el sistema.
4. El sistema debe permitir visualizar en resumen la cantidad de pasos, horas de sueño y tiempo pasado en parques o áreas verdes.
5. El sistema debe permitir visualizar la posición actual del usuario mediante un mapa.
6. El sistema debe permitir visualizar la cantidad de pasos realizados en el día comparados con el número de pasos propuestos como meta por el usuario.

### 2.2.3 Requerimientos funcionales para el usuario Administrador

1. El sistema debe permitir la autenticación del usuario Administrador mediante su correo electrónico y contraseña.
2. Se debe permitir la visualización de un resumen de la cantidad de pasos, horas de sueño y cantidad de tiempo en áreas verdes de todos los usuarios registrados como clientes.

### 2.2.4 Requerimientos no funcionales

A continuación, se presentan los requerimientos no funcionales del sistema adquiridos posterior al análisis de los requerimientos funcionales.

1. Crear un prototipo de aplicación móvil con el framework Flutter.
2. El sistema debe contar con una interfaz de usuario sencilla e intuitiva.

3. El sistema debe autenticar mediante los servicios de *Firebase Authentication.*
4. El sistema debe permitir almacenar los datos del usuario en el servicio de alojamiento de Firebase.

## 2.3 Actores y roles del prototipo

Del análisis de requerimientos funcionales y no funcionales del sistema se identifican los siguientes actores y como cada uno interactúan con el sistema.

- Administrador: Visualiza el número de usuarios registrados y los valores promedio relacionados con actividad física, tiempo de sueño y tiempo en áreas verdes de los usuarios registrados en el sistema y puede cerrar su sesión.
- Cliente: Puede registrarse, autenticarse y visualizar los datos sobre su actividad física, tiempo de sueño y tiempo en áreas verdes recolectados en el día, visualiza el mapa y puede cerrar su sesión.

## 2.4 Diagrama de casos de uso

Un administrador tiene acceso únicamente a la pantalla de autenticación y administración, donde puede visualizar los valores promedios de los usuarios registrados en el sistema; mientras que el usuario puede ingresar y visualizar todas las pantallas que conforman el sistema, excepto la pantalla de administración.

El usuario podrá registrarse, autenticarse, visualizar sus actividades físicas, ver su ubicación en el mapa y cerrar sesión, así como lo indica la Figura *2.2*.

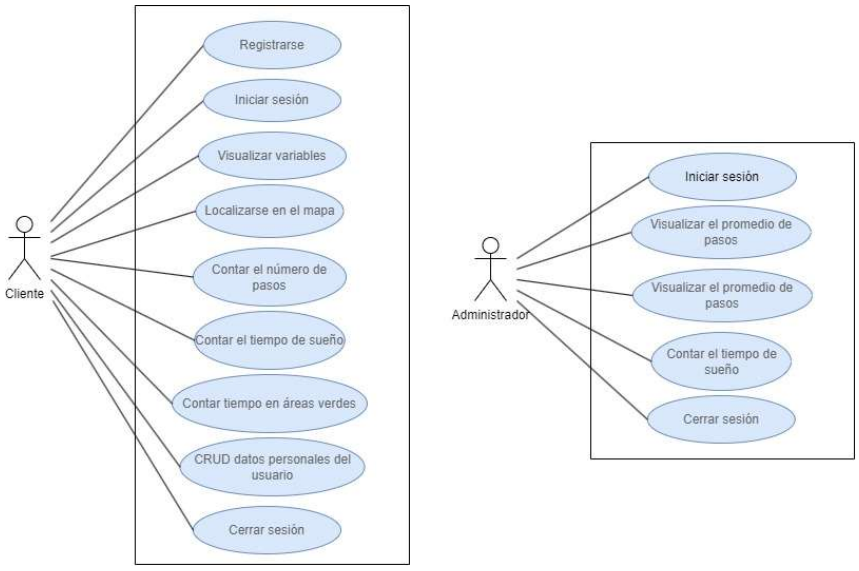



Figura 2.2. Diagramas de casos de uso

## 2.5 Funcionalidades del sistema

En esta sección se detallan las actividades que el usuario puede realizar en el sistema, para ello, se elaboran diagramas UML que permiten visualizar las siguientes actividades: El sistema permite el registro de nuevos usuarios, el ingreso de usuarios autenticados y la visualización de actividades diarias por parte del usuario.

En la Figura 2.3, se presenta el diagrama de actividades para el registro y autenticación del cliente.

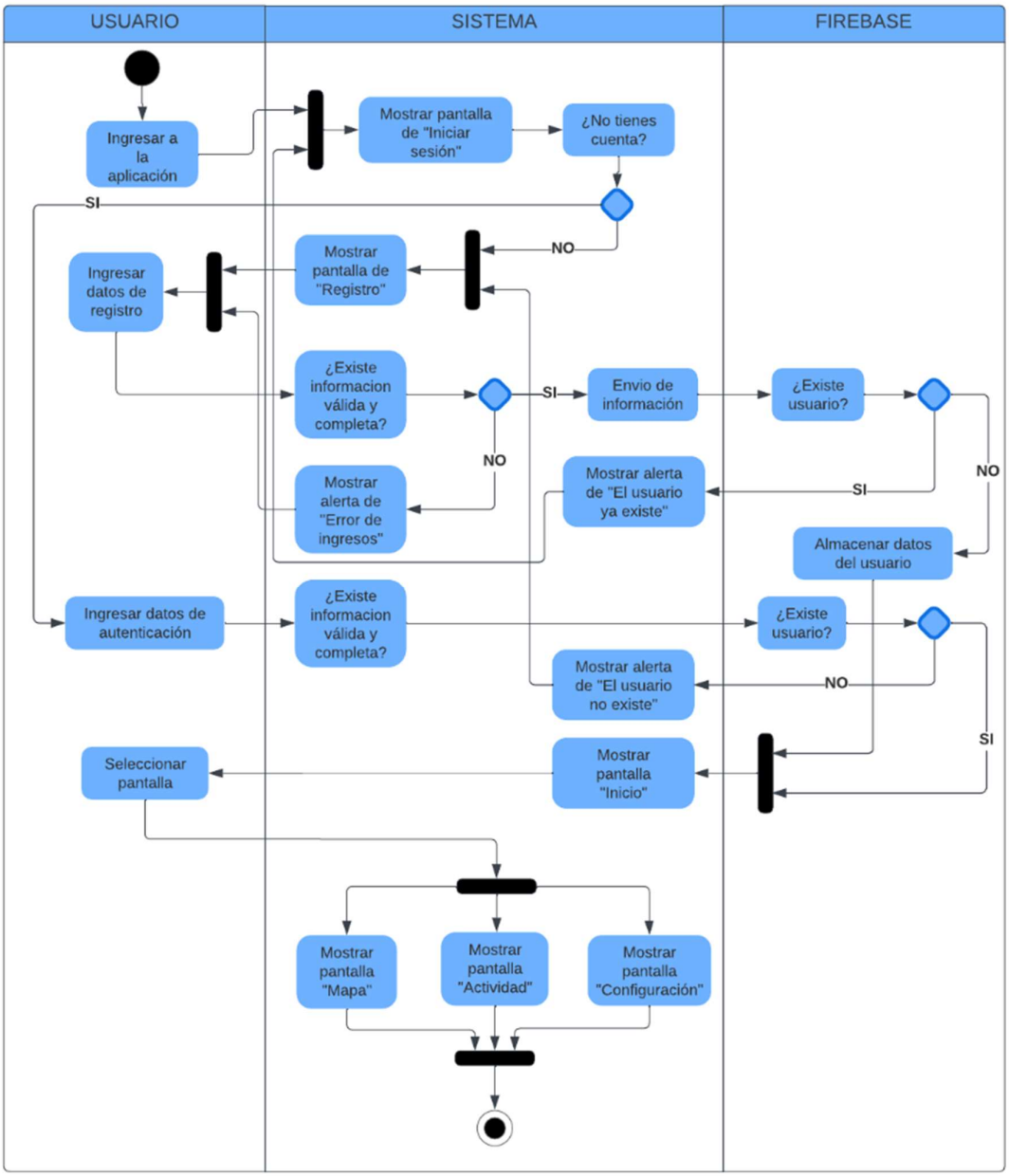



*Figura 2.3. Diagrama de actividades general del sistema*

En la Figura 2.4 y la Figura 2.5, se visualizan los diagramas de actividades para los casos en el que el cliente selecciona cada una de las pantallas que forman el sistema, y cómo este interactúa con el usuario y la base de datos.

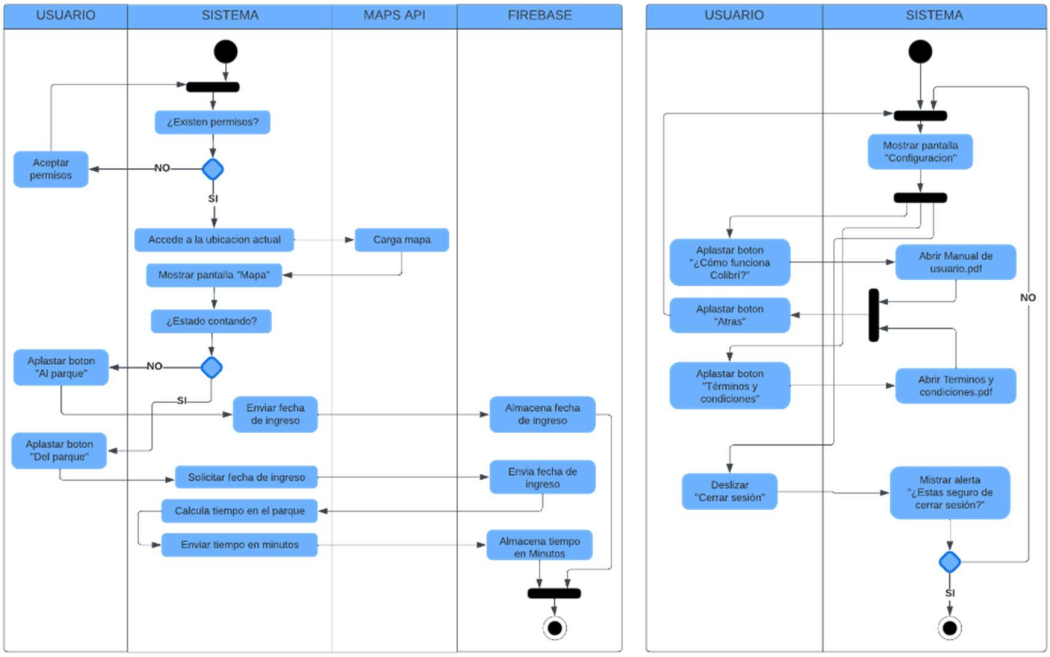



Figura 2.4. Diagrama de actividades módulos Mapa y Configuración

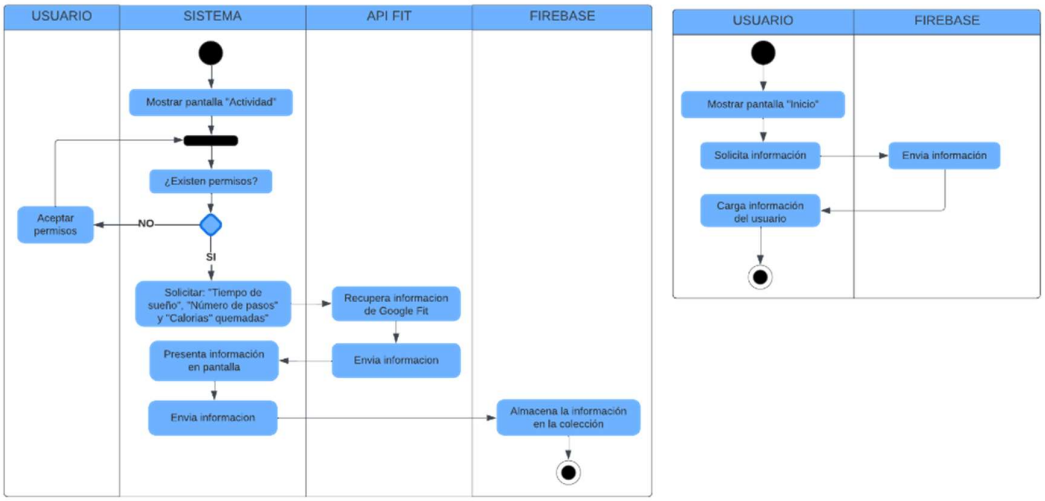



Figura 2.5. Diagrama de actividades módulos Actividad e Inicio

## 2.6 Diseño de la interfaz gráfica

El sistema cuenta con varios módulos que permiten interactuar con el usuario para visualizar la información del cliente, dependiendo del rol con el cual se autentique. A continuación, se describen las funcionalidades de cada uno de los módulos que pertenecen a la interfaz gráfica tanto para el cliente como para el administrador.

### 2.6.1 Actualización del tablero Kanban – Diseño de interfaces

En la Figura *2.6*, se muestra la actualización del tablero Kanban para la fase de implementación del prototipo.


Figura 2.6. Actualización del tablero Kanban Diseño de interfaces

### 2.6.2 Modelado de la interfaz gráfica del cliente

A continuación, se indican los modelos de las interfaces graficas de cada uno de los módulos que forman parte del aplicativo móvil, así como una breve descripción de las funcionalidades de cada uno de ellos.

**Módulo de registro:** Permite a los usuarios crear una cuenta para poder acceder a los servicios del prototipo de aplicación.

**Módulo de Autenticación:** Permite iniciar sesión en la cuenta previamente creada con los datos suministrados por el usuario.

En Figura 2.7, se muestran los modelados de los módulos de autenticación y registro en los cuales se solicita el ingreso de la información necesaria para registrarse e ingresar al sistema.

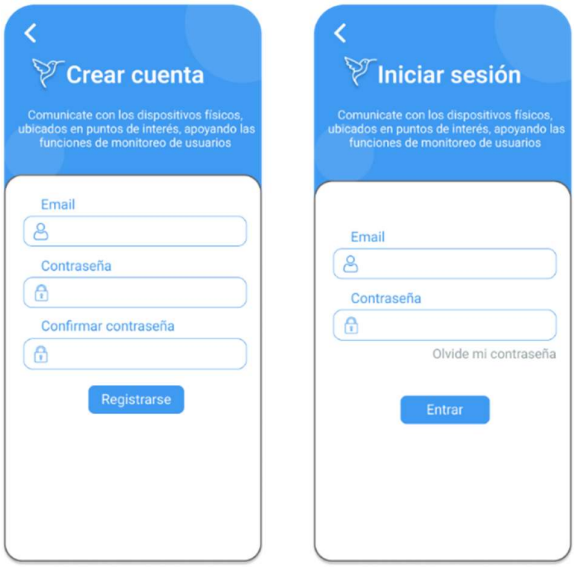



Figura 2.7. Modelado de la interfaz de registro y autenticación.

**Módulo de Inicio:** Permite visualizar la actividad física, horas de sueño y tiempo pasado en áreas verdes de los usuarios clientes.

**Módulo mapa:** Permite visualizar la ubicación en tiempo real del usuario cliente; además permite registrar manualmente el tiempo pasado en áreas verdes.

El modelado de estos dos módulos se muestra en la Figura 2.8.

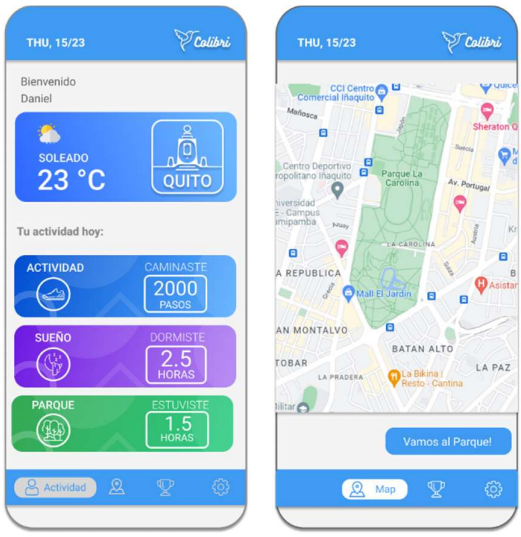



Figura 2.8. Modelado de la interfaz de Inicio y Mapa

**Módulo de actividad:** Permite visualizar la cantidad de pasos realizados en el día en comparación con la cantidad de pasos recomendadas y la cantidad de pasos fijada como objetivo.

**Módulo configuración e información:** Permite visualizar el Manual de usuario, los términos y condiciones del sistema y cerrar la sesión del usuario.

En la Figura 2.9 se prestan los modelados de la interfaz de Actividad y Configuración.

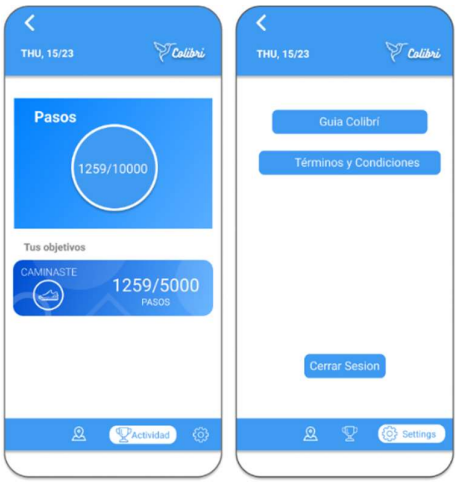



Figura 2.9. Modelado de la interfaz de Actividad y Configuración.

### 2.6.3 Modelado de la interfaz gráfica del administrador

A continuación se muestra el modelado de la interfaz gráfica del usuario administrador donde se puede la visualizar el número de usuarios registrados y las variables de actividad física promedio recolectados en las últimas 24 horas.


Figura 2.10. Modelado de la interfaz del administrador

## 2.7 Implementación del prototipo

En esta sección se describe el proceso para la implementación del prototipo de aplicación móvil previamente diseñado en la sección 2.7. En esta sección se diseñan específicamente las diferentes funcionalidades e interfaces gráficas los cuales conforman el sistema.

### 2.7.1 Actualización del tablero Kanban – Implementación

En la Figura *2.11*, se muestra la actualización del tablero Kanban para la fase de implementación del prototipo.


Figura 2.11. Tablero Kanban – Implementación

### 2.7.2 Configuración de servicios en *Firebase*

En esta sección se presenta la configuración realizada en la base de datos para el almacenamiento de la información, la cual es necesaria para la autenticación, recopilación de información y para la presentación de las variables requeridas para el sistema.

### *2.7.3* Configuración de *Firebase Authentication*

En la Figura *2.12*, se presenta el proveedor de acceso configurado en Firebase para el método de autenticación mediante correo electrónico y contraseña.

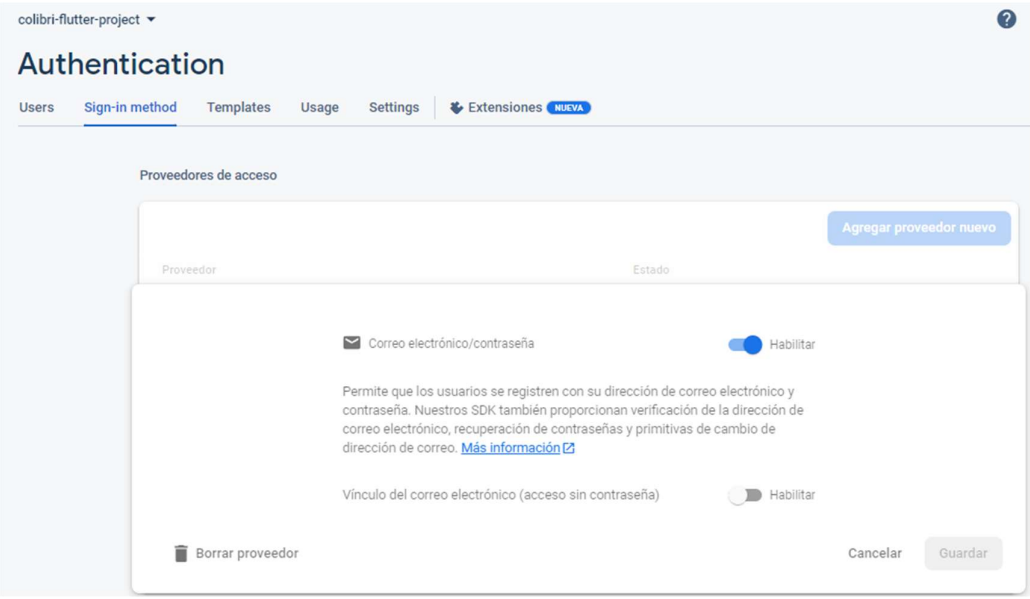



Figura 2.12. *Firebase Authentication*

Dentro del módulo autenticación también se realiza la configuración del servicio para la recuperación de contraseña, el cual se realiza íntegramente fuera del sistema debido a que es un servicio que lo presta Firebase.

En la Figura *2.13*, se presenta la configuración realizada para el envío del correo de recuperación de contraseña del sistema.

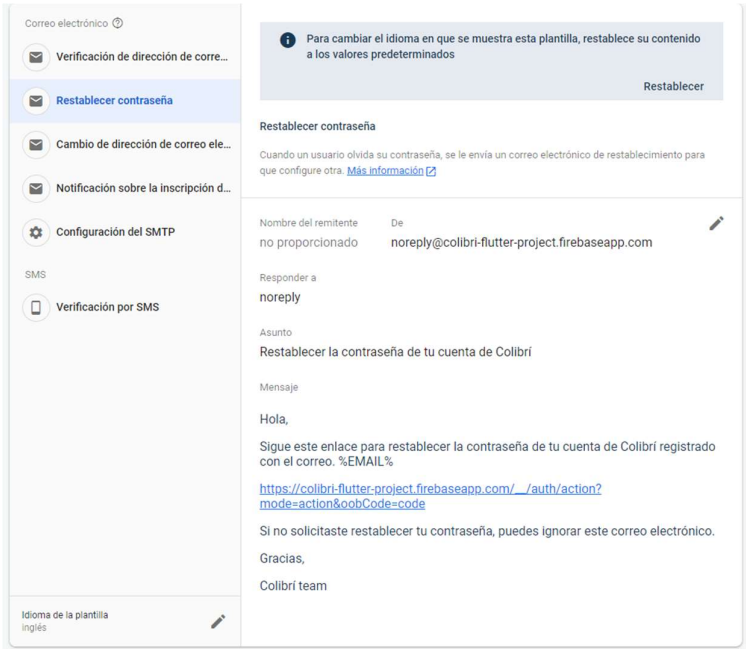



Figura 2.13. Configuración para el restablecimiento de la contraseña

### 2.7.4 Configuración de *Firestore Database*

La colección usuarios de la Figura *2.14*, muestra los campos con los cuales cuenta el rol administrador, los cuales son: *email, name, role, surname*; los cuales son utilizados para la autenticación y la navegación desde la pantalla de *login* hasta su pantalla de administrador.

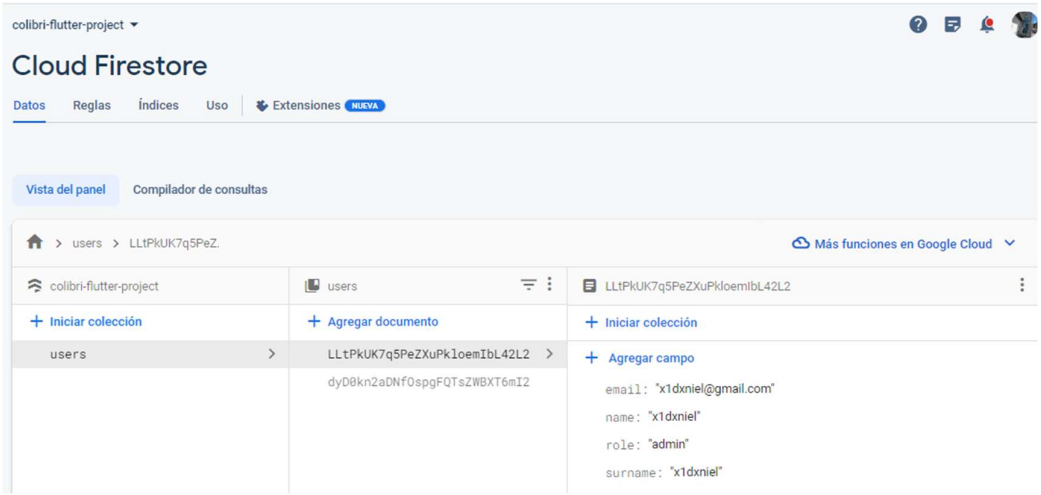



Figura 2.14. Colección usuarios del sistema

En la Tabla *2.2*, se muestra el nombre del campo, el tipo y su respectiva descripción con los cuales cuenta el rol usuario dentro del sistema y los cuales se almacenarán en el servicio *Google Cloud Firestore.*

Tabla 2.2. Campos de la colección usuario

| Campo | Tipo | Descripción |
|---|---|---|
| *Arrive* | *String* | Permite almacenar la última fecha de llegada hacia un parque o zona verde. |
| *Birthday* | *String* | Permite almacenar la fecha de nacimiento del usuario |
| *Contando* | *Bool* | Permite conocer si el conteo de horas en áreas verdes se encuentra activado. |
| *Email* | *String* | Permite almacenar el correo del usuario registrado. |
| *Role* | *String* | Permite identificar los tipos de usuarios registrados en el sistema |
| *Sleep* | *Number* | Permite almacenar el número de minutos dormidos en el intervalo de tiempo proporcionado. |
| *Steps* | *Number* | Permite almacenar el número de pasos realizados en el intervalo de tiempo proporcionado. |
| *Steps_goal* | *Number* | Permite almacenar los objetivos de pasos del usuario. |
| *Surname* | *String* | Permite almacenar el nombre de usuario registrado en el módulo de registro. |
| *Time* | *Number* | Permite almacenar el número de minutos que el usuario permanece en áreas verdes o parques. |

En la Figura 2.15, se presenta la disposición de los campos de la colección usuario requeridos para la autenticación y presentación de los datos relacionados al usuario cliente.

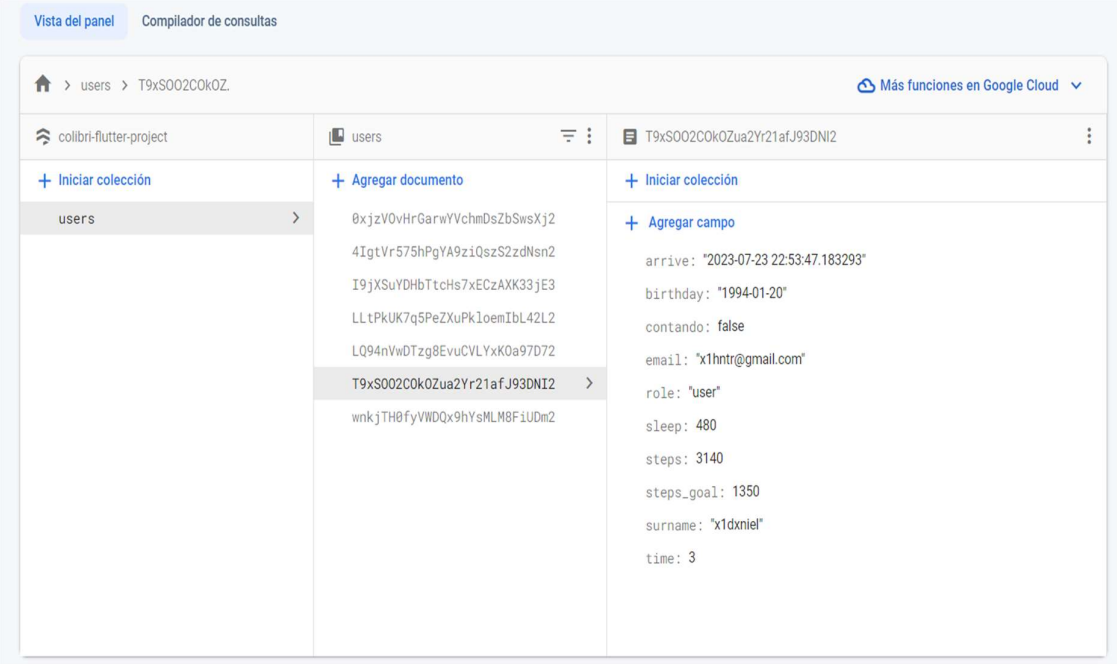



Figura 2.15. Colección usuarios del sistema en Firebase

### 2.7.5 Configuración *Google Maps y Places API*

En el siguiente apartado se detalla la configuración del servicio de *Google Maps y Place API* dentro de la consola de *Google Cloud Platform* y la implementación de la pantalla mapa en el sistema.

Para activar los servicios de *Google Maps y Places* API en el sistema, es necesario dirigirse a la consola para desarrolladores de *Google Cloud Platform*. Dentro de esta consola basta con buscar los servicios mencionados y en el apartado descripción general de cada uno habilitar el servicio para el sistema.

La Figura *2.16*, muestra el panel de administración de *Google Maps Platform* en el cual se puede visualizar las opciones de configuración y visualización de este servicio.

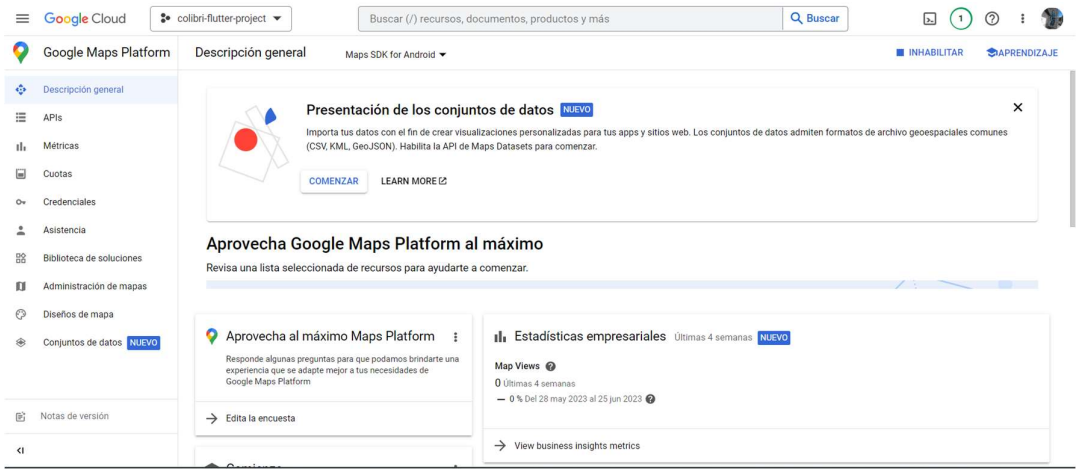



Figura 2.16. Consola de *Google Maps Platform*

En la Figura *2.17*, se puede visualizar el estado activo del servicio *Place API*, el mismo que se utiliza para recuperar el parque o área verde más cercano al usuario.

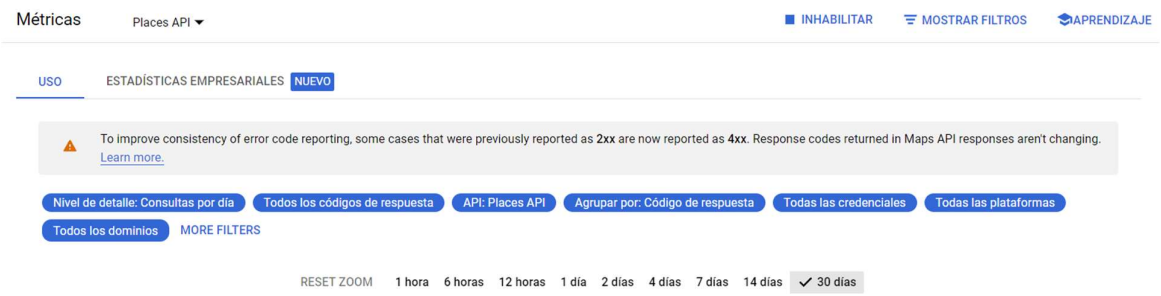



Figura 2.17. *Places API*

## 2.8 Implementación del Sistema

A continuación, se indica la codificación de la interfaz gráfica de cada una de las pantallas y de las funciones pertenecientes al prototipo de aplicación móvil modelados anteriormente.

### 2.8.1 Implementación de módulos Registro y Autenticación

Dentro de la implementación del módulo Registro se requiere el uso del método *createUserWithEmailPassword* de la Clase *FirebaseAuth*, que permite la inserción en un documento de una colección específica.

En la Figura *2.18*, se indica cómo se agrega un nuevo usuario dentro del sistema a través de la función *createUserWithEmailandPassword*, la cual se encarga de recolectar los datos ingresados en el formulario de registro, completar la información necesaria para crear un nuevo usuario y enviar la colección a la base de datos en *Firestore*.

```
FirebaseAuth.instance
    .createUserWithEmailAndPassword(email: emailTextController.text, password: passwordTextController.text)
    .then((value) {
  CollectionReference users = FirebaseFirestore.instance.collection('users');
  users
      .doc(value.user!.uid)
      .set({
        'arrive': '',
        'contando': false,
        'email': emailTextController.text,
        'role': 'user',
        'sleep': 1,
        'steps': 1,
        'steps_goal': 1,
        'surname':
            usernameTextController.text,
        'time': 1,
        'birthday': dateTextController.text
      })
```

Figura 2.18. Implementación de la función crear nuevo usuario.

Para la implementación del módulo de autenticación se requiere del uso de funciones de la librería de Firebase para Flutter. El método *signInWithEmailAndPassword* de la clase *FireabseAuth* permite el registro de nuevos usuarios en el servicio de autenticación, previamente configurado de Firebase, mediante un email y una contraseña.

En la Figura *2.19*, se presenta la implementación de la función autenticación del sistema, en el cual se suministra el correo y la contraseña previamente registrados a través de los controladores *emailTextController* y *PasswordTextController*.

```
  final value = await FirebaseAuth.instance.signInWithEmailAndPassword(
          email: emailTextController.text,
          password: passwordTextController.text);
  var uidUser = value.user!.uid;
  CollectionReference users =FirebaseFirestore.instance.collection('users');
  DocumentSnapshot documentSnapshot = await users.doc(uidUser).get();
  if (documentSnapshot.exists) {
    dynamic userDoc = documentSnapshot.data();
    if (userDoc!["role"] == "user") {
      // ignore: use_build_context_synchronously
      Navigator.push(context, MaterialPageRoute(builder: (context) => const HomePage()));
    } else {
      // ignore: use_build_context_synchronously
      Navigator.push(context, MaterialPageRoute(builder: (context) => const AdminPage()));
    }
  } else {
    print("Error");
  }
```

Figura 2.19. Implementación de la función para recuperar contraseña

Por otra parte, en la Figura *2.20*, se presentan las interfaces visuales para los módulos de registro y autenticación junto a los campos necesarios para realizar los dos procesos respectivamente.

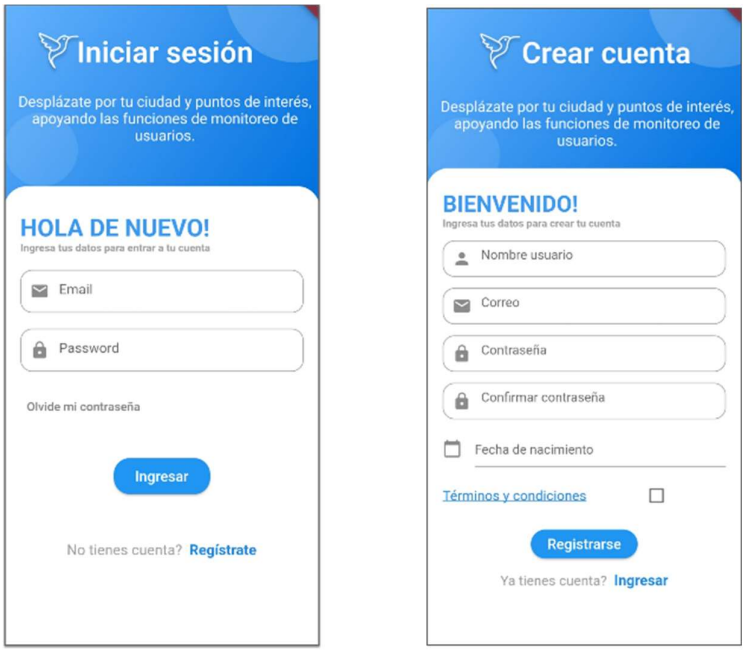



Figura 2.20. Módulos de Registro y autenticación

### 2.8.2 Implementación módulo recuperar contraseña

Este módulo permite el cambio de contraseña en caso de que el usuario olvide la ingresada y almacenada anteriormente, a través del servicio externo de restablecimiento de *Firebase*. El módulo recuperar contraseña está compuesto por el método *sendPasswordResetEmail*, el cual se encarga de enviar un correo electrónico a la dirección ingresada a través de la interfaz gráfica.

En la Figura *2.21*, se muestra el método *sendResetPassword* de la clase Firebase con el cual se envía al servicio de recuperación de contraseña de Firebase, tomando el valor del *TextFormField* en el cual el usuario ingresa el correo electrónico.

```
FirebaseAuth.instance
    .sendPasswordResetEmail(email: emailTextController.text);
Navigator.of(context).pop();
```

Figura 2.21. Método para encargado del envió de correo de restitución de contraseña

### 2.8.3 Implementación del tutorial del sistema

El sistema cuenta con una guía de inicio rápido, la cual se despliega por una única ocasión al ingresar por primera vez al sistema. Este módulo permite al usuario tener una idea general de las funciones con las cuales cuenta Colibrí y cómo puede interactuar con el sistema.

En la Figura 2.22, se muestra la pantalla de tutorial, el cual cuenta con siete pasos con los cuales el usuario puede interactuar a través del botones “Anterior” o “Siguiente”.

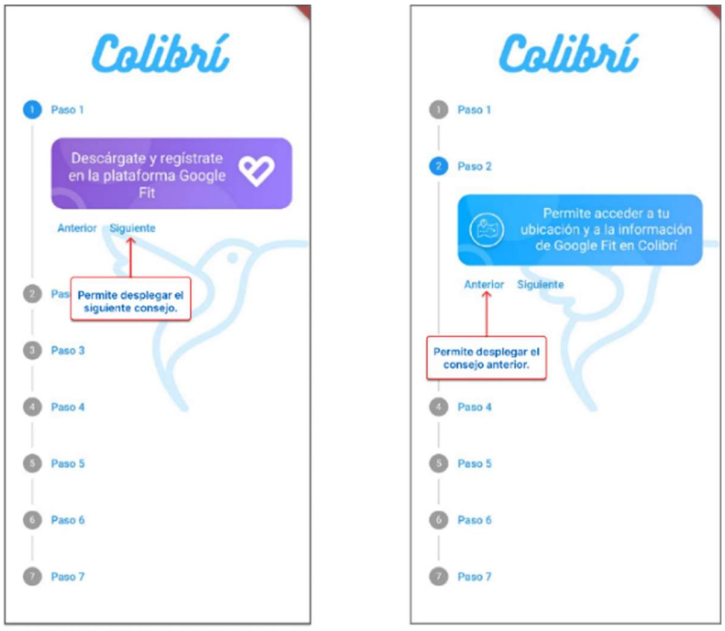



Figura 2.22. Guía de inicio rápido

### 2.8.4 Implementación del módulo de Inicio

En esta sección se describe la implementación de la pantalla de Inicio. Esta pantalla está formada por el panel de información básica y el panel de información de las variables de actividad física del usuario. Esta información es consultada a la base de datos a través de la función *getData,* encargada de recuperar los registros almacenados en la colección respectiva de cada usuario. La función *getData* descarga la información enlazada al identificador único del usuario autenticado en la instancia actual, el cual recupera a través del método *collection* de la clase *FirebaseFirestore* y lo almacena en una copia instantánea de la colección en *Firestore* del usuario.

```
getData() async {
  try {
    final value = FirebaseAuth.instance.currentUser;
    final uidUser = value!.uid;
    print(uidUser.toString());
    CollectionReference users = FirebaseFirestore.instance.collection('users');
    DocumentSnapshot documentSnapshot = await users.doc(uidUser).get();
    if (documentSnapshot.exists) {
      dynamic userDoc = documentSnapshot.data();
      nofSteps = userDoc!['steps'];
      nofSleep = userDoc!['sleep'].toDouble();
      nofArea = userDoc!['time'];
      surname = userDoc!['surname'];
      email = userDoc!['email'];
      birthday = DateTime.parse(userDoc!['birthday']);
```

Figura 2.23. Implementación del módulo de inicio.

En la Figura 2.23, se presenta la interfaz gráfica implementada, donde se puede visualizar los diferentes componentes que forman parte del módulo de inicio.


Figura 2.24. Módulo de Inicio

### 2.8.5 Implementación módulo Mapa

Para la implementación de la pantalla Mapa en el sistema, luego de haber habilitado el SDK de Google para mapas y la instalación de la dependencia para Flutter, se requiere cargar el *widget GoogleMap* dentro de un widget *Stack* para su visualización y posterior navegación.

En la Figura *2.25*, se muestra la implementación del widget *GoogleMaps*, el cual requiere ser inicializado con un objeto posición inicial del tipo *LatLng*, que provee la latitud y longitud.

```
GoogleMap(
  mapType: MapType.normal,
  initialCameraPosition: const CameraPosition(
    target: LatLng(-0.18523526393205988, -78.48430776506714),
    zoom: 16,
  ), // CameraPosition
  onMapCreated: (GoogleMapController controller) {
    _controller.complete(controller);
  },
  myLocationEnabled: true,
), // GoogleMap
```

Figura 2.25. *Widget Google Maps*

Para la visualización del mapa se requiere implementar funciones auxiliares, encargadas de la obtención de los permisos para acceder a los datos del GPS y de la ubicación actual del dispositivo, adicionalmente de los puntos de interés cercanos a esa ubicación.

En la Figura 2.26, se muestra la implementación de las funciones *determinePosition* y *getCurrentLocation* encargados de solicitar los permisos necesarios para acceder a los datos del GPS y de la posición actual del dispositivo.

```
Future<Position> determinePosition() async {
  LocationPermission permission;
  permission = await Geolocator.checkPermission();
  if (permission == LocationPermission.denied) {
    permission = await Geolocator.requestPermission();
    if (permission == LocationPermission.denied) {
      return Future.error('error');
    }
  }
  return await Geolocator.getCurrentPosition();
}

void getCurrentLocation() async {
  Position _position = await determinePosition();
  _initialPosition = LatLng(_position.latitude, _position.longitude);
}
```

Figura 2.26. Funciones para determinar los permisos de acceso a los datos del GPS.

En la Figura *2.27*, se muestra la forma con la cual se solicita el permiso para obtener la ubicación del dispositivo móvil en tiempo real. Este permiso se le presenta por una única ocasión al usuario cuando ingresa por primera vez al módulo Mapa.


Figura 2.27. Pantalla de permisos para la ubicación

### 2.8.6 Cálculo de horas en áreas verdes

Para recuperar la cantidad de horas en las que el usuario permanece en áreas verdes se implementa la función *_goToThePark*. Esta función al ejecutarse envía a la base de datos la fecha, hora de llegada al área verde; además del estado de conteo dentro de la aplicación.

Además, es esta misma función la encargada de posteriormente al terminar la medición de tiempo, recuperar la hora de llegada almacenada en la base de datos y calcular el tiempo de estadía en esa área verde.

En la Figura *2.28*, se muestra la implementación de la función requerida para iniciar el conteo del tiempo que permanece un usuario en un área verde.

```
Future<void> _goToThePark() async {
  if (status == false) {
    print('LLEGANDO');
    status = true;
    getNearbyPlaces();
    llegada = DateTime.now();
    print(llegada.toString());
    setArrive(true, llegada.toString());
  } else {
    getData();
    status = false;
    print(llegada.toString());
    print(DateTime.now().toString());
    Duration tiempo = DateTime.now().difference(llegada);
    print(tiempo.inMinutes);
    setTime(false, tiempo.inMinutes.toInt() + _time);
  }
}
```

Figura 2.28. Función para iniciar conteo de tiempo en áreas verdes

Para validar que el usuario se encuentre en un área verde se implementa la función asincrónica *getNearbyPlaces*. Esta función permite recuperar a través del servicio *Places API* de Google los puntos de interés cercanos a la ubicación actual del dispositivo. El resultado obtenido mediante esta función es un documento de formato JSON, que posteriormente al decodificarlo da como resultado el nombre del parque más cercano al dispositivo móvil.

En la Figura *2.29*, se indica la implementación de la función *getNerbyPlaces* la cual permite obtener el parque o área verde más cercano al usuario.

```
void getNearbyPlaces() async {
  var url = Uri.parse(
      'https://maps.googleapis.com/maps/api/place/search/json?radius=' +
          radius +
          '&name=park&key=' +
          apiKey +
          '&location=' +
          _initialPosition!.latitude.toString() +
          ',' +
          _initialPosition!.longitude.toString());
  var response = await http.post(url);
  nearbyPlaces = NearbyPlaces.fromJson(jsonDecode(response.body));
  Results results = Results();
  results = nearbyPlaces.results![0];
  print(results.name);
}
```

Figura 2.29. Función para obtener parques más cercanos al usuario

En la Figura 2.30, se indica la respuesta del servicio *Places* mediante una petición HTTP en la cual se muestra el parque más cercano a la ubicación actual del usuario.


```
▼ "results": [
  ▼ {
      "business_status": "OPERATIONAL",
    ▶ "geometry": { 2 items },
      "icon": "https://maps.gstatic.com/mapfiles/place_api/icons/v1/png_71/park-71.png",
      "icon_background_color": "#4DB546",
      "icon_mask_base_uri": "https://maps.gstatic.com/mapfiles/place_api/icons/v2/tree_pinlet",
      "name": "Parque de la Dammer",
    ▼ "opening_hours": {
        "open_now": true
```


Figura 2.30. Resultados de la consulta HTTP a *Places API*

En la Figura *2.31*, se presentan los pasos que el usuario realiza para empezar el conteo de horas en áreas verdes, y las posibles respuestas que el sistema presenta ante el inicio del conteo dentro y fuera de un parque.

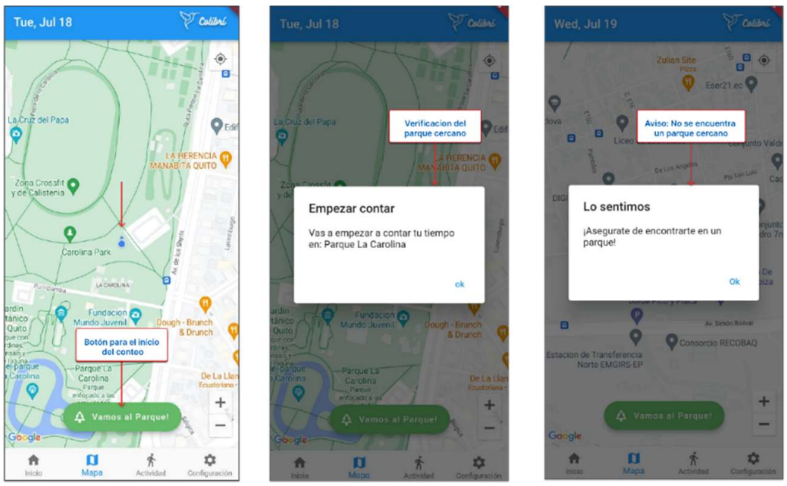



Figura 2.31. Módulo mapa

### 2.8.7 Implementación Módulo Actividad

Esta sección incluye la codificación del módulo actividad, así como de las funciones necesarias para la adquisición de las variables que resultan de interés para monitorear el comportamiento ciudadano dentro de una ciudad inteligente.

El módulo actividad cuenta con diferentes submódulos los cuales se encargan de la adquisición y presentación de la cantidad de pasos que el usuario realiza en lapso de 24 horas, la presentación y solicitud del número de pasos que el usuario se plantea como objetivo y la presentación del número de calorías que el usuario quema en la última sección de actividad física.

En la Figura *2.32*, se indica el módulo actividad implementado, en el cual se presentan los pasos registrados por el usuario, la cantidad de calorías quemadas en la última actividad física y la selección de los pasos objetivos planteados por el usuario.

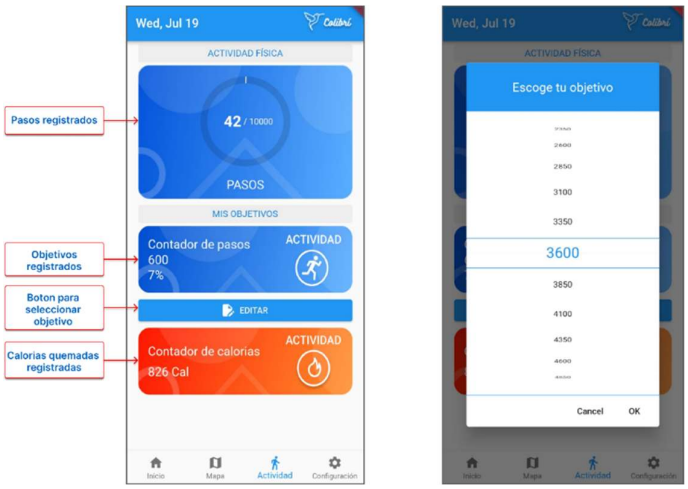



Figura 2.32. Módulo actividad

### 2.8.8 Adquisición de la información de actividad física

Para la adquisición de los valores de actividad física provenientes de *Google Fit* es necesario la configuración previa del permiso para el mismo. Este proceso es posible a través de la función *initRequest* que realiza la gestión de la solicitud de permiso, el cual es presentado al usuario mediante un cuadro emergente de diálogo, este solicita el correo con el cual el usuario está registrado en la aplicación *Google Fit*.

En la Figura *2.33*, se presenta la implementación de la función booleana *initRequest*, encargada de solicitar y verificar los permisos hacia las variables de salud que se las recupera a través de la aplicación *Google Fit*.

```dart
Future<bool> initRequest() async {
  try {
    bool requested = false;
    requested = await health.requestAuthorization(allTypes);
    return requested;
  } catch (e) {
    return false;
  }
}
```

Figura 2.33. Función encargada de solicitar permisos de la actividad física del usuario

La Figura *2.34*, muestra la ventana emergente encargada de solicitar el permiso adecuado para la información recolectada por *Google Fit.*

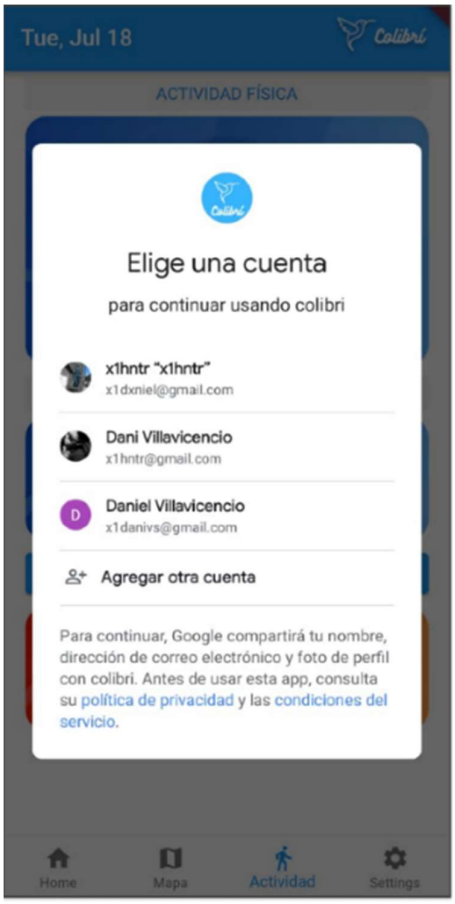



Figura 2.34. Pantalla de solicitud de permisos

### 2.8.9 Adquisición del número de pasos y horas de sueño

La adquisición del número de pasos y horas de sueño dentro del prototipo de aplicación móvil se las obtiene a través de la aplicación *Google Fit*, que mediante el servicio

configurado en *Google Cloud Platform* y la librería *Healt 7.0.1* permite obtener los datos correspondientes a salud desde *Google Fit o Apple Health*.

A continuación, se presentan las funciones necesarias para la recuperación del número de pasos y la cantidad de horas de sueño.

En primer lugar, la clase *StepData* permite recuperar los datos relacionados con actividad física, en este caso específico la cantidad de pasos realizados en el día. El método *fetchStepData* permite recuperar de forma asincrónica la cantidad de pasos en un intervalo de tiempo entre la media noche y la hora en la cual se abre la aplicación.

En la Figura *2.35*, se muestra la implementación para la adquisición del número de pasos desde la app *Google Fit*.

```
class StepData {
  HealthFactory health = HealthFactory();

  Future<int?> fetchStepData() async {
    int? steps;
    final now = DateTime.now();
    final midnight = DateTime(now.year, now.month, now.day);
    bool requested = await health.requestAuthorization([
      HealthDataType.STEPS,
      HealthDataType.SLEEP_IN_BED,
    ]);
    if (requested) {
      try {
        steps = await health.getTotalStepsInInterval(midnight, now);
      } catch (error) {
        print("Error getTotalStepsInInterval: $error");
      }
      print('Numero de pasos: $steps');
      return steps;
    } else {
      print("Error de autorizacion");
    }
  }
}
```

Figura 2.35. Función encargada de la adquisición del número de pasos

La clase *SleepData* permite recuperar la información relacionada con las horas de sueño. El método futuro *fetchStepData* permite recuperar de forma asincrónica el número de horas de sueño en un intervalo de tiempo delimitado por las variables *midnight, now* y *types,* esta última variable define el tipo de dato que se requiere, teniendo como opciones y tiempo despierto.

En la Figura *2.36*, se presenta la implementación para la adquisición del número de horas de sueño.

```
class SleepData {
  HealthFactory health = HealthFactory();
  Future<double?> fetchStepData() async {
    double? sleep = 0;
    String sleep2 = 'hi';
    final types = [
      HealthDataType.SLEEP_IN_BED,
    ];
    List<HealthDataPoint> healthData = [];
    final now = DateTime.now();
    final from = DateTime.now().subtract(const Duration(days: 1));
    bool requested = await health.requestAuthorization(types);
    if (requested) {
      try {
        healthData = await health.getHealthDataFromTypes(from, now, types);
      } catch (error) {
        print("Error getTotalStepsInInterval: $error");
      }

      sleep2 = "${healthData[0].value}";
      sleep = double.parse(sleep2);
      return sleep;
    } else {
      print("Error de autorizacion");
    }
  }
}
```

Figura 2.36. Función encargada de la adquisición del tiempo de sueño

### 2.8.10 Almacenamiento del número de pasos y horas de sueño

Para el almacenamiento de las variables de actividad física, el sistema requiere la implementación de los métodos *setData* y *setSteps*. Estos métodos se encargan de armar un documento temporal con los datos suministrados por los métodos de adquisición y almacenarlos en la colección respectiva del usuario, a través del método *update* de la clase *FirebaseFirestore* con el respectivo UID del usuario autenticado, el cual es recuperado a través del *currentUser* del método *FirebaseAuth*.

En la Figura *2.37*, se observa la implementación de las funciones necesarias para el almacenamiento en la base de datos de las variables de actividad física tales como el número de pasos y el tiempo de sueño.

```
setData(nofSteps, nofSleep) async {
  try {
    final arrive = {
      "sleep": nofSleep,
      "steps": nofSteps,
    };
    final value = FirebaseAuth.instance.currentUser;
    final uidUser = value!.uid;
    final datos = FirebaseFirestore.instance;
    datos.collection('users').doc(uidUser).update(arrive);
  } catch (e) {
    print("$e");
  }
}

setSteps(steps_goal) async {
  try {
    final arrive = {
      "steps_goal": steps_goal!,
    };
    final value = FirebaseAuth.instance.currentUser;
    final uidUser = value!.uid;
    final datos = FirebaseFirestore.instance;
    datos.collection('users').doc(uidUser).update(arrive);
  } catch (e) {
    print("$e");
  }
}
```

Figura 2.37. Funciones para almacenar la información en *Firebase Storage*

### 2.8.11 Implementación del módulo configuración e Información

El módulo configuración e información permite al usuario visualizar los términos y condiciones del sistema, el manual de usuario en formato PDF y permite cerrar sesión del sistema.

La implementación de las funciones de visualización de los términos y condiciones y del manual de usuario en formato PDF, se las implementa a través del método *launchUrl*, el cual se encarga de abrir una ventana externa a la aplicación, direccionada mediante una url donde están alojados estos documentos.

La función para cerrar sesión se la implementa en un cuadro alerta para confirmar la intensión por parte del usuario para realizar esta acción. Dentro de este cuadro se pregunta al usuario si "¿Estás seguro de que quieres cerrar sesión?"; además, se le presentan las opciones “ok” y “cancelar”, donde al presionar “ok” la aplicación realiza la función de cerrar sesión y redirige al usuario a la pantalla de autenticación, finalizando así el ciclo del proceso total de la aplicación.

En la Figura *2.38*, se presenta la función para cerrar sesión implementada mediante el método *signOut* de la clase *FirebaseAuth* para el usuario autenticado en la instancia actual.

```
Widget okButton = TextButton(
  child: Text("Si"),
  onPressed: () async {
    await FirebaseAuth.instance.signOut();
    Navigator.of(context).pushAndRemoveUntil(
        MaterialPageRoute(builder: (context) => LoginPage()),
        (route) => false);
  },
); // TextButton
// set up the AlertDialog
AlertDialog alert = AlertDialog(
  title: Text("Cerrar Sesión"),
  content: Text("¿Estas seguro que quieres cerrar sesión?"),
  actions: [
    okButton,
    cancelButton,
  ],
); // AlertDialog
```

Figura 2.38. Funciones para cerrar sesión

En la Figura *2.39*, se puede visualizar la pantalla de configuración e información; además del cuadro de dialogo encargado de confirmar la intensión del usuario para cerrar sesión.

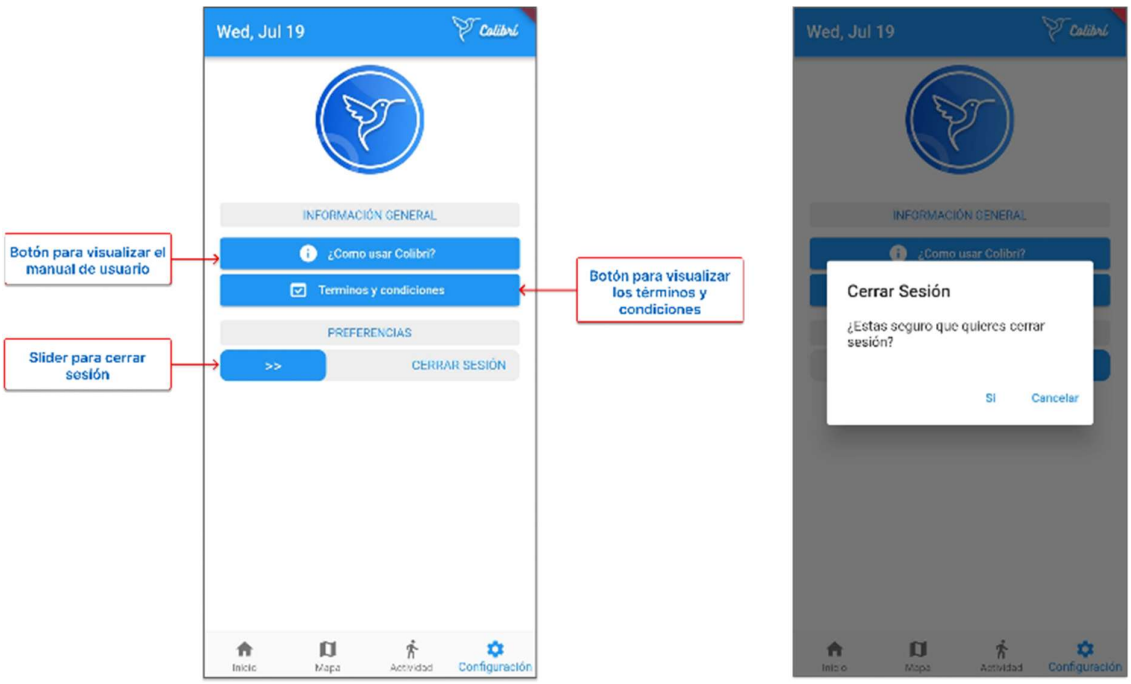



Figura 2.39. Módulo Configuración e Información

### 2.8.12 Implementación del backend

Para la obtención de los valores promedios de los usuarios registrados es necesario la implementación de una función mediante la cual se recojan los datos de los usuarios con el rol "*user*" almacenados en Firebase *Database*. Este servicio es almacenado a través de una función implementada en Firebase, la cual se encarga de filtrar los usuarios y determinar los promedios requeridos.

En la Figura 2.40, se presenta la implementación del filtro para obtener los datos del usuario; así como la obtención de los datos recogidos y el cálculo de cada uno de los promedios requerido.

```
const filterUsers = users.docs.filter((user)=>user.data()["role"]==="user");
nofUsers = filterUsers.length;
let sumSteps = 0;
let sumSleep = 0;
let sumArea = 0;

for (const user of filterUsers) {
  const steps = user.data()["steps"];
  const sleep = user.data()["sleep"];
  const area = user.data()["time"];
  sumSteps += steps;
  sumSleep += sleep;
  sumArea += area;
}
nofSleep = sumSleep/nofUsers;
nofSteps = sumSteps/nofUsers;
nofAreas = sumArea/nofUsers;
return {
  nofUsers: nofUsers,
  nofAreas: nofAreas,
  nofSleep: nofSleep,
  nofSteps: nofSteps,
```

Figura 2.40. Implementación del backend

En la Figura 2.41, se muestra la función implementada en *Firebase*, como las características de las peticiones *HTTP* del servicio.

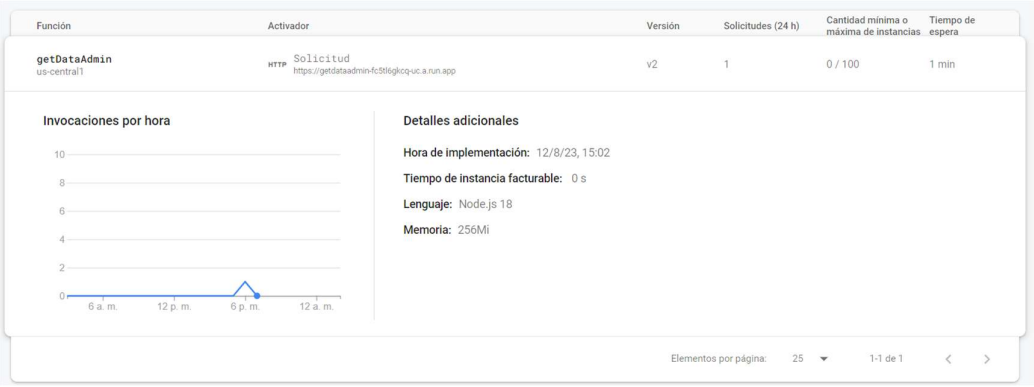



Figura 2.41. Función implementada en *Firebase*

### 2.8.13 Implementación módulo de administración

El módulo de administración permite al usuario administrador visualizar los usuarios registrados, los promedios del número de pasos, horas de sueño y tiempo en áreas verdes. Este módulo hace uso del método *httpsCalleable* de la clase *FirebaseFunction* para obtener la información que suministra la función almacenada en Firebase.

La Figura *2.42*, se puede visualizar la función *getInformation*, la cual con el método *httpsCalleable*, obtiene los valores promedios de actividad física y tiempo en áreas verdes implementadas en el *backend.*

```
Future<void> getInformation() async {
  HttpsCallable callable =
      FirebaseFunctions.instance.httpsCallable('getDataAdmin');
  final results = await callable();
  print("el data: " + results.data.toString());
  nofUsers = results.data["nofUsers"];

  nofSteps = results.data["nofSteps"];
  nofSleep = results.data["nofSleep"];
  nofArea = results.data["nofAreas"];

  print('Caminar: ' + nofSteps.toString());
  print('Dormir: ' + nofSleep.toString());
  print('Areas: ' + nofArea.toString());
  print('Users: ' + nofUsers.toString());
}
```

Figura 2.42. Función para obtener valores promedios para el administrador

En la Figura *2.43*, se puede visualizar el módulo de administración donde se muestra el número de usuarios registrados en el sistema y los promedios de las variables almacenadas de todos los usuarios.


Figura 2.43. Módulo de administración

# 3 PRUEBAS, RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

Para validar el sistema implementado, en la presente sección se describen los resultados que se obtuvieron al realizar pruebas de funcionalidad a los componentes que conforman el sistema. Donde se comprueban la correcta implementación de los requerimientos funcionales y no funcionales definidos en el capítulo 2.

Las pruebas de las funcionalidades del sistema se realizaron con cinco habitantes de la ciudad de Quito dentro del casco urbano, los cuales registraron su actividad por 15 días de forma continua entre las fechas 24 de julio y 6 de agosto del 2023.

## 3.1 Actualización del tablero Kanban – Pruebas

Previamente a las pruebas de funcionalidades del sistema se actualiza el tablero Kanban y se colocan las actividades de la fase de diseño e implementación en la columna de actividades completadas y las actividades de pruebas de funcionalidad corrección de errores y presentación de resultados en la columna de actividades en proceso.

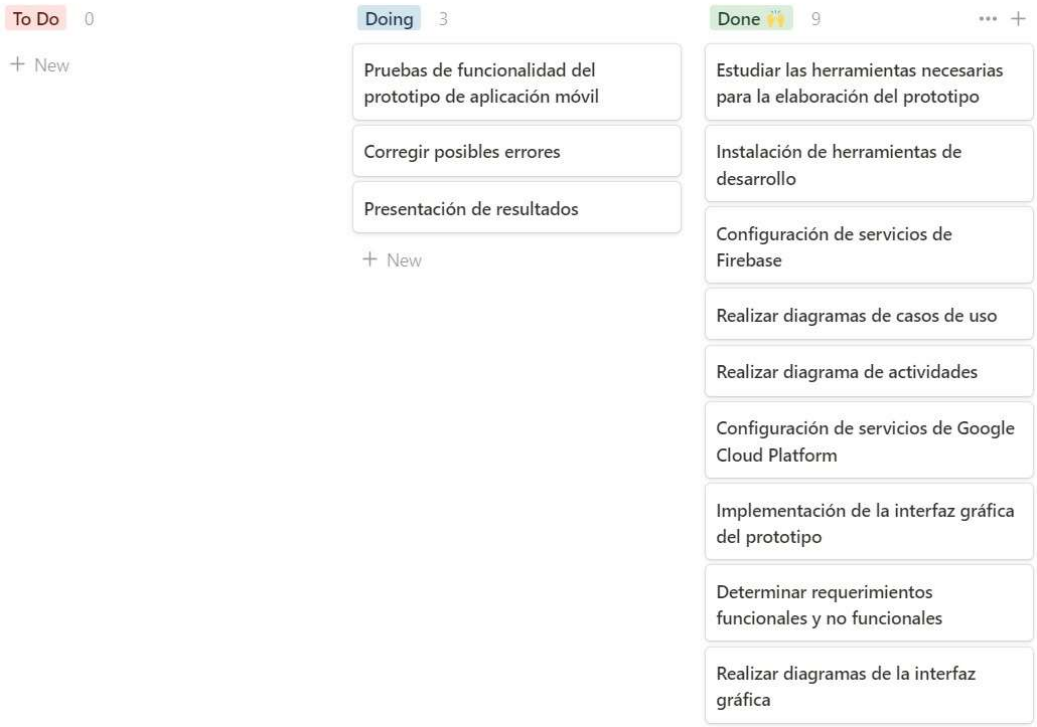



Figura 3.1. Tablero Kanban – Pruebas de funcionalidad

## 3.2 Pruebas

En esta sección se detallan las pruebas realizadas del prototipo de aplicación móvil, revisando las validaciones con las cuales cuentan las pantallas que requieren el ingreso de información de parte del usuario. Además, se describe el comportamiento y la funcionalidad de cada de los módulos y componentes de la aplicación móvil como se detalla en el manual de usuario mostrado en el ANEXO *VII*

### 3.2.1 Módulo de registro

El módulo de registro cuenta con las validaciones necesarias para que el usuario ingrese correctamente el nombre de usuario, correo, contraseña, y fecha de nacimiento. El sistema también verifica que la contraseña ingresada es correcta mediante la repetición de ésta en otro cuadro de texto.

Las pruebas en el módulo de registro consisten en el ingreso de dos contraseñas diferentes en los campos contraseña y confirmación de contraseña, y de esta forma visualizar a través de cuadro de dialogo emergente que estos campos no coinciden. Adicionalmente, a través de otro cuadro de dialogo informar si el usuario ingresó una contraseña de al menos 6 caracteres.

En la Figura *3.2*, se muestra las validaciones implementadas para el ingreso de contraseña y la verificación de coincidencia de contraseña.

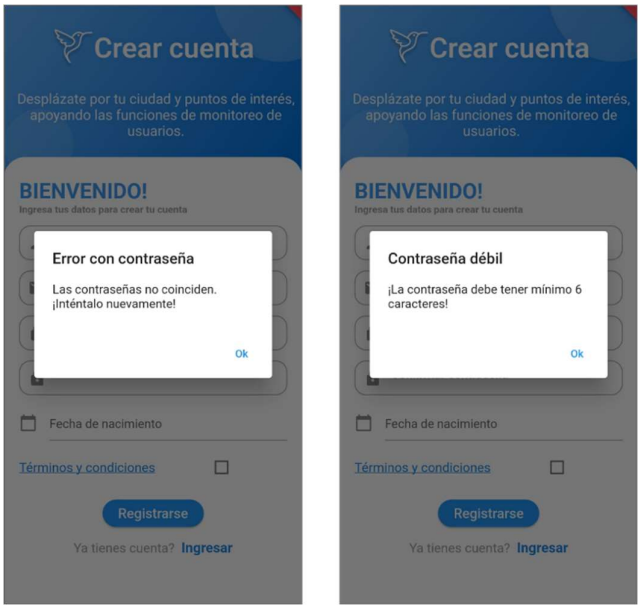



Figura 3.2. Validaciones en el módulo de registro

### 3.2.2 Módulo de Autenticación

Mediante la información ingresada el sistema permite la autenticación del usuario para su posterior ingreso al sistema. El módulo de autenticación cuenta con la validación para verificar la información ingresada por el usuario es correcta, y advierte mediante un cuadro de diálogo emergente si no lo es.

La prueba del módulo de autenticación consiste en ingresar correos y contraseñas no válidas, de esta forma el sistema muestra una ventana emergente donde se le indica al usuario que está ingresando información errónea.

La Figura *3.3* presenta la validación del sistema para el correo y contraseña ingresada, este verifica si el correo ingresado tiene una cuenta registrada y si la contraseña es la correcta para ese usuario.

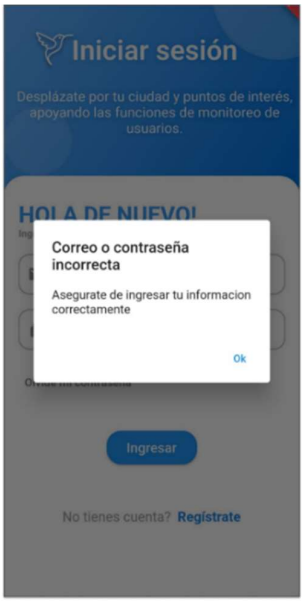



Figura 3.3. Validaciones módulo de Inicio de sesión

### 3.2.3 Módulo recuperación de cuenta

En este módulo se permite al usuario recuperar su cuenta mediante el cambio de contraseña. El módulo recuperación de cuenta envía un correo a través del servicio Firebase Autenticación al correo registrado por el usuario.

Las pruebas de funcionamiento del módulo recuperación de cuenta consiste en realizar el ejercicio de solicitar el enlace de restablecimiento de contraseña el cual se envía al correo electrónico solicitado en la interfaz gráfica como lo indica en la Figura *3.4*.

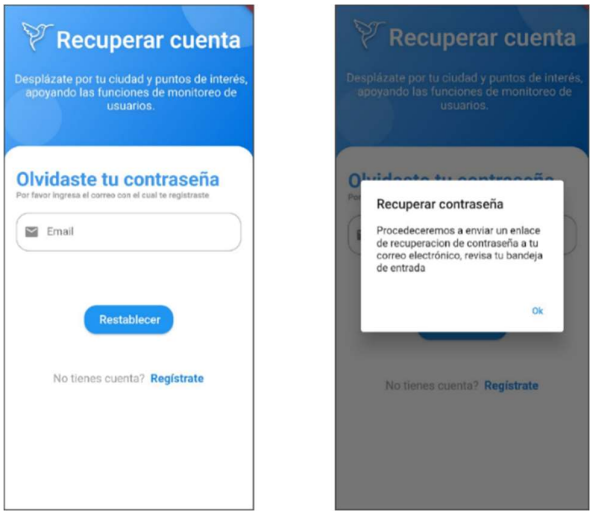



Figura 3.4. Prueba de recuperación de cuenta

A través del sistema de recuperación de contraseña de Firebase se gestiona el correo de recuperación que se envía a la bandeja de entrada del usuario registrado. Este enlace se encarga de redireccionar al usuario al formulario de ingreso de una contraseña nueva.

En la Figura *3.5*, se muestra el correo enviado a la bandeja de entrada del usuario, así como el formulario de ingreso de nueva contraseña.

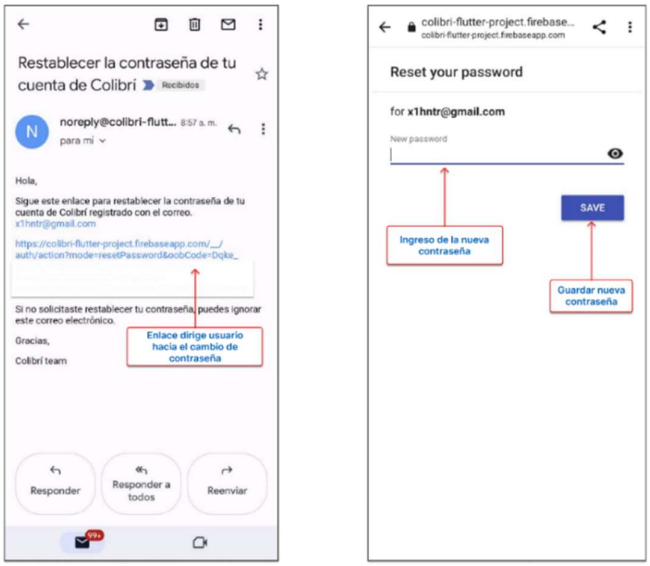



Figura 3.5. Correo recuperación de cuenta

### 3.2.4 Módulo Mapa

El módulo mapa cuenta con la validación para que el sistema se asegure de comenzar a medir el tiempo en áreas verdes a través de notificaciones emergentes posterior al presionar el botón “¡Vamos al parque!”.

Las pruebas en este módulo consisten en intentar comenzar a medir el tiempo dentro de un área verde y fuera de ésta, y visualizar como el sistema responde ante estas peticiones.

En la Figura *3.6*, se muestra el módulo Mapa y cómo el sistema responde ante la interacción en el instante en el que el usuario desea comenzar a medir su tiempo en áreas verdes y fuera de ellas.

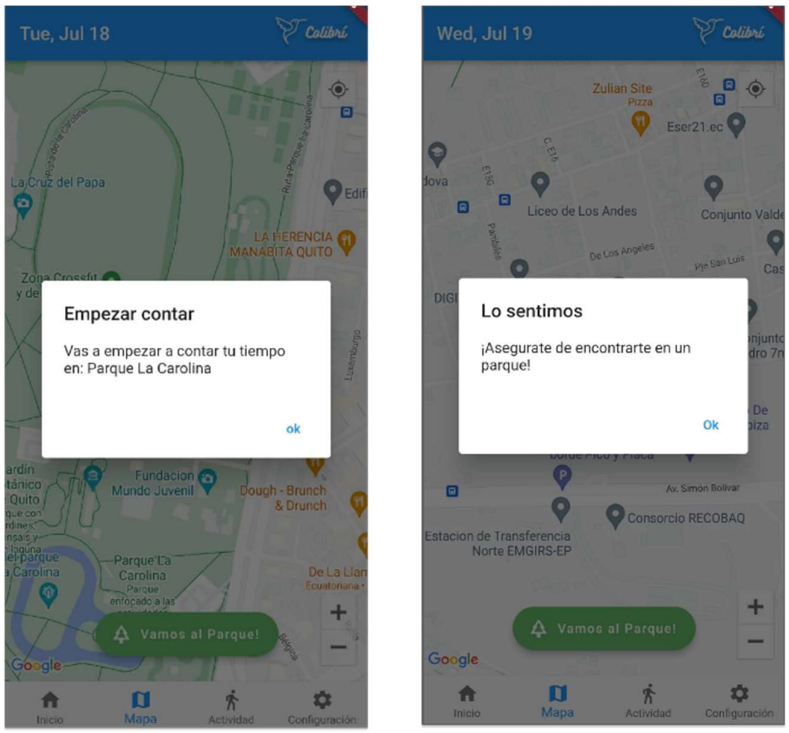



Figura 3.6. Pruebas conteo tiempo en áreas verdes

### 3.2.5 Módulo de configuración

En la pantalla de configuración se permite la visualización del manual de usuario, y los términos y condiciones de uso; además, en este apartado se puede cerrar la sesión que se encuentra activa.

Las pruebas en el módulo de configuración consisten en desplazar la opción de cerrar sesión y visualizar el cuadro de dialogo emergente que pregunta si el usuario está seguro de cerrar sesión.

La Figura *3.7* muestra la validación para confirmar si el usuario desea cerrar sesión. Al presionar “Si” el usuario confirma su salida del sistema dirigiéndole a la pantalla de Inicio de sesión; mientras que al presionar “Cancelar” el sistema lo vuelve a dirigir hacia la pantalla de Inicio.

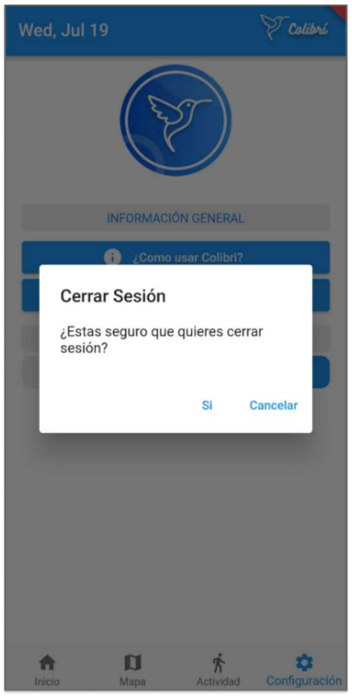



Figura 3.7. Prueba cerrar sesión del usuario

## 3.3 Pruebas de integración

En esta sección se muestran las pruebas de integración de la aplicación para los módulos en los cuales están presentes los servicios prestados por Firebase.

Esta prueba se realiza al intentar integrar un nuevo usuario a través de la pantalla de registro del sistema, así como lo muestra la Figura 3.8


Figura 3.8. Registro de nuevo usuario

En Figura *3.9*, se muestran los usuarios registrados a través del módulo de registro del sistema, estos usuarios registrados pueden hacer uso del módulo de autenticación para el ingreso al sistema por parte de los clientes usuarios y administradores.

| Identificador | Proveedores | Fecha de creación ↓ | Fecha de acceso | UID de usuario |
|---|---|---|---|---|
| m.d.gomezoth@gmail.com | ✉ | 5 ago 2023 | 5 ago 2023 | 6n67nxRhuRT1ybkSo5YEke7mi1h1 |
| x1hntr@gmail.com | ✉ | 25 jul 2023 | 14 ago 2023 | IT0eG0sWzVOBdz6R9p8hT7Vl7cN2 |
| danilosebastian2207@gm... | ✉ | 24 jul 2023 | 24 jul 2023 | WVmD9E3FYmWSNwUBwzqlldyG... |
| jessenia.chuquimarca2020... | ✉ | 24 jul 2023 | 24 jul 2023 | I9jXSuYDHbTtcHs7xECzAXK33jE3 |
| ksisa94@gmail.com | ✉ | 23 jul 2023 | 23 jul 2023 | wnkjTH0fyVWDQx9hYsMLM8FiUD... |
| erikapupiales98@gmail.com | ✉ | 23 jul 2023 | 23 jul 2023 | LQ94nVwDTzg8EvuCVLYxKOa97D... |
| martinamantilla@gmail.com | ✉ | 18 jul 2023 | 18 jul 2023 | 0xjzVOvHrGarwYVchmDsZbSwsXj2 |
| julio.gonzalez2014a@gmai... | ✉ | 17 jul 2023 | 21 jul 2023 | 4IgtVr575hPgYA9ziQszS2zdNsn2 |
| x1dxniel@gmail.com | ✉ | 6 jun 2023 | 14 ago 2023 | LLtPkUK7q5PeZXuPkloemlbL42L2 |

Figura 3.9. Usuarios registrados en Firebase a través del sistema

La Figura *3.10*, presenta los usuarios registrados en el sistema y las variables que se recolectan en cada uno de los módulos que forman parte del prototipo de aplicación y tienen funciones de almacenamiento de información.

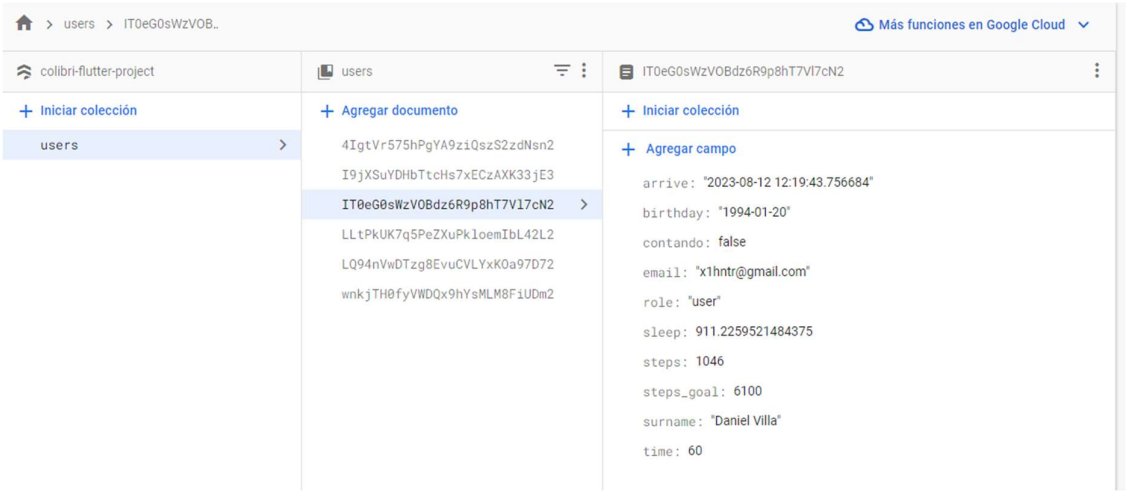



Figura 3.10. Campos almacenados de los usuarios a través del sistema

## 3.4 Pruebas de validación

Con el objetivo de validar cada uno de los módulos y las funcionalidades con las cuales cuenta el sistema, se ha puesto el mismo a prueba con cinco habitantes en la ciudad de Quito dentro del casco urbano, los cuales registraron su actividad por 15 días de forma continua para el rol usuario, y 5 personas con el rol administrador los cuales registraron las actividades de los usuarios.

En la Figura 3.11, se muestra un ejemplo de un usuario probando el prototipo de aplicación en el Parque Navarro ubicado al centro norte de Quito,

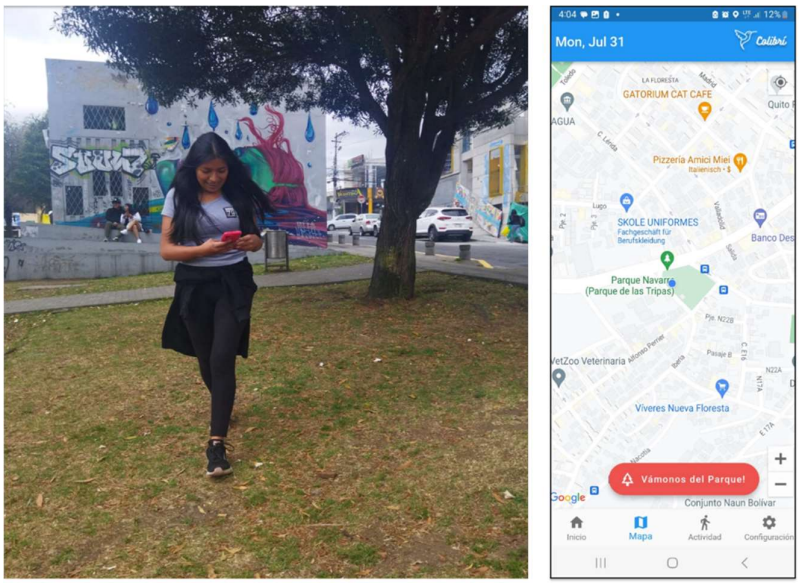



Figura 3.11. Pruebas funcionamiento

Con el objetivo de obtener las impresiones de los usuarios y luego de haber terminado el tiempo de prueba del prototipo de aplicación móvil es necesario realizar una encuesta a cada uno de los usuarios registrados en el sistema.

La Tabla 3.1 muestra las preguntas referentes a las pruebas de las funcionalidades disponibles para el usuario, las cuales son el registro, autenticación, navegación en el sistema, registro de actividades y tiempo en áreas verdes tomadas del formulario del ANEXO III

Tabla 3.1. Resultado de las encuestas realizadas a los usuarios con rol "*user*"

| | Pregunta | Si % | No % |
|---|---|---|---|
| 1 | ¿Pudo instalar la aplicación en su dispositivo móvil? | 100 | 0 |
| 2 | ¿Pudo abrir la aplicación móvil? | 100 | 0 |
| 3 | ¿El prototipo de aplicación móvil permite el registro de nuevos usuarios? | 100 | 0 |
| 4 | ¿El prototipo de aplicación móvil permite el cambio de contraseña? | 100 | 0 |
| 5 | ¿El prototipo de aplicación móvil permite el ingreso del usuario autenticado? | 100 | 0 |
| 6 | ¿El prototipo de aplicación móvil permite visualizar la cantidad de pasos realizados en día? | 100 | 0 |
| 7 | ¿El prototipo de aplicación móvil permite visualizar la cantidad horas de sueño? | 100 | 0 |
| 8 | ¿El prototipo de aplicación móvil permite visualizar el tiempo en áreas verdes? | 100 | 0 |
| 9 | ¿El prototipo de aplicación móvil permite navegar entre las diferentes pantallas que forman el sistema? | 100 | 0 |
| 10 | ¿El prototipo de aplicación móvil permite visualizar el mapa tomando la ubicación actual del usuario? | 100 | 0 |
| 11 | ¿El prototipo de aplicación móvil permite iniciar el conteo de tiempo dentro de un parque? | 100 | 0 |
| 12 | ¿El prototipo de aplicación móvil permite parar el conteo de tiempo en un área verde al momento que el usuario así lo desee? | 100 | 0 |
| 13 | ¿El prototipo de aplicación móvil permite visualizar la pantalla de actividad física? | 100 | 0 |
| 14 | ¿El prototipo de aplicación móvil permite visualizar los pasos realizados en el día comparados con los objetivos propuestos? | 100 | 0 |
| 15 | ¿El prototipo de aplicación móvil permite modificar los pasos propuestos como objetivos por el usuario? | 100 | 0 |
| 16 | ¿El prototipo de aplicación móvil permite visualizar la pantalla de Configuración e Información? | 100 | 0 |
| 17 | ¿El prototipo de aplicación móvil permite visualizar el documento "Guía de usuario"? | 100 | 0 |
| 18 | ¿El prototipo de aplicación móvil permite visualizar el documento "Términos y Condiciones"? | 100 | 0 |
| 19 | ¿El prototipo de aplicación móvil permite cerrar el sistema? | 100 | 0 |

Con el fin de obtener las impresiones de los administradores y luego de haber terminado el tiempo de prueba del prototipo de aplicación móvil, es necesario realizar una encuesta a cada uno de los administradores encargados de monitorear la actividad diaria en el panel de administración de la aplicación.

En la Figura *3.12* se muestra el panel de administración en el cual se visualiza las mediciones promedio de los usuarios registrados en el sistema para el día 24 de Julio del 2023.


Figura 3.12. Panel de administración

En la Tabla 3.2, muestra la recolección de datos de los usuarios registrados en el sistema adquiridos desde el módulo de administrador.

Tabla 3.2. Muestra de resultados promedios para el día 24 de Julio 2023

| Res | Fecha | Sleep (min) | Steps (pasos) | Time (min) |
|---|---|---|---|---|
| 1 | 24/07 | 372 | 3250 | 0 |
| 2 | 24/07 | 192 | 1417 | 0 |
| 3 | 24/07 | 425 | 3226 | 0 |
| 4 | 24/07 | 362 | 8420 | 0 |
| 5 | 24/07 | 432 | 4616 | 16 |
| | | 356.6 | 4185.8 | 3.2 |

La Tabla *3.3*. Resultado de las encuestas realizadas a los usuarios con rol “admin” muestra las pruebas de las funcionalidades que registraron los usuarios en la encuesta realizada mediante un formulario alojado en el ANEXO *IV*

Tabla 3.3. Resultado de las encuestas realizadas a los usuarios con rol “*admin*”

| | Pregunta | Si % | No % |
|---|---|---|---|
| 1 | ¿Pudo instalar la aplicación en su dispositivo móvil? | 100 | 0 |
| 2 | ¿Pudo abrir la aplicación móvil? | 100 | 0 |
| 3 | ¿El prototipo de aplicación móvil permite visualizar la cantidad de usuarios registrados? | 100 | 0 |
| 4 | ¿El prototipo de aplicación móvil permite visualizar la cantidad promedio de pasos realizados en día de los usuarios registrados? | 100 | 0 |
| 5 | ¿El prototipo de aplicación móvil permite visualizar el tiempo de sueño promedio de los usuarios registrados? | 100 | 0 |
| 6 | ¿El prototipo de aplicación móvil permite visualizar el tiempo promedio en áreas verdes de los usuarios registrados? | 100 | 0 |
| 7 | ¿El prototipo de aplicación móvil permite cerrar la sesión actual en el sistema? | 100 | 0 |

## 3.5 Resultados

En la presente sección se indican los resultados obtenidos luego de la recolección de información de la actividad de los usuarios registrados en el sistema, entre 24 de Julio y el 7 de agosto del 2023.

En la Tabla *3.4*, se presentan los promedios de las variables recolectados a lo largo del intervalo propuesto para la fase de pruebas para los 5 usuarios registrados.

Tabla 3.4. Resultados promedios

| N° | Fecha | Pasos | Sueño (min) | T. Areas V (min) |
|---|---|---|---|---|
| 1 | 24/07 | 4185.8 | 356.6 | 16 |
| 2 | 25/07 | 4841.6 | 382.4 | 0 |
| 3 | 26/07 | 5208.8 | 378.2 | 0 |
| 4 | 27/07 | 5482.8 | 375.8 | 0 |
| 5 | 28/07 | 3760.4 | 301.6 | 0 |
| 6 | 29/07 | 5382.8 | 447.6 | 27.6 |
| 7 | 30/07 | 6367.6 | 454.4 | 45 |
| 8 | 31/07 | 3530.2 | 356.6 | 0 |
| 9 | 01/08 | 4386.6 | 378 | 0 |
| 10 | 02/08 | 5652.6 | 333.6 | 0 |
| 11 | 03/08 | 4404.2 | 320.4 | 0 |
| 12 | 04/08 | 3139.4 | 435.6 | 0 |
| 13 | 05/08 | 4136.4 | 403.2 | 83 |
| 14 | 06/08 | 6367.6 | 318 | 30 |
| 15 | 07/08 | 4206.8 | 396 | 0 |
| Promedio | | 4736.9 | 375.87 | 13.44 |

### 3.5.1 Análisis de resultados

En esta sección se realiza el análisis de resultados de las mediciones obtenidas a los usuarios registrados durante el periodo de prueba. Estos resultados serán contrastados con los valores medidos para una población sana los cuales oscilan entre 4000 a 18000 pasos diarios. [25]

En la Figura *3.13*, se indica a través de un histograma el promedio de pasos diarios realizados por los usuarios registrados en el sistema a lo largo del intervalo de tiempo propuesto.

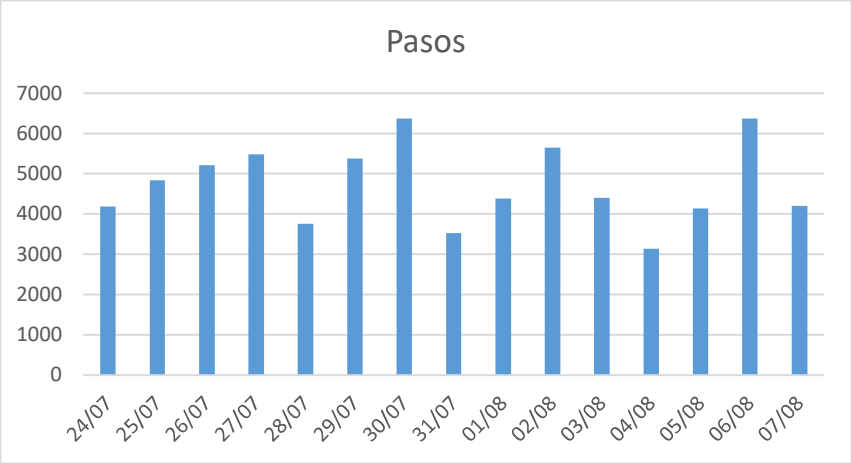



Figura 3.13. Histograma del promedio de pasos

En la Figura *3.14*, se indica a través de un histograma el promedio de horas de sueño diarios realizados por los usuarios registrados en el sistema a lo largo del intervalo de tiempo propuesto.

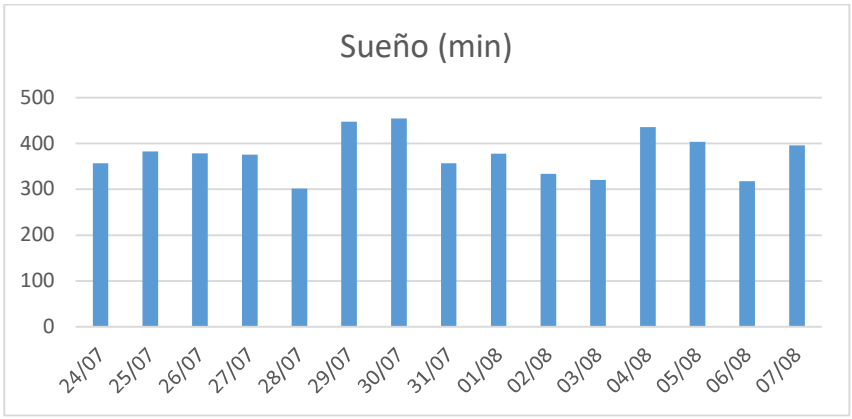



Figura 3.14. Histograma del promedio del tiempo de sueño en minutos

En la Figura *3.15*, se indica a través de un histograma la suma de las mediciones de tiempo en áreas verdes a diario de parte de los usuarios registrados en el sistema a lo largo del intervalo propuesto para pruebas.

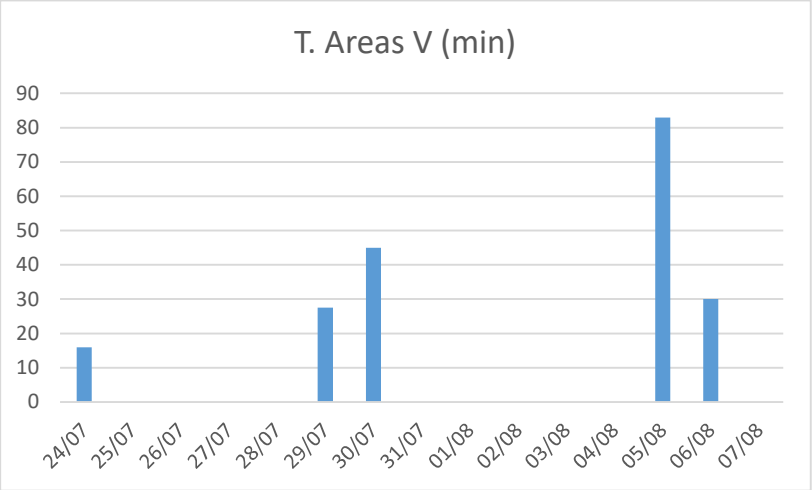



Figura 3.15. Histograma de la suma de tiempo pasado en áreas verdes y parques

En la Figura 3.16, se muestra la cantidad de pasos promedio de los usuarios registrados en el sistema en comparación con el número de pasos recomendados por la Organización Mundial de la Salud. De lo cual se puede visualizar que el promedio de pasos de los usuarios no supera al menos la mitad del número de pasos recomendados.


Figura 3.16. Promedio de pasos medidos comparado con número de pasos recomendados

En la Figura 3.17, se muestra la cantidad de horas de sueño de los usuarios registrados en el sistema en comparación con el número de pasos recomendados por la Organización Mundial de la Salud para una persona adulta. De lo cual se puede visualizar que los usuarios registrados no cumplen con el número de horas de sueño recomendadas.


Figura 3.17. Promedio de horas de sueño comparados con las horas recomendadas

## 3.6 Corrección de errores

Durante la fase de desarrollo, integración y pruebas del prototipo de aplicación se presentaron múltiples errores los cuales fueron corregidos al largo de las fases previas a esta sección los cuales se describen y la solución implementada en la Tabla *3.5*

Tabla 3.5. Errores del sistema y soluciones implementadas

| | Error | Solución |
|---|---|---|
| 1 | No se guarda el objetivo de pasos del usuario en el sistema. | Almacenar el objetivo de pasos en la base de datos y consultarlo cada vez que se renderiza el componente. |
| 2 | No se actualizan los valores del número de pasos. | Asegurar los permisos para importar los valores desde Google Fit. |
| | El mapa no se visualiza la primera vez que se ingresa a la pantalla correspondiente | Se fija una ubicación aleatoria dentro de la ciudad de Quito para que el mapa se renderice y luego se actualiza la ubicación por la actual y se modifica la animación para que se muestre como cambia la ubicación. |
| 3 | El botón de inicio y finalización de conteo de áreas verdes no se diferencian entre sí. | Se cambio el color y la leyenda del botón. Para iniciar se cambia a "Vamos al parque" de color verde y para la finalizar "Vámonos del parque" de color rojo. |
| 4 | El sistema permite iniciar el conteo de tiempo en áreas verdes fuera de estos. | Reducir el radio de proximidad al parque a 10 metros. |
| 5 | Al iniciar el conteo el sistema muestra el anterior parque o área verde en la que usuario se encontraba. | Cambiar las funciones referentes a la ubicación a funciones asíncronas con el objetivo de que al momento de renderizar el componente tomar la ubicación actual y esperar a obtener la información del área verde real en la que se encuentra actualmente el usuario. |

## 3.7 Actualización final del tablero Kanban

Finalmente, en la Figura *3.18*, se presenta la última actualización del tablero Kanban, donde se actualizan todas las actividades propuestas al inicio del desarrollo de este trabajo de integración curricular.

Cabe destacar que para esta última actualización todas las actividades propuestas en la columna por hacer y en procesos pasaron a la columna de actividades realizadas.

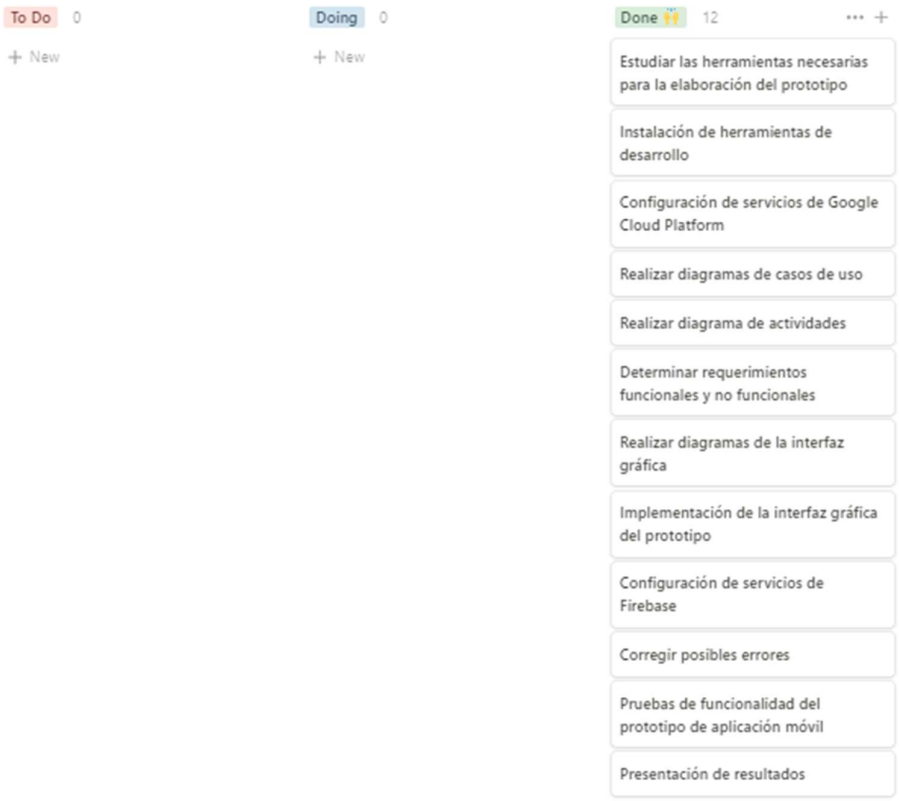



Figura 3.18. Tablero Kanban final del desarrollo del sistema

## 3.8 Conclusiones

El presente trabajo de integración curricular ha permitido implementar un prototipo de aplicación para dispositivos móviles con sistema operativo Android, el cual obtiene y almacena información relevante en el contexto de una ciudad inteligente referente a las actividades diarias de los usuarios registrados.

La recopilación de información representa una parte importante cuando se desea implementar una ciudad inteligente y sostenible, debido a que permiten encontrar patrones en el comportamiento de los ciudadanos. Esta información está directamente relacionada a los servicios que esta ciudad ofrece y la infraestructura que debe construir para mejorar el bienestar ciudadano.

Las validaciones implementadas en el prototipo con respecto al correcto ingreso de la información solicitada en el formulario de registro evitan el almacenamiento de información errónea en la base de datos.

Los servicios prestados por *Google Cloud Platform* representaron una parte importante dentro del desarrollo del presente trabajo, ya que, debido a estos se pudieron implementar funciones necesarias para cumplir los requerimientos propuestos en el diseño del prototipo.

La plataforma Firebase ofrece facilidad en el desarrollo de aplicaciones web y móviles escalables, debido a que presta diversos servicios como el almacenamiento de información y archivos, autenticación, bases de datos, funciones, entre otros. Estos servicios han permitido implementar algunas de los requerimientos con los cuales cuenta el prototipo de aplicación móvil.

El marco de desarrollo *Flutter* cuenta con un sin número de widgets y dependencias útiles para el desarrollo de aplicaciones móviles, webs y de escritorios. Estos permiten la integración de una forma sencilla con servicios externos para el almacenamiento, procesamiento y presentación de información.

## 3.9 Recomendaciones

Debido a que el sistema admite únicamente el registro con correo electrónico y contraseña, se recomienda para futuros desarrollos admitir otros proveedores adicionales para el proceso de autenticación tales como número telefónico, Facebook, Gmail, Twitter, etc.

Es recomendable incluir dentro del módulo mapa una sección en la cual el usuario pueda visualizar una lista de los parques y áreas verdes más cercanos, además de la información de cada uno de ellos con el fin de ayudar al usuario a encontrar nuevos lugares.

Con el fin de mantener actualizada la información de actividad física del usuario, se recomienda implementar un sistema de notificaciones de tipo *push* que recuerde al usuario ingresar la información requerida por el sistema de forma continua.

Para futuras implementaciones se recomienda añadir en el sistema un módulo de historial, con el objetivo de que el usuario pueda visualizar la información registrada a lo largo de un intervalo de tiempo.

Para mejorar la adquisición de las variables de actividad física es recomendable desarrollar las funciones adecuadas para conectar la aplicación con dispositivos externos como relojes inteligentes.

Para el correcto funcionamiento del prototipo de aplicación móvil se recomienda la instalación y uso en dispositivos móviles con sistemas operativos Android con una versión igual o superior a la 11.

Dentro del módulo de administración se podría incluir estadísticas referentes a los parques más concurridos por los usuarios y el horario en los cuales están más utilizados.

# 4 REFERENCIAS BIBLIOGRÁFICAS


[1] T. Pflanzner, K. Z. Leszko, and A. Kertesz, “SUMMON: Gathering smart city data to support IoT-Fog-Cloud simulations,” *2018 3rd International Conference on Fog and Mobile Edge Computing, FMEC 2018*, pp. 71–78, May 2018, doi: 10.1109/FMEC.2018.8364047.

[2] I. H. Sarker, “Smart City Data Science: Towards data-driven smart cities with open research issues,” *Internet of Things*, vol. 19, p. 100528, Aug. 2022, doi: 10.1016/J.IOT.2022.100528.

[3] K. Ahmad, M. Maabreh, M. Ghaly, K. Khan, J. Qadir, and A. Al-Fuqaha, “Developing future human-centered smart cities: Critical analysis of smart city security, Data management, and Ethical challenges,” *Comput Sci Rev*, vol. 43, p. 100452, Feb. 2022, doi: 10.1016/J.COSREV.2021.100452.

[4] A. Rivoira *et al.*, “Automatización de la Medición de Software para Flotas Dinámicas mediante un Modelo de Calidad Mixto para la Movilidad en Smart Cities.”

[5] A. y T. E. Ministerio del Ambiente, “MAE trabaja en programas de mitigación y adaptación para reducir emisiones de Co2 en Ecuado,” *https://www.ambiente.gob.ec/mae-trabaja-en-programas-de-mitigacion-y-adaptacion-para-reducir-emisiones-de-co2-en-ecuador/#:~:text=%C2%BFSabes%20cu%C3%A1nto%20CO2%20produce%20Ecuador,de%20emisiones%20a%20nivel%20mundial.*, 2022.

[6] Ministerio de Transporte y Obras Públicas, “El MTOP impulsa el Plan Nacional de Movilidad Urbana Sostenible,” *https://www.obraspublicas.gob.ec/el-mtop-impulsa-el-plan-nacional-de-movilidad-urbana-sostenible/*, 2023.

[7] R. C. E. M. y C. L. A. Monzón, “‘PMUS: Guía práctica para la elaboración e implantación de planes de movilidad urbana sostenible’.,” Madrid, 2016.

[8] P. F. Saint-Maurice *et al.*, “Association of Daily Step Count and Step Intensity with Mortality among US Adults,” *JAMA - Journal of the American Medical Association*, vol. 323, no. 12, pp. 1151–1160, Mar. 2020, doi: 10.1001/jama.2020.1382.

[9] Dabitzan Hiriak, “Principios Red Ciudades Que Caminan,” *https://ciudadesquecaminan.org/*, 2023.

[10] Redacción National Geographic, “Ciudades inteligentes, sostenibles y verdes: ¿qué significa cada una y cuáles son sus diferencias?,” *https://www.nationalgeographicla.com/medio-ambiente/2022/10/ciudades-inteligentes-sostenibles-y-verdes-que-significa-cada-una-y-cuales-son-sus-diferencias*, Oct. 28, 2022.

[11] Quito Informa, “Quito es la ciudad con más espacios verdes por habitante,” *http://www.quitoinforma.gob.ec/2018/02/08/quito-es-la-ciudad-con-mas-espacios-verdes-por-habitante/#:~:text=Quito%20se%20mantiene%20como%20la,par%C3%A1metro%20internacional%209%20metros%20cuadrados.*, Feb. 02, 2018.

[12] Michael Thomsen, “Dart-lang,” *https://github.com/dart-lang/site-www/blob/main/src/overview.md*, Apr. 30, 2023.

[13] Flutter, “Flutter,” *https://esflutter.dev/*, 2022.

[14] Google, “Firebase Docs,” *https://firebase.google.com/docs/guides?hl=es-419*, 2023.

[15] Google, “Firebase Authentication,” *https://firebase.google.com/docs/auth?hl=es-419*, 2023.

[16] Google, “Cloud Storage Firebase,” *https://firebase.google.com/docs/storage?hl=es-419*, 2023.

[17] Google, “Firebase Realtime Database,” *https://firebase.google.com/docs/database?hl=es-419*, 2022.

[18] OpenBootcamp, “¿Qué es VSC (Visual Studio Code)?,” *https://open-bootcamp.com/aprender-programar/tipos-de-ide-visual-studio-code*, 2022.

[19] Reclu IT, “¿Qué es Visual Studio Code?,” *https://recluit.com/que-es-visual-studio-code/*, Apr. 12, 2021.

[20] Romain Vimont, “Scrcpy (v2.0),” *https://github.com/Genymobile/scrcpy*, 2023.

[21] Google, “Google Fit,” *https://support.google.com/fit/answer/6075067?hl=es-419&co=GENIE.Platform%3DAndroid*, Jul. 01, 2023.

[22] Google, “Google Cloud Platform,” *https://cloud.google.com/docs/overview?hl=es-419*.

[23] Google, “Places API,” *https://developers.google.com/maps/documentation/places/web-service/overview?hl=es-419*, Jul. 01, 2023.

[24] Moisés Dias, “Cidades Inteligentes,” 2022.

[25] C. Tudor-Locke *et al.*, “How many steps/day are enough? for adults,” *International Journal of Behavioral Nutrition and Physical Activity*, vol. 8, Jul. 2011, doi: 10.1186/1479-5868-8-79.

# 5 ANEXOS

ANEXO *I*.1 ENTREVISTA PROYECTO URSOVERDE

ANEXO *II*.1 FORMATO DE PREGUNTAS REALIZADAS AL EQUIPO DE TRABAJO DE URSO VERDE

ANEXO *III*.1 ENCUESTA FUNCIONALIDAD DEL USUARIO

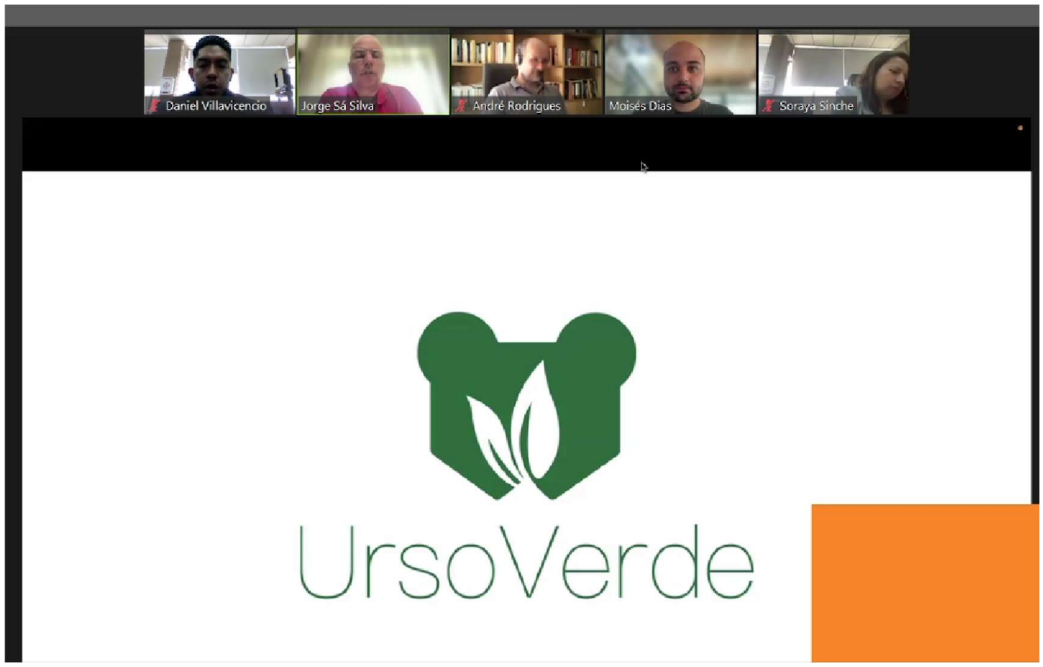
ANEXO *IV*.1 ENCUESTA FUNCIONALIDAD ADMINISTRADOR

ANEXO *V*.1 CÓDIGO FUENTE DEL PROTOTIPO DE APLICACIÓN MÓVIL

ANEXO *VI*.1 CÓDIGO FUENTE BACKEND DEL SISTEMA

ANEXO *VII*.1 MANUAL DE USUARIO Y APLICATIVO MÓVIL

ANEXO I.1 ENTREVISTA PROYECTO URSOVERDE



ANEXO II.1 FORMATO DE PREGUNTAS REALIZADAS AL EQUIPO DE TRABAJO DE URSO VERDE

| N° | PREGUNTA |
|---|---|
| 1 | ¿Es útil una aplicación que permita visualizar las variables de actividades diarias de usuarios? |
| 2 | ¿Considera útil que en la interfaz del usuario se pueda visualizar la información personal de este? |
| 3 | ¿Considera necesario que el usuario pueda visualizar la información de la ciudad en la cual se encuentra? |
| 4 | ¿Considera útil que la aplicación se encuentre distribuida en diferentes pantallas divididas en Inicio, Mapa, Actividad y Configuración? |
| 5 | ¿Considera adecuado que la información diaria del usuario se encuentre en la pantalla de Inicio? |
| 6 | ¿Considera útil llevar un registro del tiempo en la que el usuario se encuentra en un área verde? |
| 7 | ¿Considera útil el registro de la cantidad de pasos que el usuario realiza en un lapso de 24 horas? |
| 8 | ¿Considera útil el registro de la cantidad de tiempo que el usuario durmió en las últimas 24 horas? |

ANEXO III.1 ENCUESTA FUNCIONALIDAD DEL USUARIO

https://forms.gle/kpLiGT89ox3NcNpU7

ANEXO IV.1 ENCUESTA FUNCIONALIDAD ADMINISTRADOR

https://forms.gle/J8kGYprzytppMuaJ9

ANEXO V.1 CÓDIGO FUENTE DEL PROTOTIPO DE APLICACIÓN MÓVIL

https://github.com/x1hntr/Colibri

ANEXO VI.1 CÓDIGO FUENTE BACKEND DEL SISTEMA

https://github.com/x1hntr/Colibri-Backend

ANEXO VII.1 MANUAL DE USUARIO Y APLICATIVO MÓVIL

https://drive.google.com/drive/folders/1oJ7Q9K3LXFThp0lEgOg4_HVjS_wDp_SG?usp=drive_link