
# UNIVERSIDAD TÉCNICA DEL NORTE

## FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

## CARRERA DE TELECOMUNICACIONES

## INFORME FINAL DEL TRABAJO DE INTEGRACIÓN CURRICULAR, MODALIDAD PRESENCIAL

### TEMA:

SISTEMA DE RECOLECCIÓN DE RESIDUOS SÓLIDOS URBANOS (RSU) BASADO EN OPTIMIZACIÓN DE RUTAS Y NOTIFICACIÓN EN TIEMPO REAL PARA EL BARRIO CENTENARIO, CIUDAD DE SAN GABRIEL

**Trabajo de Titulación previo a la obtención del título de Ingeniero en Telecomunicaciones**

**Línea de investigación:** Desarrollo, aplicación de software y cybersecurity (seguridad cibernética)

**AUTOR:**

Cristian Iván Quelal Guadir

**DIRECTOR:**

Ing. Jaime Roberto Michilena Calderón, MsC.

**Ibarra, julio 2024**

# UNIVERSIDAD TÉCNICA DEL NORTE
## BIBLIOTECA UNIVERSITARIA

## AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD

### 1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

| DATOS DE CONTACTO | | | |
|---|---|---|---|
| **CÉDULA DE IDENTIDAD:** | 0401865951 | | |
| **APELLIDOS Y NOMBRES:** | Quelal Guadir Cristian Iván | | |
| **DIRECCIÓN:** | San Gabriel, Carchi | | |
| **EMAIL:** | ciquelalg@utn.edu.ec \| quelal.cristian21@gmail.com | | |
| **TELÉFONO FIJO:** | XXXXXXX | **TEL. MOVIL:** | 0988330543 |

| DATOS DE LA OBRA | |
|---|---|
| **TÍTULO:** | SISTEMA DE RECOLECCIÓN DE RESIDUOS SÓLIDOS URBANOS (RSU) BASADO EN OPTIMIZACIÓN DE RUTAS Y NOTIFICACIÓN EN TIEMPO REAL PARA EL BARRIO CENTENARIO, CIUDAD DE SAN GABRIEL |
| **AUTOR:** | Quelal Guadir Cristian Iván |
| **FECHA DE APROBACIÓN:** | 29/07/2024 |
| **CARRERA/PROGRAMA:** | ■ GRADO ☐ POSGRADO |
| **TITULO POR EL QUE OPTA:** | Ingeniero en Telecomunicaciones |
| **ASESOR/DIRECTOR:** | Ing. Mauricio Domínguez MSc./Ing. Jaime Roberto Michilena Calderón MSc. |

### 2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 29 días del mes de julio de 2024

**EL AUTOR:**

.....

Quelal Guadir Cristian Iván

# CERTIFICACIÓN DEL DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Ibarra, 29 de julio de 2024

Ing. Jaime Michilena Msc.
**DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR**

**CERTIFICA:**

Haber revisado el presente informe final del trabajo de Integración Curricular, el mismo que se ajusta a las normas vigentes de la Universidad Técnica del Norte; en consecuencia, autorizo su presentación para los fines legales pertinentes.

.....
*Ing. Jaime Roberto Michilena Calderón MSc.*
*C.C.: 1002198438*

# APROBACIÓN DEL COMITÉ CALIFICADOR

El Comité Calificador del trabajo de Integración Curricular "SISTEMA DE RECOLECCIÓN DE RESIDUOS SÓLIDOS URBANOS (RSU) BASADO EN OPTIMIZACIÓN DE RUTAS Y NOTIFICACIÓN EN TIEMPO REAL PARA EL BARRIO CENTENARIO, CIUDAD DE SAN GABRIEL" elaborado por CRISTIAN IVÁN QUELAL GUADIR, previo a la obtención del título de INGENIERO EN TELECOMUNICACIONES, aprueba el presente informe de investigación en nombre de la Universidad Técnica del Norte:

.....

Ing. Jaime Roberto Michilena Calderón MSc.
C.C.: 1002198438

.....

Ing. Hernán Mauricio Domínguez Limaico MSc.
C.C.: 1002379301

# DEDICATORIA

*A mis queridos padres, Luis Quelal y Rosa Guadir, pilares fundamentales en mi vida. Su amor incondicional, enseñanzas y apoyo constante me han impulsado a perseguir mis sueños. Los valores que me han inculcado son la base de cada meta alcanzada en mi vida.*

*A mis hermanos: Carlitos, Miguelito, Marinita, Ximenita y Fernandita por su amor y aliento incansable en cada etapa de mi vida.*

*A mis primas Kattyta y Moni por su cariño y valiosos consejos, que me han motivado a seguir adelante sin desfallecer.*

*A mis tías y tíos por su respaldo y ser ejemplo de trabajo en mi desarrollo profesional.*

*A ustedes que creyeron en mí, les dedico con todo mi corazón este trabajo.*

***Quelal Guadir Cristián Iván***

# AGRADECIMIENTO

*Al concluir esta etapa significativa de mi vida, en primer lugar, agradezco a Dios por bendecirme con salud y fortaleza para alcanzar una meta más.*

*Agradezco profundamente a mis padres, por su sacrificio incansable y su apoyo inquebrantable. Este logro académico es un reflejo de su amor y esfuerzo.*

*A mis amigos: Edin, Erik, Juan Diego y Marlon, acompañantes en este viaje académico. Su amistad y el conocimiento compartido han enriquecido mi experiencia universitaria.*

*A los docentes de la carrera de Telecomunicaciones, por su compromiso en transmitir no solo conocimientos, sino también la pasión por nuestra profesión.*

*En especial al Ing. Jaime Michilena, director de mi trabajo de titulación, y al Ing. Mauricio Domínguez, asesor, por brindarme su valiosa guía, fundamental en el desarrollo de este proyecto.*

*Al GADM Montufar, particularmente al departamento de Protección Ambiental, por su disposición y colaboración, elementos clave para la realización exitosa de este trabajo.*

*A todos ustedes, mi eterno agradecimiento por ser parte integral de este logro académico y personal.*

***Quelal Guadir Cristián Iván***

# RESUMEN EJECUTIVO

El presente trabajo de titulación aborda la problemática de recolección de residuos sólidos urbanos (RSU) en el barrio Centenario de San Gabriel, Carchi, Ecuador; en donde el crecimiento poblacional ha incrementado la generación de estos, ocasionando problemas de contaminación debido a deficiencias en el sistema actual de recolección, por tanto, se diseña un sistema que optimice este proceso mediante una aplicación móvil para la interacción en tiempo real entre moradores del sector y recolectores.

Se implementa la metodología en cascada, iniciando con la fase de requerimientos, donde se determinan las necesidades del sistema. En la fase de diseño, se integran tecnologías como GPS, 4G LTE, servicios en la nube y se desarrolla una aplicación móvil. Sucesivamente, en la fase de codificación y pruebas unitarias se desarrolla el sistema cumpliendo los requerimientos y verificando individualmente los componentes. Posteriormente, en la fase de integración y pruebas, se unen los diferentes elementos y se realizan pruebas del sistema completo, verificando la optimización del proceso de recolección. Finalmente, para la fase de operación y mantenimiento, se elabora un plan de implementación.

Los resultados validan el funcionamiento óptimo del sistema, demostrando su capacidad para optimizar rutas de recolección y notificar a los clientes sobre el proceso de su solicitud en tiempo real, mejorando la interacción cliente-recolector. Por otra parte, el análisis costo/beneficio proyecta beneficios operativos, sociales, ambientales y económicos, aunque se requiere de una inversión inicial para la puesta en marcha del sistema, se prevé su viabilidad a largo. Asimismo, el sistema se presenta como un modelo replicable para la transformación digital urbana, contribuyendo a las iniciativas de una ciudad inteligente.

**Palabras clave**: Residuos Sólidos Urbanos, recolección, rutas optimas, seguimiento GPS, 4G LTE, servicios de mapas, Mapbox, aplicación móvil, interacción en tiempo real, metodología en cascada, impacto ambiental.

# ABSTRACT

This project addresses the problem of urban solid waste collection in the Centenario neighborhood of San Gabriel, Carchi, Ecuador, where population growth has increased waste generation, causing contamination problems due to deficiencies in the current collection system. Therefore, a system is designed to optimize this process through a mobile application for real-time interaction between residents and collectors.

The waterfall methodology is implemented, starting with the requirements phase, where the system needs are determined. In the design phase, technologies such as GPS, 4G LTE, cloud services, and a mobile application are integrated. Subsequently, in the coding and unit testing phase, the system is developed to meet the requirements and individual components are verified. Later, in the integration and testing phase, the different elements are combined, and tests of the complete system are carried out, verifying the optimization of the collection process. Finally, for the operation and maintenance phase, an implementation plan is developed.

The results validate the optimal functioning of the system, demonstrating its ability to optimize collection routes and notify clients about the status of their request in real time, improving client-collector interaction. On the other hand, the cost/benefit analysis projects operational, social, environmental, and economic benefits. Although an initial investment is required to launch the system, its long-term viability is foreseen. Likewise, the system is presented as a replicable model for urban digital transformation, contributing to smart city initiatives.

**Keywords:** Municipal Solid Waste, waste collection, optimal routes, GPS tracking, 4G LTE, map services, Mapbox, mobile application, real-time interaction, cascading methodology, environmental impact.

# ÍNDICE DE CONTENIDOS

# ÍNDICE DE TABLAS

# ÍNDICE DE FIGURAS

# CAPÍTULO I: ANTECEDENTES

En este capítulo se presentan los apartados que permiten contextualizar el desarrollo del presente trabajo de integración curricular. Se exponen de forma clara y consistente: el tema con el que se identifica, los objetivos generales y específicos planteados, el alcance de este proyecto y la justificación que lo respalda. Estos antecedentes permiten obtener una visión general para enmarcar adecuadamente el desarrollo del presente proyecto.

## 1.1 Tema

Sistema de recolección de Residuos Sólidos Urbanos (RSU) basado en optimización de rutas y notificación en tiempo real para el barrio Centenario, ciudad de San Gabriel.

## 1.2 Planteamiento del Problema

A nivel global, la gestión inadecuada de los residuos sólidos urbanos (RSU) plantea diversos problemas ambientales y de salud pública. La generación excesiva de basura, la falta de infraestructura adecuada para su recolección y disposición final, así como la ausencia de sistemas eficientes de gestión, contribuyen a la contaminación ambiental, visual y a la propagación de enfermedades. Según Kaza et al. (2018) en el informe What a Whaste 2.0 (Los desechos 2.0) del Banco Mundial, en el mundo se generan anualmente 2010 millones de toneladas de desechos sólidos municipales, y al menos el 33 % de ellos no se gestionan adecuadamente poniendo en riesgo al medio ambiente. El Banco Mundial estimó que la generación de desechos aumentará a 3400 millones de toneladas en 2050. Estos problemas requieren soluciones integrales y sostenibles para garantizar un ambiente saludable y la calidad de vida de los habitantes.

En Ecuador, a nivel urbano, se generó un promedio de 0,83 kg/día de desechos por habitante en el 2020. La provincia que tiene la producción per cápita de desechos sólidos más elevada es Guayas con un total de 25 cantones, con 1,04 kg/día, le sigue Pichincha (8 cantones) con 0,88 kg/día. La provincia del Carchi tiene un valor alto de 0.63 kg/día, considerando que

cuenta con 6 cantones siendo una cantidad menor en comparación a los otros cantones mencionados, entre los cantones pertenecientes a la provincia del Carchi está el cantón Montufar en donde se encuentra el barrio Centenario (Instituto Nacional de Estadística y Censo (INEC) et al., 2021).

De acuerdo con GADM Montúfar (2020), mismo que hace referencia al Plan de Desarrollo y Ordenamiento Territorial (PDOT) 2020 – 2035 del Gobierno Autónomo Descentralizado Municipal de Montúfar, detalla que el nivel de percepción del servicio de residuos sólidos por parte de la ciudadanía es del 70.7% en el sector urbano, mientras que en el sector rural es del 61.3%, por lo que se debe incorporar mejoras en el proceso desde su generación, almacenamiento, recolección, transporte, tratamiento y disposición final en el territorio cantonal.

El barrio Centenario de la ciudad de San Gabriel perteneciente al cantón Montufar, a cargo de la administración del GADM Montufar, enfrenta una problemática significativa en la recolección de los residuos sólidos urbanos generados en cada domicilio debido a que, en base a mi propia experiencia en observación directa, en toda la ciudad de San Gabriel, que incluye al barrio Centenario, el proceso de recolección de RSU se basa en la recolección puerta a puerta en horarios matutinos que son irregulares, aunque algunos de los recolectores cuentan con un sistema de notificación sonora(canción), esta no siempre está funcional, ocasionando que los moradores del sector saquen sus residuos en fundas a las puertas de las viviendas, en algunos casos con mucho tiempo de anticipación a que llegue el recolector o después de que este haya realizado su recorrido por un determinado lugar, esto genera que la fauna urbana (perros y gatos) del sector en su búsqueda de comida, rompan las fundas de residuos sólidos que se encuentran afuera de los domicilios y estos residuos se dispersen por el sector, especialmente cuando hay viento, de tal manera que no son recolectados al no estar adecuadamente almacenados en fundas, provocando que roedores como ratas/ratones, plagas como cucarachas,

entre otros, se vean atraídos, se alimentan y se reproducen rápidamente, generando proliferación de plagas y contaminación que a futuro podría ocasionar graves problemas de salud. A esto se suma que según moradores del barrio Centenario, existe un crecimiento continuo de la población del barrio, el cual pertenece al cantón Montufar de la provincia del Carchi, una zona dedicada exclusivamente a la agricultura y que ofrece varias plazas de trabajo en esta área, considerando que Carchi es una provincia fronteriza con el país de Colombia, muchas personas de nacionalidad colombiana lo elijen al barrio Centenario como un buen lugar para residir y trabajar debido a que tiene un costo bajo en cuotas mensuales de arriendos, de esta forma se tiene mayor población y consecuentemente se genera mayores cantidades de residuos, por tanto, los problemas se hacen más evidentes y requieren una mayor atención para crear soluciones efectivas/sostenibles que mejoren el proceso de recolección de los residuos sólidos urbanos (RSU) que actualmente se lleva a cabo en tiempos irregulares y sin una planificación técnica adecuada, demandando una utilización ineficiente de recursos del GADM Montufar.

Para atender los problemas que enfrenta el barrio Centenario en relación a la recolección de residuos sólidos urbanos; el diseño de un sistema que permita la interacción en tiempo real mediante una aplicación móvil entre los moradores y el recolector de residuos sólidos urbanos, tiene el potencial de transformar el proceso actual de recolección de RSU, proporcionando una plataforma móvil fácil de usar para solicitar la recolección de residuos y dar seguimiento al proceso de manera rápida y eficiente, gestionando un sistema de notificaciones tanto para el controlador del recolector para avisar que está cerca a llegar a un punto de recolección, así como para el usuario para que este sea notificado de que el recolector está a pocos metros del domicilio y este tenga el tiempo adecuado para sacar los residuos sólidos urbanos afuera del domicilio y estos sean debidamente recolectados en un periodo mínimo de tiempo, logrando que la funda de los residuos sólidos urbanos no queden afuera de

los domicilios por extensos periodos de tiempos exponiéndose a que la funda pueda ser rota por la fauna urbana, evitando problemas de contaminación y proliferación de plagas que puedan generar afectaciones en la salud de los moradores. Por tanto, el diseño del sistema ofrece una solución efectiva para mejorar la recolección de residuos sólidos urbanos, promoviendo un entorno más limpio y saludable en el barrio Centenario.

## 1.3 Objetivos

### *1.3.1 Objetivo general*

Diseñar un sistema que optimice el proceso de recolección de Residuos Sólidos Urbanos (RSU) en el barrio Centenario de la ciudad de San Gabriel, provincia del Carchi, a través de una aplicación móvil para la interacción en tiempo real de usuario-recolector.

### *1.3.2 Objetivos específicos*

- Desarrollar un estado del arte de sistemas de gestión de RSU y sus tecnologías: hardware/softwares adecuados para mejorar el proceso de recolección, considerando factores como alcance, eficiencia y costo, permitiendo definir los requerimientos idóneos.
- Realizar un análisis detallado de las necesidades de los habitantes del barrio Centenario en términos de recolección de RSU, considerando factores como la cantidad de residuos generados, la infraestructura existente y los recursos disponibles.
- Implementar el sistema de recolección de Residuos Sólidos Urbanos (RSU) basado en tecnología GPS, que incluya la generación de la ruta optima de recolección, una aplicación móvil para la solicitud y notificación de recolección, visualización de ubicación, la ruta seguida y tiempo aproximado de llegada del recolector al domicilio, utilizando la metodología en cascada.
- Realizar las pruebas de funcionamiento para determinar la viabilidad técnica y económica del sistema de recolección de RSU diseñado, considerando aspectos como:

costos de implementación, el retorno de la inversión y beneficios sociales/ambientales esperados.

## 1.4 Alcance

El presente proyecto tiene como propósito diseñar un sistema de recolección de residuos sólidos urbanos (RSU), basado en optimización de rutas mediante tecnología GPS y una aplicación móvil para la interacción de usuario-recolector, en el barrio Centenario de la ciudad de San Gabriel. El diseño se compone de recursos de hardware, software y técnicos que en conjunto generan un sistema orientado a optimizar el proceso de recolección y reducir el impacto ambiental/social negativo asociado al proceso de recolección inadecuado de RSU en el sector.

En este proyecto, se utiliza la metodología en Cascada para dividir el desarrollo en distintas fases secuenciales que permitirá llevar un proceso ordenado e identificación temprana de posibles problemas. De acuerdo con Instituto Europeo de Postgrado - IEP (2022), se describen las siguientes fases pertenecientes a la metodología en Cascada:

Para la fase inicial que comprende a Análisis de Requerimientos, se realizará una investigación descriptiva que permita documentar las características de la situación actual y necesidades de los moradores del barrio Centenario en términos de recolección de RSU, considerando factores como la cantidad de residuos generados, la infraestructura existente y los recursos disponibles, con el fin de establecer datos estadísticos actualizados que permitan dimensionar el proyecto en el sector. Además, con la elaboración de un estado del arte de los sistemas modernos de gestión de RSU, se definirán los requerimientos de hardware y software adecuados.

Para la segunda fase denominada Diseño, se elaborará una arquitectura óptima que involucre los requerimientos de hardware y software para el sistema de recolección de RSU en el barrio Centenario; se desarrollará una aplicación móvil que requerirá de conexión a internet

mediante WiFi o datos móviles, permitiendo la interacción de usuarios y recolectores en tiempo real, en donde cada usuario dependiendo de su necesidad, ingresara la solicitud de recolección de los residuos sólidos urbanos en un tiempo predeterminado cada día, luego de este tiempo no se receptarán nuevas solicitudes, por tanto, los usuarios que no solicitaron la recolección por la App móvil, deberán esperar al día siguiente para ingresar la solicitud; las solicitudes que sean enviadas en el tiempo establecido, se transmitirán a los servidores en una plataforma en la nube (Cloud) para ser procesadas y generar la ruta de recolección más optima haciendo uso de herramientas tecnológicas adecuadas que contribuyan al diseño de un sistema eficiente, por tanto, el recolector pasará por cada punto de recolección notificado en el aplicativo, siguiendo la ruta más optima generada, evitando de esta forma realizar recorridos innecesarios, logrando así, minimizar los tiempos de recolección; la tecnología GPS (módulo) que implementará el recolector de residuos, determinará la ubicación y ruta del recolector en tiempo real, el módulo hará uso de tecnología de comunicación inalámbrica 4G a partir de su interconexión con el módulo GPRS/GSM del microcontrolador Arduino, garantizando la transmisión confiable de los datos entre los servidores(cloud) y los actores involucrados; en la aplicación móvil se visualizará la ubicación y la ruta que sigue del camión recolector en tiempo real, además se complementará con la generación de notificaciones push para los usuarios sobre tiempo aproximado de llegada y proximidad del recolector al destino de recolección. Considerando casos excepcionales en donde los moradores no dispongan de conexión a internet, el recolector hará uso de su notificación sonora (canción) para informar a los usuarios que se encuentra cerca y el usuario pueda sacar sus residuos a tiempo, logrando evitar que los residuos permanezcan por tiempos extensos en los exteriores de los domicilios y que la fauna urbana rompa las fundas de residuos generando contaminación visual/ambiental y proliferación de plagas.

Para la fase de Implementación, se integrarán los diferentes componentes del sistema de recolección de RSU para generar un prototipo idóneo y eficaz, además, se considerará la

construcción de una cantidad adecuada de prototipos para realizar las respectivas pruebas en la siguiente fase.

Para la fase de Pruebas, se verificará la eficiencia del sistema a partir de métricas que valoren el rendimiento del sistema en diferentes escenarios; se evaluará la viabilidad técnica/económica de la implementación del sistema diseñado mediante aspectos relacionados a la factibilidad y costo/beneficio.

Finalmente, se elaborará un plan de implementación y recomendaciones técnicas para la puesta en marcha y mantenimiento del sistema de recolección de RSU en el barrio Centenario.

## 1.5 Justificación

El diseño de un sistema de recolección de residuos sólidos urbanos (RSU) para el barrio Centenario de la ciudad de San Gabriel, es importante por la necesidad de abordar los problemas sociales y ambientales causados por la gestión inadecuada en términos de recolección de los RSU en el sector. La generación excesiva de basura, la falta de infraestructura adecuada, la ausencia de sistemas eficientes y la contaminación ambiental y visual que se han derivado de las deficiencias encontradas, requieren soluciones integrales y sostenibles para garantizar un ambiente saludable y la calidad de vida de los habitantes en el sector.

Según moradores del barrio, el crecimiento continuo de la población en este barrio ha generado mayores cantidades de basura. El proceso de recolección de residuos sólidos urbanos (RSU) en el barrio, al igual que en la ciudad de San Gabriel, no cuenta con iniciativas actualizadas para ser mejorado. Deficiencias como no implementación de contenedores de basura, la variación significativa entre los horarios programados y los horarios reales de recolección, la falta de control de las mascotas (perros y gatos) que deambulan sin control por el sector que rompen las fundas de basura y la escasa concientización de los moradores en

relación con el tratamiento de RSU en los domicilios, derivan en problemas sociales/ambientales que agravan la situación del sector.

A nivel global, la generación excesiva de basura y la falta de infraestructura adecuada para su recolección y disposición final plantean desafíos significativos. Es relevante mencionar la importancia de los Objetivos de Desarrollo Sostenible (ODS) establecidos por la Organización de las Naciones Unidas (ONU), en particular el objetivo número 11, el cual se refiere a lograr que las ciudades y los asentamientos humanos sean inclusivos, seguros, resilientes y sostenibles, dentro de las metas de este objetivo se plantea la necesidad de reducir el impacto ambiental per cápita de las ciudades y mejorar la gestión de los desechos municipales, con el fin de garantizar la salud y el bienestar de los habitantes, así como la protección del medio ambiente (Organización de las Naciones Unidas - ONU, 2015).

En Plan de Desarrollo y Ordenamiento Territorial 2020-2035 del GADM Montufar destaca la importancia de mejorar el servicio integral de residuos sólidos en el sector urbano, lo que evidencia la necesidad de incorporar mejoras en el proceso de generación, almacenamiento, recolección, transporte, tratamiento y disposición final en el territorio cantonal (GADM Montúfar, 2020).

El presente proyecto se enmarca en la planificación territorial de los gobiernos locales y busca contribuir al logro de los Objetivos de Desarrollo Sostenible planteados por Naciones Unidas.

Asimismo, se sustenta legalmente en el artículo 14 de la Constitución de Ecuador que consagra el derecho de la población a vivir en un ambiente sano y ecológicamente equilibrado. Otros artículos relevantes son el 30, que establece el derecho a un hábitat y vivienda adecuados, y el 32 sobre el derecho a la salud vinculado a condiciones ambientales saludables (Asamblea Nacional Constituyente de Ecuador, 2008).

En cuanto a funciones sobre la gestión de residuos sólidos, el artículo 264 numeral 4 de la Constitución otorga esta competencia a los Gobiernos Autónomos Descentralizados Municipales (Asamblea Nacional Constituyente de Ecuador, 2008). Esto también se establece en el artículo 55 literal d del Código Orgánico de Organización Territorial, Autonomía y Descentralización (COOTAD) (Asamblea Nacional del Ecuador, 2010). Asimismo, el artículo 27 numerales 6 y 7 del Código Orgánico Ambiental asigna a los municipios formular planes, normas y procedimientos en esta materia (Asamblea Nacional del Ecuador, 2017a).

El "Sistema de Recolección de Residuos Sólidos Urbanos (RSU) basado en optimización de rutas y notificación en tiempo real para el barrio Centenario, ciudad de San Gabriel", se alinea con las metas de mejorar la calidad ambiental, reducir la contaminación y promover un entorno más saludable. Además, servirá como un modelo para la posible replicación en otros sectores que enfrentan desafíos similares en la recolección de RSU en el cantón Montufar.

# CAPÍTULO II: FUNDAMENTACIÓN TEÓRICA

En el presente capítulo expone los conceptos teóricos que fundamentan este trabajo de titulación, mediante el desarrollo de la investigación teórica sobre los residuos sólidos urbanos se analiza detalladamente el procesamiento de los residuos sólidos urbanos, junto con aspectos técnicos como hardware/software y tecnologías clave (4G y Sistema de Posicionamiento Global (GPS)) que permiten mejorar estos procesos. Esta fundamentación teórica permite establecer los principios teóricos y técnicos sobre los cuales se sustenta este proyecto.

## 2.1 Residuos Sólidos Urbanos (RSU)

De acuerdo con Gob. Argentina (2019), los residuos se refieren a cualquier tipo de material, objeto o sustancia que resulta descartado o dejado de lado como resultado de las actividades y procesos de consumo humanos. Los residuos sólidos urbanos, también conocidos por sus siglas RSU, corresponden a aquellos residuos vinculados a entornos domiciliarios, y pueden provenir de diversas fuentes como residencias, zonas urbanas, negocios, instalaciones de salud, sectores industriales o instituciones, excluyendo aquellos sujetos a regulaciones específicas.

Los residuos sólidos urbanos, comúnmente conocidos como "basura", se han convertido en una problemática para la población a medida que su producción se incrementa sustancialmente. El exceso de generación de residuos puede llegar a abrumar los espacios donde las personas habitan, transiten o usan para recreación. Cuando los volúmenes de basura superan la capacidad de gestión de una localidad, se empiezan a acumular y dispersar en el entorno, invadiendo áreas que deberían estar libres de desechos. Esta situación afecta negativamente la calidad de vida de los ciudadanos (Barradas, 2009) .

Para gestionar los residuos sólidos urbanos de manera adecuada, es esencial su clasificación a partir de su composición, ya que esto facilita la identificación de los distintos

tipos de RSU y posibilita la planificación de estrategias para un manejo sostenible. En el próximo apartado se profundizará en la composición de los RSU.

### *2.1.1 Composición*

La composición de los residuos sólidos urbanos es un factor cambiante y dinámico que puede experimentar variaciones notables en diferentes momentos y lugares. Este aspecto está influenciado por diversos factores, como la inclusión de nuevos materiales en los productos y transformaciones en los patrones de consumo de la población, entre otros aspectos relevantes. Conocer la composición de los residuos sólidos resulta fundamental para evaluar las necesidades de equipamiento, sistemas, así como para desarrollar programas y planes de gestión eficientes (UNEP, 2018).

La composición de los residuos sólidos propuesta por Ambientum (2022), varía dependiendo de diferentes factores y componentes:

- **Materia Orgánica:** Restos de comida, materiales de jardinería y otros elementos fermentables constituyen el componente principal de los residuos, aunque tiende a disminuir en las sociedades más avanzadas.
- **Vidrio:** Botellas, envases de alimentos, entre otros. La recolección selectiva de vidrio está en constante expansión en muchos lugares.
- **Papel y Cartón:** Incluye periódicos, papel en general, así como cajas y envases. Esta fracción ha experimentado notables aumentos en los últimos años, y la recolección en la fuente de origen está creciendo.
- **Plásticos:** Botellas y envases para líquidos, entre otros embalajes. Esta categoría agrupa diversos polímeros.
- **Otros componentes**: Incluye madera, cenizas, textiles, goma, latas metálicas, entre otros

Con el avance de la tecnología y la innovación, surgen constantemente nuevos materiales y productos en el mercado. Estos pueden contener componentes que no se descomponen fácilmente en el medio ambiente o que requieren métodos específicos de tratamiento para su disposición adecuada (Ambientum, 2022).

Los cambios en la composición de los RSU plantean desafíos para su gestión y tratamiento adecuado, ya que requieren sistemas de gestión que contemplen clasificación, reciclaje y disposición final que puedan adaptarse a estas variaciones. Uno de los procesos importantes es la clasificación en donde los residuos sólidos urbanos se segmentan en grupos definidos a partir de criterios que les permitan optimizar la gestión de los residuos, en el siguiente apartado se explica detalles específicos sobre la clasificación de los residuos sólidos urbano.

### *2.1.2 Clasificación*

De acuerdo con Revelo Morales (2019) , los desechos sólidos se categorizan de manera general basándose en su composición y de manera más específica según su procedencia. La clasificación ofrece beneficios como la posibilidad de reutilización, el reciclaje, la reducción en la generación inicial de desechos, la disminución de costos de procesos de gestión de los residuos, la creación de valor adicional a los residuos y el cumplimiento de las normativas vigentes (UNEP, 2018). En la Tabla 1 se muestra el ejemplo de dicha clasificación de residuos, especificando su origen según fuente generadora, el tipo de gestión requerida, el nivel de peligrosidad de los materiales y la composición en términos de residuos orgánicos e inorgánicos. Esta segmentación permite comprender mejor las múltiples cualidades de los RSU para implementar alternativas adecuadas de manejo. Así, estos atributos diferenciadores llevan a distintos enfoques para su tratamiento, desde el reciclaje total o parcial hasta la disposición final en caso de residuos no aprovechables (Revelo Morales, 2019).

**Tabla 1**

*Clasificación de Residuos*

| Caracterización | Tipo de residuo |
|---|---|
| Origen | Residencial (*domicilios*), Comercial *(restaurantes, tiendas),* Institucional *(hospitales, escuelas)*, Áreas públicas *(calles, parques, playas)* |
| Gestión | Municipal y no Municipal. |
| Peligrosidad | Peligrosos *(patogénicos, tóxicos, inflamables)* y No peligrosos *(reciclables y biodegradables)* |
| Naturaleza | Orgánicos ***(****Restos de alimentos, podas de jardín, etc.)* e Inorgánicos ***(****Plásticos, papeles, vidrios, metales, etc.)* |

*Nota.* Adaptado de (Revelo Morales, 2019)

La clasificación de los residuos sólidos urbanos, como se detalla en la Tabla 1, es esencial para comprender la diversidad de fuentes generadoras, su gestión, nivel de peligrosidad y composición. Esta segmentación no solo facilita la identificación de los distintos tipos de desechos, también permite establecer la naturaleza de estos y su manejo posterior. Esta información es importante para establecer una conexión con cada una las etapas de procesamiento de los residuos que involucra un sistema de gestión de residuos sólidos. Los sistemas de gestión de residuos deben considerar desde generación de los residuos sólidos urbanos, recolección, almacenamiento, transporte y destino final, en cada etapa se crean criterios específicos para asignar los recursos correspondientes, esto se analiza en el siguiente apartado.

### *2.1.3 Sistema de Gestión de Residuos Sólidos Urbanos*

Un sistema de gestión de residuos sólidos urbanos comprende el conjunto interrelacionado de procesos, métodos, infraestructura y recursos orientados a optimizar la administración de los residuos generados en áreas urbanas. Este sistema tiene como objetivo principal garantizar la limpieza, el orden y la salubridad de los entornos urbanos, así como gestionar adecuadamente los residuos producidos por la población desde su generación hasta su disposición final.

De acuerdo con Asamblea Nacional del Ecuador (2017b), se establecen 8 etapas para un sistema de gestión de residuos sólidos urbanos sea eficiente y cumpla con los estándares de manejo y tratamiento de los RSU. En paralelo, Revelo Morales (2019) en su estudio de la gestión de residuos adiciona la etapa de Generación que se considera importante para comprender el proceso total de manejo y tratamiento de residuos. A continuación, se detallan estas etapas que buscan alcanzar el objetivo de lograr un tratamiento más efectivo y sostenible de los residuos. Según el Asamblea Nacional del Ecuador (2017b)y Revelo Morales (2019), las 9 etapas de manejo y tratamiento de los residuos sólidos urbanos son:

1. **Generación:** Esta etapa depende completamente de las actividades humanas y está relacionada con la producción inicial de desechos.
2. **Almacenamiento:** Implica la retención temporal de los residuos sólidos por parte del generador hasta su entrega al servicio de recolección.
3. **Entrega:** En áreas con dificultades para la recolección, se debe trasladar los desechos a sitios designados por el ente encargado. Los generadores deben presentar los residuos según las condiciones establecidas en la normativa municipal y respetar horarios, rutas y días de recolección.
4. **Barrido y Limpieza de vías públicas:** Comprende acciones como barrer aceras y sumideros para mantener la limpieza y el orden en espacios urbanos, esenciales para la imagen de la ciudad.
5. **Recolección:** Es el proceso que vincula al generador con el encargado de la disposición final, procurando mantener la limpieza y evitar problemas como malos olores o desorden durante la recolección. Para optimizar el rendimiento, se deben establecer cronogramas, rutas y días específicos de recolección para cada sector, evitando así la acumulación de desechos en aceras, calles o terrenos, lo cual puede generar contaminación.

6. **Transporte o Transferencia:** Consiste en el traslado de los residuos desde vehículos recolectores a lugares de mayor capacidad, para luego ser llevados al relleno sanitario.
7. **Tratamiento:** Depende de las características de los residuos, pudiendo someterse a procesos técnicos, operativos, ambientales y económicos para su gestión.
8. **Disposición final:** Implica la ubicación permanente de los residuos no peligrosos en rellenos sanitarios, los cuales deben garantizar nulos o mínimos impactos en la salud, seguridad y ambiente durante su funcionamiento y cierre.
9. **Recuperación:** Involucra la reincorporación de espacios del relleno sanitario en proyectos posteriores, junto con el sellado y control de gases, lixiviados[1] y problemas ambientales relacionados.

Como se ha descrito, un sistema de gestión de residuos sólidos urbanos consta de varias etapas interrelacionadas, desde la generación inicial de los desechos hasta su disposición final y recuperación de los espacios utilizados. Cada una de estas fases requiere de procesos, métodos y recursos específicos para lograr una gestión eficiente y sostenible. A medida que las áreas urbanas crecen en tamaño y complejidad, se vuelve necesario implementar sistemas avanzados que optimicen la gestión de grandes volúmenes de residuos sólidos (Behdad et al., 2018). En la actualidad se han implementado sistemas modernos de gestión de RSU, con ayuda de tecnología se ha optimizado los diferentes procesos que involucra, en el siguiente apartado se detalla información sobre cómo son estos sistemas.

#### *2.1.3.1 Sistemas Modernos de Gestión de RSU*

Según Behdad et al. (2018), el constante crecimiento de las áreas urbanas y los volúmenes cada vez mayores de residuos sólidos generados, en las últimas décadas se han desarrollado tecnologías avanzadas para optimizar su gestión. La introducción de sistemas

[1] *Líquidos contaminados resultantes de la interacción del agua con residuos sólidos en un relleno sanitario, con potencial impacto ambiental.*

inteligentes y automatizados es indispensable para el manejo eficiente de los residuos en las nuevas ciudades inteligentes. Estos sistemas modernos presentan innovaciones significativas en cada eslabón de la cadena de gestión de residuos, desde la recolección hasta la disposición final. Algunos de los sistemas modernos que propone Behdad et al. (2018) en su artículo "The Future of Waste Management in Smart and Sustainable 2 Cities: A Review and Concept Paper" son:

- **Sistemas automatizados de recolección:** Utilizan camiones con sensores, GPS y software para optimizar las rutas de recolección en tiempo real. Permiten recolectar más residuos en menos tiempo.
- **Instalaciones de recuperación de recursos**: Plantas avanzadas que separan y procesan los residuos, extrayendo materiales reciclables y generando combustibles derivados de residuos. Reducen los desechos enviados a rellenos sanitarios.
- **Rellenos sanitarios de alta tecnología**: Cuentan con geomembranas, sistemas de drenaje de lixiviados, tuberías de recolección de gases y estaciones de tratamiento. Minimizan el impacto ambiental.
- **Sistemas neumáticos subterráneos:** Red de tuberías y válvulas que transportan los residuos mediante flujo de aire a una estación de transferencia. Mantienen limpias las calles y mejoran la estética urbana.
- **Incineración y gasificación de alta eficiencia:** Tecnologías que convierten los residuos en energía limpia y subproductos inertes, reduciendo drásticamente el volumen de los rellenos.
- **Compostaje industrial:** Planta que procesa los residuos orgánicos mediante fermentación aeróbica controlada para producir abono orgánico de alta calidad.

- **Gestión integral de residuos:** Enfoque que involucra a autoridades, empresas y ciudadanía para mejorar las regulaciones, procesos e infraestructura del sistema de residuos desde la generación hasta la disposición final.

Dentro de los sistemas modernos de gestión de RSU se tiene los sistemas de recolección, estos permiten optimizar las labores de recolección de RSU, se debe considerar que estos sistemas requiere una planificación cuidadosa que contemple la zonificación y rutas de recolección eficientes, el establecimiento de cronogramas, horarios y frecuencias adecuadas, la selección de vehículos y equipos apropiados, la capacitación del personal de recolección, campañas para promover la separación de residuos domiciliarios y sistemas de registro, monitoreo y mejora continua (UNEP, 2018). El siguiente apartado se detalla las características principales de estos sistemas de vital importancia para un óptimo sistema de gestión de RSU.

#### *2.1.3.2 Sistemas de Recolección de RSU*

Dentro de un sistema integral de gestión de residuos sólidos urbanos, la etapa de recolección cumple un rol fundamental, ya que vincula al generador de residuos con la disposición final de los mismos. Un eficiente sistema de recolección de RSU debe procurar mantener la limpieza de las ciudades y evitar problemas sanitarios o ambientales provocados por la acumulación inapropiada de basura(UNEP, 2018).

Existen dos modalidades principales de recolección de RSU tal como lo indica Ministerio del Ambiente (2022):

- **Recolección puerta a puerta:** el camión recolector transita por una ruta predefinida que incluye todos los domicilios en un sector determinado de la ciudad. Los residentes sacan sus bolsas de residuos en horarios y días específicos de recolección.
- **Recolección por contenedores**: se instalan recipientes o contenedores de gran volumen en lugares estratégicos del área urbana. Los ciudadanos depositan sus

residuos en dichos contenedores y con una frecuencia establecida los camiones recolectores vacían su contenido para el traslado y disposición final.

En los últimos años, la modernización e integración de tecnologías han contribuido a potenciar los resultados de los sistemas de recolección de RSU y por ende la gestión de los residuos en las ciudades. Algunas innovaciones aplicables son sistemas de monitoreo del llenado de contenedores, optimización dinámica de rutas de recolección, uso de vehículos ecoeficientes, entre otros (Behdad et al., 2018) .

La existencia de leyes que regulan los sistemas de gestión de residuos sólidos y sus diversas fases, destaca la importancia de examinar el marco legal que respalda la implementación de estos sistemas en entornos urbanos. Este análisis se vuelve esencial para comprender a fondo la normativa que rige las competencias de los sistemas de gestión de residuos sólidos, ya que no solo se trata de un aspecto ambiental, sino también de una responsabilidad legal y social. Además, identificar las obligaciones y requisitos legales establecidos contribuye a fortalecer la efectividad y la sostenibilidad de los procesos de manejo de residuos. En la siguiente sección se obtiene una visión más completa del marco legal que respalda a los sistemas de gestión de RSU.

#### *2.1.3.3 Marco Legal*

El marco legal de la gestión integral de residuos sólidos urbanos (RSU) en Ecuador se encuentra conformado por un conjunto de normas jurídicas que establecen las competencias, responsabilidades y obligaciones de los diferentes actores involucrados en el proceso de gestión de RSU.

Según Galindo et al. (2018), se establece un ordenamiento jurídico basado en la pirámide de Kelsen que ubica en la cúspide a la Constitución de la República y luego tratados y convenios internacionales, seguidos de las leyes, Reglamentos y Acuerdos Ministeriales, y en la parte más baja se encuentran las ordenanzas.

En Ecuador, los cuerpos legales que respalda a los sistemas de gestión integral de residuos sólidos son:

- **Constitución Nacional de la República del Ecuador**

De acuerdo con Asamblea Nacional Constituyente de Ecuador (2008), en su Título II, capítulo segundo, artículo 14, reconoce el derecho de la población a vivir en un ambiente sano y ecológicamente equilibrado, que garantice la sostenibilidad y el buen vivir de los ciudadanos. Este derecho se encuentra estrechamente relacionado con la gestión integral de RSU, ya que esta contribuye a la protección del ambiente y la salud pública. Por otro lado, en el Título VII, sección séptima, artículo 415, menciona que los GADs tienen como objetivo cumplir funciones de reducción, reciclaje y tratamiento de residuos sólidos.

En el artículo 264, se establece que los Gobiernos Municipales serán el ente competente exclusivo de prestar los servicios básicos a la población como servicio agua potable, manejo de desechos sólidos, actividades de saneamiento ambiental y aquellos que establezca la ley. Así mismo, se designa como ente competente a los GADs municipales los cuales tienen la obligación de la gestión de los residuos, mismos a través de sus disposiciones prestará los diferentes servicios a la población local. En cuanto a funciones sobre la gestión de residuos sólidos, el mismo artículo en su numeral 4 otorga esta competencia a los Gobiernos Autónomos Descentralizados Municipales (Asamblea Nacional Constituyente de Ecuador, 2008).

Otros artículos relevantes son el 30, que establece el derecho a un hábitat y vivienda adecuados, y el 32 sobre el derecho a la salud vinculado a condiciones ambientales saludables. Bajando de nivel en relación con la pirámide de Kelsen se analiza leyes, Reglamentos y Acuerdos Ministeriales.

- **Normas de rango internacional**

Ecuador es parte de varios convenios y tratados internacionales que promueven la gestión integral de RSU, entre los que se destacan:

- La Agenda 21, que establece como objetivo la reducción de la generación de residuos y el fomento de la reutilización y el reciclaje.
- El Convenio de Basilea, que regula el movimiento transfronterizo de residuos peligrosos.

- **Código Orgánico del Ambiente (CODA)**

En el Código Orgánico Ambiental (CODA) cuyo estado es vigente desde el 12 de abril del 2017 garantiza el derecho de las personas a vivir en un ambiente sano y ecológicamente equilibrado, así como proteger los derechos de la naturaleza. Las disposiciones de este Código regulan los derechos, deberes y garantías ambientales contenidos en la Constitución, así como los instrumentos que fortalecen su ejercicio, los que deberán asegurar la sostenibilidad, conservación, protección y restauración del ambiente, sin perjuicio de lo que establezcan otras leyes sobre la materia que garanticen los mismos fines (Asamblea Nacional del Ecuador, 2017a).

En su capítulo II, establece las facultades ambientales de los Gobiernos Autónomos Descentralizados y en su artículo 27 menciona que los Municipios deberán cumplir en el marco de sus competencias, en concordancia con las políticas nacionales emitidas por el Ministerio del Ambiente. Lo relacionado a la gestión integral de residuos y desechos sólidos urbanos se establecen desde el artículo 228 hasta el artículo 234 del mismo cuerpo legal, en los que se incluye el alcance y las fases de gestión, la responsabilidad de la creación de lineamientos y normas técnicas de manejo de residuos no peligrosos, así como las obligaciones y responsabilidades de los actores públicos y privados(Asamblea Nacional del Ecuador, 2017a).

El artículo 225 hace mención sobre las políticas generales de la gestión integral de los residuos y desechos, las cuales son de cumplimiento obligatorio para todas las personas naturales o jurídicas, independientemente de su ubicación o actividad económica. Por su parte el artículo 226, especifica la jerarquización que debe cumplir la gestión de residuos sólidos:

prevención, minimización de la generación en la fuente, aprovechamiento o valorización, eliminación y disposición final. Además, señala que la disposición final se limitará a aquellos desechos que no se puedan aprovechar, tratar, valorizar o eliminar en condiciones ambientalmente adecuadas y tecnológicamente factibles(Asamblea Nacional del Ecuador, 2017a).

- **Código Orgánico de Organización Territorial, Autonomía y Descentralización (COOTAD)**

El COOTAD vigente desde inicios del 2015 en su capítulo III, sección primera, artículo 55 establece las competencias exclusivas de los Gobiernos Autónomos Descentralizados Municipales para prestar los servicios públicos en los que se incluye el manejo de desechos sólidos, actividades de saneamiento ambiental y aquellos que establezca la ley. Por otro lado, el artículo 27 numerales 6 y 7 asigna a los municipios formular planes, normas y procedimientos en esta materia en concordancia con políticas nacionales (Asamblea Nacional del Ecuador, 2010).

- **Reglamento al Código Orgánico del Ambiente (RCODA)**

De acuerdo con Asamblea Nacional del Ecuador (2019), hace referencia al Reglamento al Código Orgánico del Ambiente (RCODA) vigente desde junio del 2019, en su título VII de la Gestión Integral de Residuos y Desechos, capítulo I, artículo 565 y 566 menciona la información mínima que debe contar un plan integral municipal de residuos y desechos sólidos no peligrosos y desechos sanitarios, así como la autorización para proyectos de la misma índole. En el mismo título, capítulo III de la Gestión integral de residuos y desechos sólidos no peligrosos, sección primera, artículo 578, se dan a conocer las disposiciones generales del Plan Nacional de Gestión Integral de Residuos y Desechos Sólidos No Peligrosos, comprometiendo a los Gobiernos Autónomos Descentralizados Municipales y Metropolitanos, las entidades competentes, sector privado, sociedad civil y academia para la generación de

políticas, estrategias, planes, programas y proyectos para la gestión integral de residuos y desechos sólidos no peligrosos. En la Sección Tercera, artículo 586, se mencionan las fases de la gestión integral de residuos y desechos sólidos no peligrosos, mientras que en su artículo 584 se presentan las obligaciones y normativa aplicable a todo generador de residuos en cuanto al manejo y la minimización de la generación en la fuente de los residuos (Asamblea Nacional del Ecuador, 2019).

- **Libro VI del Texto Unificado de Legislación Ambiental TULSMA (Acuerdo Ministerial 061)**

El Acuerdo Ministerial No. 061, Reforma del Libro VI del Texto Unificado de Legislación Ambiental (TULSMA), conforme el Art 47. el cual se refiere a las Políticas Nacionales de Residuos Sólidos señala que, el Estado Ecuatoriano declara prioridad la gestión integral de residuos sólidos en el país, como una responsabilidad compartida por toda la sociedad, que contribuya al desarrollo sustentable a través de un conjunto de políticas intersectoriales nacionales. La responsabilidad compartida está enfocada en el involucramiento de la sociedad dentro del proceso, la participación de la ciudadanía es clave, ya que las prácticas como el reciclaje y separación de residuos empieza en el origen de generación de estos, y esto contribuye en la minimización de residuos, así como evitar impactos al ambiente (Asamblea Nacional del Ecuador, 2017b).

De acuerdo con Asamblea Nacional del Ecuador (2017b), en el acuerdo con el Texto Unificado de Legislación Ambiental (TULSMA), en su artículo 55 establece que, la gestión integral de los residuos sólidos no peligrosos, como el conjunto de acciones y regulaciones con el objetivo de dar a los residuos sólidos no peligrosos el destino más apropiado desde el punto de vista técnico, económico y medioambiental. En este apartado la normativa establece la posibilidad de mejorar la gestión a través de diferentes prácticas y acciones que promuevan la separación y reciclaje de los materiales. Aquellas operaciones minimizarán el envío de gran

cantidad de residuos a los rellenos evitando repercusiones al ambiente y la salud de la población. Además, estas acciones son provechosas para varias personas al beneficiarse del valor de comercialización de los materiales recuperados con el valor agregado de cada uno de los materiales.

- **Ordenanza para la Gestión Integral de Residuos Sólidos del GADM Montúfar.**

En lo más bajo de la pirámide de Kelsen se encuentran las ordenanzas, que son normas jurídicas dictadas por los gobiernos autónomos descentralizados en ejercicio de sus competencias(Galindo et al., 2018). Las ordenanzas municipales pueden regular una amplia gama de temas, incluidos los relacionados con la gestión integral de residuos sólidos.

De acuerdo con GADM Montufar (2021a), que hace referencia a la "Ordenanza para la Gestión Integral de Residuos Sólidos y Desechos Líquidos (Aceites de Cocina Usado) en el cantón Montúfar", señala en el Capítulo I, artículos 4 y 5, las obligaciones del generador de los residuos para separar en la fuente los residuos sólidos no peligrosos y garantizar el desalojo y eliminación de residuos industriales y escombros, de igual forma en el Capítulo II, artículo 10 se detallan otras obligaciones del generador establecidas por el GADM Montufar. En esta ordenanza, específicamente Capítulo VI, se enfoca al proceso de Recolección, en el artículo 14, se establecen consideraciones para antes y durante del proceso de recolección como: clasificación, formas de entrega o depósito de los residuos y obligaciones a cumplir por parte de los operarios del servicio como también de los usuarios. De igual forma existen otros capítulos que regulan la gestión integral de residuos sólidos y especifican fases, procedimientos y contravenciones de diferentes clases para pagos de multas según sea el caso.

En el caso del GADM Montufar, cuenta con una ordenanza que establece el cobro de una tarifa por el servicio de la gestión integral de residuos sólidos en el cantón. Esta ordenanza en el Artículo 18 establece que, la tarifa que se aplicará será del 37% en la zona urbana de la

ciudad de San Gabriel, el 25% en las comunidades para el sector residencial, mientras que para el sector comercial se aplicará el 57% en la zona urbana de la ciudad de San Gabriel y el 28% en las comunidades, calculado sobre el consumo mensual de energía eléctrica de los usuarios de este servicio. Esta acción se realiza a través de la empresa EMELNORTE S.A (GADM Montufar, 2021b)

El "Sistema de Recolección de Residuos Sólidos Urbanos (RSU) basado en optimización de rutas y notificación en tiempo real para el barrio Centenario, ciudad de San Gabriel", se alinea con el marco legal revisado, reduciendo la contaminación y promoviendo un entorno más saludable.

En la actualidad, a lo largo de las diferentes etapas de procesamiento de residuos sólidos, se han incorporado una amplia gama de tecnologías para mejorar y optimizar estos procesos. Estas tecnologías han sido fundamentales para la optimización de la gestión de los residuos, desde su recolección hasta disposición final y recuperación. Gracias a estas innovaciones tecnológicas, se ha logrado aumentar la eficiencia en el proceso de recolección de RSU, minimizando su impacto ambiental negativo e incentivando su posible reutilización o reciclaje.

Como se mencionó en la lista de los sistemas modernos de gestión de RSU, una de las tecnologías son los sistemas automatizados de recolección los cuales usan GPS para optimizar las rutas de recolección, el siguiente apartado nos permite comprender como operan estas tecnologías.

## 2.2 Sistema Global de Navegación por Satélite (GNSS)

Un Sistema Global de Navegación Satelital, también conocido por la sigla GNSS, se define como una red pasiva de radionavegación. Su funcionamiento se basa en una constelación de satélites que emiten señales de radio y que proporcionan un sistema de referencia espacio temporal a una cantidad ilimitada de usuarios alrededor del mundo. Este marco de

geoposicionamiento global está disponible continuamente en cualquier punto del planeta, sin importar las condiciones climatológicas. Los satélites GNSS conforman una infraestructura orbital que permite la navegación y localización precisa desde cualquier lugar de la superficie terrestre (Leal, 2013).

En la actualidad existen algunas herramientas de sistemas satelital de navegación, estas herramientas nos ayudan a determinar nuestra ubicación precisa en cualquier parte del mundo, brindándonos información sobre rutas, direcciones y puntos de interés. Además, facilitan el seguimiento y monitoreo en tiempo real de vehículos, personas y activos, lo que permite una gestión eficiente de la logística, mejora la seguridad en el transporte y optimiza la planificación en diversas industrias (Pozo-Ruz et al., n.d.).

En la Tabla 2, se visualizan las diferencias entre los tres sistemas satelital de navegación: GPS, BeiDou y Galileo. Cada uno de estos sistemas consta de una red de satélites en órbita que emiten señales para determinar la ubicación exacta de receptores en la Tierra. Además, se detallan aspectos importantes como el número de satélites, la cobertura global, la precisión, la resistencia a interferencias y el ente que desarrolla/controla cada sistema(Santerre et al., 2014). Estas diferencias son fundamentales al evaluar las aplicaciones y usos específicos.

**Tabla 2**

*Diferencias entre los sistemas globales de navegación satelital (GNSS)*

| Aspecto | GPS | BeiDou | Galileo |
|---|---|---|---|
| Número de satélites | Más de 30 satélites en órbita | Más de 35 satélites en órbita | Más de 24 satélites en órbita |
| Cobertura global | Sí | Sí | Sí |
| Precisión | ~5 metros (civil), <1 metro (militar) | ~10 metros (civil), <5 metros (militar) | ~1 metro (civil), <1 metro (militar) |
| Resistencia a interferencias | Vulnerable a interferencias | Mejora en resistencia a interferencias | Mejora en resistencia a interferencias |
| Desarrollo y control | EE. UU. | China | Unión Europea |

*Nota.* Adaptado de (Santerre et al., 2014).

La tecnología GPS, desarrollada por Estados Unidos, se destaca como la más ampliamente utilizada en el mundo debido a su precisión y disponibilidad. En el siguiente apartado, se desarrollan los fundamentos teóricos y técnicos del GPS para comprender mejor su estructura, principios de funcionamiento y los elementos clave que hacen posible su capacidad de proporcionar información precisa de ubicación y tiempo en cualquier lugar del planeta. Esto permite tener una visión más completa de cómo este sistema se ha convertido en una herramienta fundamental en una amplia gama de aplicaciones en la actualidad.

### *2.2.1 Sistema de Posicionamiento Global (GPS)*

El Sistema de Posicionamiento Global (GPS) es un sistema de radionavegación de los Estados Unidos de América, basado en el espacio, que proporciona servicios fiables de posicionamiento, navegación, y cronometría gratuita e ininterrumpidamente a usuarios civiles en todo el mundo. A todo el que cuente con un receptor del GPS, el sistema le proporcionará su localización y la hora exacta en cualquier condición atmosférica, de día o de noche, en cualquier lugar del mundo y sin límite al número de usuarios simultáneos (Oficina de Coordinación Nacional de Posicionamiento Navegación y Cronometría por Satélite, n.d.).

El GPS ha brindado servicios continuos de posicionamiento, navegación y sincronización horaria a usuarios civiles y militares a nivel mundial desde que se declaró su capacidad operativa inicial en 1993. En la actualidad, el GPS se ha convertido en una utilidad global cuyos servicios de usos múltiples son parte integral de la seguridad y crecimiento económico de EUA y el mundo, así como de la seguridad del transporte y un elemento esencial de la infraestructura económica internacional (Departamento de Defensa de Estados Unidos, 2007).

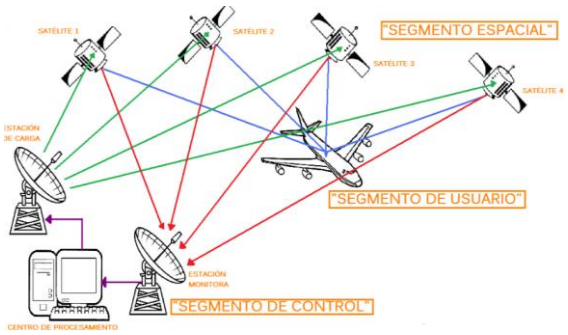
#### *2.2.1.1 Arquitectura del Sistema GPS*

La arquitectura GPS se refiere al diseño técnico y estructural del Sistema de Posicionamiento Global; en la Figura 1, se puede visualizar la arquitectura del sistema GPS, la

cual se descompone en tres segmentos básicos, dos son de responsabilidad militar y una del usuario.

**Figura 1**

*Arquitectura del Sistema GPS*



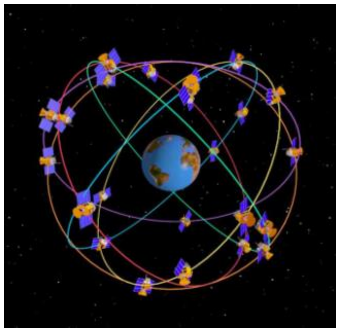

*Nota.* Tomado de (Leal, 2013)

De acuerdo con Leal (2013), el sistema GPS está conformado por tres segmentos principales:

- **Segmento espacial:** está conformado por la constelación de satélites en órbita (ver Figura 2), que actualmente consta de 24 satélites operativos y 3 de reserva. Los satélites se distribuyen en seis planos orbitales inclinados 55 grados con respecto al plano ecuatorial terrestre, a una altitud de 20200 km. Esta configuración asegura que al menos 6 satélites sean visibles desde cualquier punto del planeta en todo momento

**Figura 2**

*Constelación de Satélites en Órbita*



*Nota.* Los satélites en Órbita tienen un periodo de 12h, (Leal, 2013).

**Segmento de control:** lo integran cinco estaciones monitoras que mantienen la constelación en función, tres antenas en tierra que envían las señales a los satélites y una estación central que supervisa todo. En la Figura 3, se visualiza la localización de las estaciones de control del sistema GPS.

**Figura 3**

*Ubicación de las estaciones de control del sistema GPS*


*Nota.* Tomado de (Leal, 2013)

- **Segmento de usuarios:** se ubican las antenas receptoras y decodificadoras de la señal GPS que se encuentran en la superficie terrestre.

#### *2.2.1.2 Principio de funcionamiento del sistema GPS*

Esta tecnología se basa en el principio de triangulación. Los receptores GPS miden el tiempo que tarda en llegar la señal de radio de cada satélite. Utilizando estas mediciones, el receptor puede calcular su distancia a cada satélite. Conociendo sus distancias a los satélites, el receptor puede calcular su posición en el espacio. Los receptores GPS utilizan estas señales para calcular la posición, velocidad y dirección del receptor (Santerre et al., 2014).

El sistema de posicionamiento global GPS brinda coordenadas de latitud, longitud y altitud referidas a un elipsoide que constituye un modelo matemático de la forma de la Tierra. Fue desarrollado originalmente para uso militar por el Departamento de Defensa de Estados

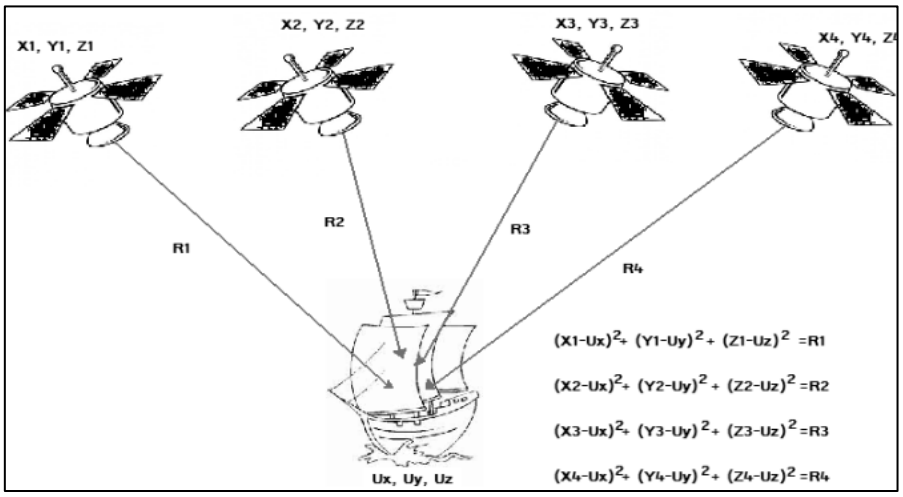
Unidos, pero actualmente su aplicación se ha ampliado al ámbito civil (Departamento de Defensa de Estados Unidos, 2007).

La ubicación de cada satélite perteneciente a la constelación de satélites del sistema GPS se conoce en todo momento gracias a sus efemérides[2]. Los satélites emiten ondas portadoras en dos frecuencias denominadas L1 (1575.42 MHz) y L2 (1227.6 MHz), que están moduladas con un código binario que permite determinar tiempos de viaje (Leal, 2013).

En la Figura 4, se muestra la representación esquemática para calcular una posición a partir de las distancias, el receptor mide distancias a satélites mediante el retardo entre la señal generada internamente y la recibida desde el espacio. En este segmento, las coordenadas Earth-Centered, Earth-Fixed (ECEF) que se traducen en español como Tierra-Centradas y Fijas en la Tierra: X, Y, Z se conectarán con las ubicaciones de los satélites indicadas por los parámetros orbitales después de procesar la información de tiempo y posición de cada señal proveniente de un satélite específico (Leal, 2013).

**Figura 4**

*Representación esquemática del cálculo de la distancia en el GPS*



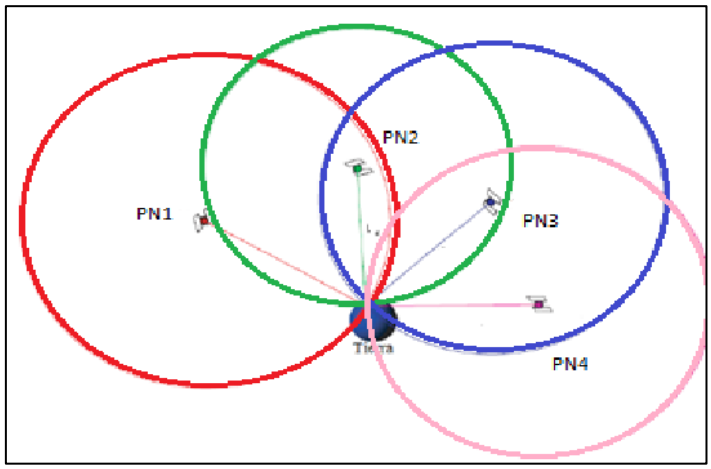

*Nota.* Tomado de (Leal, 2013)

[2] *Datos orbitales precisos que permiten a los receptores GPS rastrear y ubicar la posición de los satélites en órbita en todo momento.*

Se requiere sincronizar relojes comparando señales de 4 o más satélites para corregir errores, tal y como se muestra en Figura 5. Luego se calcula la intersección de esferas cuyos centros son las coordenadas de cada satélite y los radios las distancias medidas.

**Figura 5**

*Principio de Posicionamiento GPS*




*Nota.* Tomado de (Leal, 2013)

De acuerdo con Mancera (2004), el sistema GPS utiliza dos tipos principales de códigos para modular sus señales:

- **Código de Adquisición Gruesa (C/A)**: tiene una tasa de repetición de 1023 MHz y período de 1 ms. Está asignado al Servicio Estándar (SPS) para usuarios civiles. Este código sufría una degradación intencional en su precisión por temas de seguridad nacional de EE.UU.
- **Código de Precisión (P):** tiene una frecuencia de repetición de 10 MHz destinado al Servicio de Posicionamiento Preciso (PPS) de uso militar. Su acceso estaba encriptado y controlado.

Cada satélite transmite en dos frecuencias portadoras:

- **L1** = 1575.42 MHz, modulada con códigos C/A y P
- **L2** = 1227.6 MHz, modulada solo con código P.

Ambas frecuencias se modulan de forma continua con un mensaje de navegación de 50 bits/s que contiene datos sobre estado de satélites, correcciones de reloj, efemérides, ionosfera, entre otros (Departamento de Defensa de Estados Unidos, 2007).

Es así como la tecnología GPS se han popularizado en múltiples aplicaciones que aprovechan su capacidad de geolocalización precisa en tiempo real para actividades de transporte, respuesta a emergencias, coordinación de infraestructura crítica, agricultura, topografía y otras más. A medida que evolucionaba el GPS, se crearon tecnologías para enviar y recibir los datos de ubicación, y una de las más prominentes es la tecnología móvil. En la actualidad, la tecnología de cuarta generación (4G) ha revolucionado no solo nuestra comunicación a través de dispositivos móviles, sino también la transmisión y recepción de datos de GPS de manera más eficiente y veloz. En el próximo apartado, se profundiza los principios teóricos y técnicos de la tecnología móvil, enfocándonos en la tecnología 4G, la cual es fundamental en el desarrollo de este trabajo de titulación.

## 2.3 Tecnología Móvil

La tecnología móvil, caracterizada por la transferencia inalámbrica de información entre dispositivos, como teléfonos móviles, mediante el uso de ondas electromagnéticas, constituye una red de comunicaciones integrada por antenas distribuidas y terminales móviles. Esta evolución tecnológica, desde 2G hasta 5G, ha revolucionado la comunicación al proporcionar no solo llamadas y mensajes de texto, sino también altas velocidades de datos, menor latencia y una capacidad mejorada para aplicaciones avanzadas como la Internet de las cosas (IBM, n.d.).

En la Tabla 3, se recopila los datos característicos de las diferentes tecnologías móviles, proporcionando especificaciones técnicas clave de las tecnologías 2G, 3G, 4G y 5G, lo que facilita la comparación entre cada generación.

**Tabla 3**

*Características de tecnologías móviles*

| Tecnología Móvil | Generación | Año | Velocidad de Tx | Latencia | AB[3] | Modo de Acceso |
|---|---|---|---|---|---|---|
| 2G (GSM[4]) | 2G | Década 1990 | Hasta 384 Kbps | Decenas de milisegundos | 200 kHz | TDMA[5] |
| 3G (UMTS[6]) | 3G | Década 2000 | Hasta 2 Mbps | Decenas de milisegundos | 5 MHz | CDMA[7] |
| 4G (LTE) | 4G | Década 2010 | Hasta 1 Gbps | Milisegundos | 20 MHz | OFDMA[8] |
| 5G | 5G | 2019 | Hasta 20 Gbps | Milisegundos | Hasta 100 MHz | OFDMA |

*Nota.* Adaptado de (Iqbal et al., 2021).

En el contexto ecuatoriano, la tecnología móvil ha experimentado una evolución notable, encontrando a la tecnología 4G LTE como la más avanzada y accesible en la actualidad. Este progreso tecnológico sirve como base fundamental en el desarrollo de ideas innovadoras para la gestión de RSU. En este trabajo de titulación, se hace un enfoque especial en el uso del 4G LTE que radica en su capacidad para transmitir y recibir datos precisos de GPS, permitiendo así el diseño y la implementación de rutas eficientes para la recolección de residuos. La velocidad y fiabilidad inherentes al 4G facilitan una transferencia fluida de información geoespacial, factor esencial para optimizar el proceso de gestión de RSU en el entorno urbano (IBM, n.d.). En la sección que sigue, se profundiza los conceptos teóricos fundamentales acerca del 4G LTE. Esto incluye información sobre la tecnología, su arquitectura y su modo de funcionamiento, proporcionando así un entendimiento detallado de esta tecnología.

[3] Ancho de Banda
[4] Global System for Mobile Communications
[5] Time Division Multiple Access
[6] Sistema Universal de Telecomunicaciones Móviles
[7] Code Division Multiple Acces
[8] Orthogonal Frecuency Division Multiple Access

### *2.3.1 4G LTE*

El sistema móvil 4G LTE (Long-Term Evolution) se encuentra completamente fundamentado en el protocolo de Internet (IP, por sus siglas en inglés). Su propósito primordial es asegurar altas velocidades, calidad óptima, gran capacidad, seguridad robusta y servicios de bajo costo para la transmisión de servicios de voz y datos, así como multimedia e internet a través del protocolo IP. En el uso de la red de comunicación móvil 4G, los dispositivos móviles de los usuarios deben ser capaces de elegir y conectarse al sistema inalámbrico designado(Mompó, 2019).

Según Mompó (2019), las características principales que definen las ventajas y diferencias notables del 4G LTE respecto a sus predecesoras se encuentran en aspectos específicos del sistema:

- **Flexibilidad espectral:** El diseño permite operar en diferentes asignaciones de espectro, desde 1,4 MHz hasta 20 MHz tanto en sentido ascendente UpLink (UL) como descendente DownLink (DL). Esta versatilidad posibilita implementaciones del LTE en distintos espectros y con diversas configuraciones, como Dúplex de División de Frecuencia (FDD, por sus siglas en inglés) y Dúplex de División de Tiempo (TDD, por sus siglas en inglés).
- **OFDM en DL y SC-FDMA en UL:** Se emplea OFDM en el DL, que requiere un procesamiento más complejo en la Estación Base, mientras que en el UL se utiliza SC-FDMA, optimizando el transmisor y reduciendo el consumo de potencia.
- **Control de tasa binaria a potencia constante:** Se logra mantener una tasa de datos constante independientemente de la potencia transmitida.
- **Múltiples antenas para transmisión y recepción:** La inclusión de MIMO (Multiple Input, Multiple Output) y SIMO (Single Input, Multiple Output), con

más antenas en transmisión y recepción, incrementa la velocidad de transmisión y recepción, mejorando la eficiencia espectral y la velocidad de transferencia.

- **Simplificación de arquitectura de red:** Mayor inteligencia en las Estaciones Base permite una distribución eficiente de recursos, lo que reduce el número de nodos en la red y, por ende, la latencia de esta.
- **Todo IP, conmutación de paquetes:** La red está basada en protocolo IP, siguiendo una arquitectura completamente orientada a paquetes.
- **Transmisión de contenidos multicast/broadcast:** Se habilita la transmisión de datos a múltiples usuarios simultáneamente, utilizando conexiones punto a multipunto, sin limitaciones en la cantidad de usuarios dentro del área de cobertura que puedan acceder al servicio al mismo tiempo.

Las características esenciales del 4G LTE representan un salto significativo en la evolución de las redes móviles, ofreciendo una serie de ventajas técnicas que han revolucionado la comunicación inalámbrica. Con las características detalladas anteriormente, las redes móviles han experimentado una significativa mejora en la eficiencia de gestión y distribución de información. Para comprender cómo estos atributos configuran y optimizan el rendimiento del 4G LTE, es esencial hacer el análisis de su arquitectura, la cual esta especificada detalladamente en el siguiente apartado.

#### *2.3.1.1 Arquitectura*

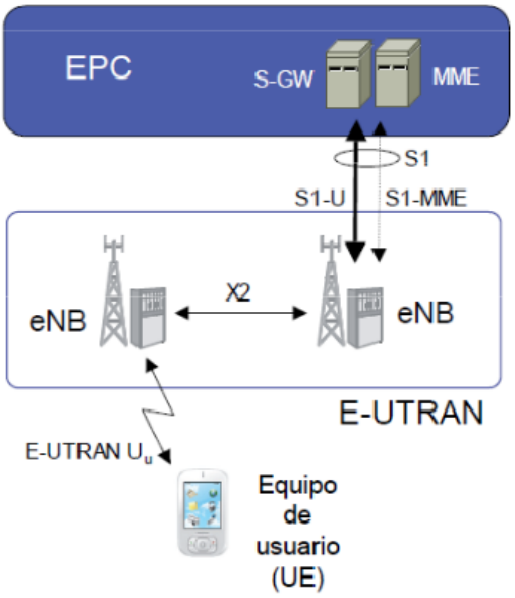
La arquitectura de la red 4G LTE representa una evolución significativa con respecto a la arquitectura del GPRS[9]. En el caso del 4G, se adopta el Evolved Packet System (EPS), una estructura que marca un cambio importante en la forma en que se gestiona y opera la conectividad inalámbrica. La Figura 6 ilustra detalladamente los componentes que integran el

[9] General Packet Radio Service (Servicio General de Paquetes vía Radio). Estándar de comunicación para teléfonos móviles.

EPS y que son fundamentales para el funcionamiento y la eficacia de esta tecnología. Estos componentes se combinan para proporcionar una infraestructura altamente eficiente y optimizada, lo que permite al 4G LTE alcanzar su potencial máximo en términos de velocidades de datos, menor latencia y una conectividad más estable y confiable para los usuarios finales(Mompó, 2019).

**Figura 6**

*Arquitectura Evolved Packed System*




*Nota.* Tomado de (Mompó, 2019)

De acuerdo con Mompó (2019), la EPS se compone de dos partes principales:

- **Parte radio**: también conocida como E-UTRAN, proporciona la interfaz entre los terminales de usuario y la red. Está formada por los siguientes elementos:
  - **User Equipment (UE):** es el terminal de usuario.
  - **Evolved Node B (eNodeB):** es la estación base que proporciona la interfaz radio.

- **Parte central:** también conocida como Core Network, proporciona los servicios de control y transporte de la red. Está formada por los siguientes elementos:
    - **Mobility Management Entity (MME):** es el nodo principal de control de la red.
    - **Serving Gateway (S-GW):** es el nodo que se encarga del transporte de los datos de usuario entre la eNodeB y la red de transporte.

La comunicación entre la Estación Base (eNodeB) y la Red Central se efectúa mediante dos protocolos distintos: el S1-U se utiliza para la transmisión de datos de usuario, mientras que el S1-MME se encarga de la información de control, que abarca aspectos como el cambio de conexión (handovers), la paginación, entre otros. Además, existe una conexión directa entre diferentes Estaciones Base (eNodeB) a través de la interfaz X2 (Mompó, 2019).

La Figura 7 muestra un análisis más detallado de la arquitectura de 4G LTE, que de acuerdo con Mompó (2019), verifica como se lleva a cabo la conexión detalladamente entre cada uno de los componentes de la arquitectura Evolved Packed System.

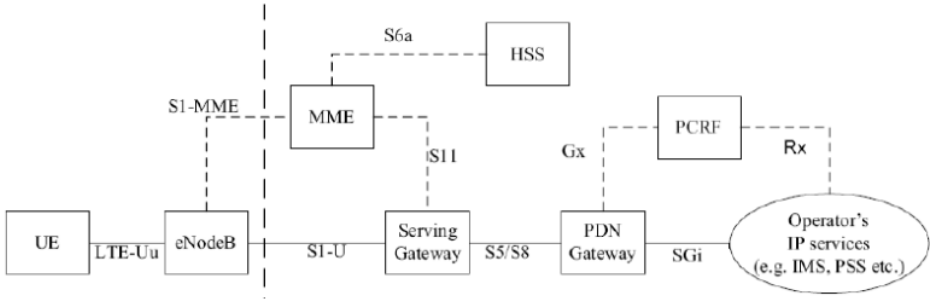
En la red troncal, la conexión entre el PDN Gateway y el SGW (Serving Gateway) se establece mediante la interfaz S5 si ambos pertenecen al mismo operador, o a través de la interfaz S8 en el caso de que sean de operadores diferentes. Esta conexión emplea el protocolo GTP-U (Protocolo de Tunelización de Paquetes GPRS-Usuario) para transportar datos del PDN Gateway al SGW. Asimismo, se utiliza un túnel GTP-C para el transporte de información de control(Mompó, 2019).

En el núcleo del EPC (Evolved Packet Core), existe una entidad denominada Home Subscriber Service (HSS) que mantiene una conexión con el MME a través de la interfaz S6a para gestionar y autorizar el acceso a la red LTE, entre otras funciones esenciales. La interfaz S6a emplea el protocolo diameter, basado en conexiones (similar a TCP), para enlazar el MME

con el HSS, por último, tenemos el Policy and Charging Rules Function (PCRF), el cual se encarga de la gestión de la política de tarificación de la red.

**Figura 7**

*Arquitectura Evolved Packed System I*




*Nota.* Tomado de (Mompó, 2019)

Después de comprender la arquitectura fundamental del 4G LTE, es importante tener en cuenta aspectos específicos que influyen en su funcionamiento óptimo, como las bandas de operación, las cuales son un componente vital en el despliegue y rendimiento de la tecnología, la diversidad de frecuencias puede afectar directamente la cobertura, velocidad y eficacia de las redes 4G LTE. En la sección siguiente se exponen las distintas bandas de operación asociadas a esta tecnología.

#### *2.3.1.2 Bandas de Frecuencia de Operación*

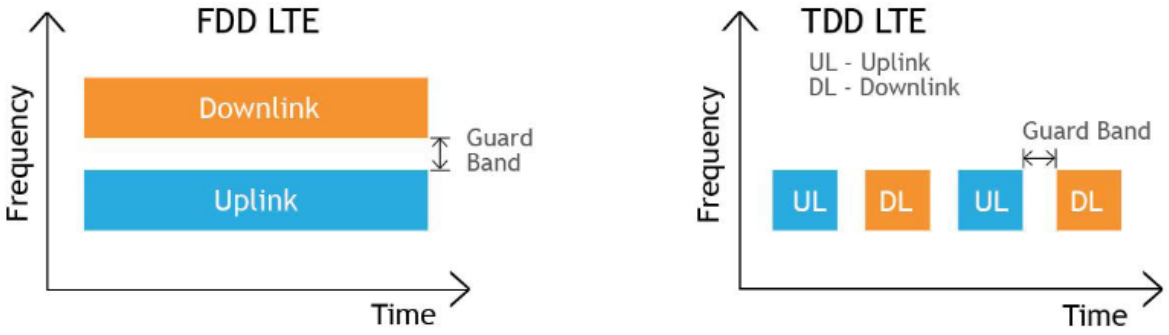
LTE tiene la capacidad de emplear dos métodos para la transmisión de datos: Dúplex de División de Frecuencia (FDD) y Dúplex de División de Tiempo (TDD). En la mayoría de las implementaciones habituales de LTE, se opta por utilizar el método FDD (Mompó, 2019).

En la Figura 8 se verifica de modo gráfico las diferencias entre FDD y TDD. Por su parte FDD opera con dos canales de frecuencia simétricos separados, utilizando una banda de guarda para distinguir entre los canales de transmisión y recepción. Sin embargo, su desventaja radica en la necesidad de pares de frecuencias para diferenciar entre los enlaces ascendente y descendente, lo que limita su utilización del espectro cuando se admiten servicios asimétricos,

mientras que TDD separa los canales de transmisión y recepción mediante el uso del tiempo en un único canal. En este método, la estación base y la estación móvil coordinan la transmisión y recepción en intervalos de tiempo específicos en una misma portadora de frecuencia, permitiendo así la comunicación bidireccional con una asignación eficiente de recursos temporales(Mompó, 2019).

**Figura 8**

*Comparación gráfica entre FDD y TDD*




*Nota.* Tomado de (Mompó, 2019)

En la Tabla 4, se proporciona un resumen de las bandas de frecuencia utilizadas en las redes de comunicaciones 4G LTE (Long Term Evolution) y sus características principales, la información que contiene permite entender la distribución de las frecuencias LTE y su utilización en diversas regiones geográficas y tecnologías de duplexación.

**Tabla 4**

*Bandas de Frecuencia de Operación de 4G LTE*

| Banda | Frecuencia (MHz) | Duplexación | Región | DL (MHz) | UL (MHz) |
|---|---|---|---|---|---|
| 700 | 700-780 | FDD | Europa, América, Asia | 700-728 | 752-780 |
| 800 | 800-850 | FDD | Europa, Asia | 800-828 | 832-850 |
| 1800 | 1800-1880 | FDD | Europa, América, Asia | 1800-1827 | 1852-1880 |
| 2100 | 2100-2170 | FDD | Europa, América, Asia | 2100-2127 | 2152-2170 |
| 2600 | 2500-2690 | FDD | Europa, América, Asia | 2500-2527 | 2552-2690 |
| 1800 | 1710-1885 | TDD | Europa, Asia | 1710-1745 | 1845-1885 |

| | | | | | |
|---|---|---|---|---|---|
| 2600 | 2570-2620 | TDD | Europa, América, Asia | 2570-2595 | 2615-2620 |
| 3500 | 3400-3600 | TDD | Europa, América, Asia | 3400-3425 | 3575-3600 |

*Nota.* Adaptado de (3GPP, 2021)

Las bandas de frecuencia y tecnologías de duplexación descritas previamente son fundamentales para la operación de las redes 4G LTE, las cuales brindan servicios de comunicaciones móviles de alta velocidad. Estos avances en las tecnologías de telecomunicaciones, junto con el desarrollo de sistemas de posicionamiento global (GPS) y los conceptos relacionados a la gestión de residuos sólidos urbanos, abren nuevas oportunidades para optimizar procesos como la recolección de residuos mediante técnicas de geolocalización y el uso de dispositivos móviles. La integración de tecnologías como 4G LTE y GPS permite el monitoreo en tiempo real, la trazabilidad y la optimización de rutas de los vehículos recolectores. En este contexto, el hardware desempeña un papel crucial al brindar la plataforma física para la implementación de soluciones innovadoras en el ámbito de la recolección de residuos sólidos urbanos. En el siguiente apartado, se analizan los componentes de hardware y dispositivos electrónicos que conforman la base de esta solución tecnológica.

## 2.4 Componentes de Hardware del sistema de recolección RSU

Según Sasaki (1993), los componentes de Hardware hacen referencia a todas las partes tangibles de un sistema como sensores, placas de desarrollo, cables de conexión, entre otros.

Para que el sistema funcione de forma óptima, todos los componentes de hardware deben estar en perfectas condiciones. Esto se debe a que los componentes de hardware interactúan entre sí para ejecutar las funciones del sistema. Si un componente de hardware está dañado o defectuoso, puede afectar el rendimiento de todo el sistema(Sasaki, 1993).

Una plataforma que aprovecha múltiples componentes de hardware para facilitar el prototipado electrónico es Arduino, la cual es explorada con más detalle en la siguiente sección.

### *2.4.1 Arduino*

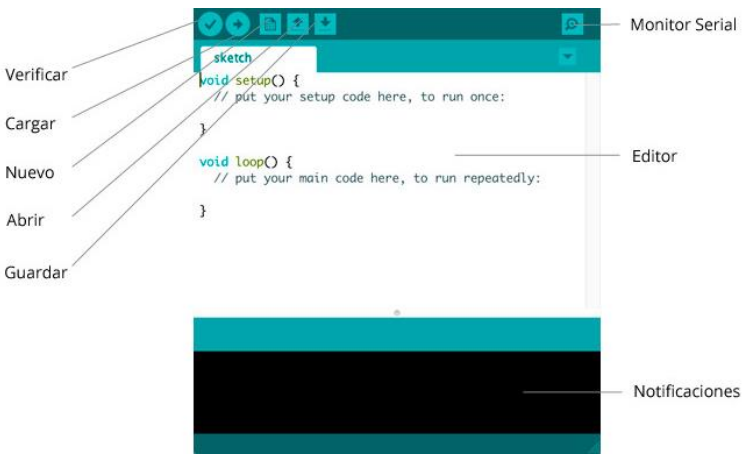
Arduino es una plataforma de hardware y software de código abierto que ayuda a los aficionados, estudiantes y profesionales a crear proyectos electrónicos. Esta es una excelente opción para aprender electrónica, programación y robótica y desarrollar proyectos prácticos en una variedad de campos debido a su diseño modular y versatilidad (Arduino, 2018)

Los microcontroladores en las placas Arduino se pueden programar para leer entradas (inputs) desde una variedad de sensores y pueden controlar luces, motores y otros componentes físicos por medio de sus pines de E/S digitales y analógicas. Los proyectos creados con Arduino pueden ser autónomos o pueden comunicarse con software ejecutándose en un ordenador. El software Arduino es de código abierto, fácil de usar y corre en Windows, macOS y Linux (Arduino, 2018).

En la Figura 9 se muestra la interfaz del Arduino IDE con sus herramientas básicas, este entorno de desarrollo integrado es utilizado para programar placas Arduino y está disponible de forma gratuita en los repositorios en línea de Arduino, este software es fundamental para desarrollar proyectos de hardware y software. Ofrece herramientas esenciales para la programación y la visualización de resultados, lo que facilita tanto la escritura de código como la interacción con los dispositivos físicos (Arduino, 2018)

**Figura 9**

*Interfaz de Arduino IDE*



El Arduino Integrated Development Environment (IDE) es la plataforma de código abierto diseñada para simplificar el desarrollo y la programación de proyectos utilizando placas Arduino. Con una interfaz gráfica amigable, un editor de código integrado y la capacidad de compilar y cargar programas directamente en los diferentes microcontroladores de Arduino (Arduino, 2022).

- **Tipos de microcontroladores de Arduino**

Las placas Arduino son la base de la plataforma y vienen en diferentes tipos, cada una con características específicas para adaptarse a diversas necesidades. Arduino ofrece una amplia variedad de placas para satisfacer las necesidades de diferentes proyectos y usuarios (Arduino, 2018).

La Tabla 5 resume las características técnicas de algunas de las placas Arduino más populares. Entre los aspectos clave que diferencian a estas placas como el voltaje de operación, la memoria disponible, la velocidad del reloj y los periféricos e interfaces integradas.

**Tabla 5**

*Tipos de placas de Arduino y sus características técnicas*

| Placa | Procesador | Voltaje de Alimentación | Memoria Flash | Memoria RAM | Velocidad de Reloj | Interfaces Importantes |
|---|---|---|---|---|---|---|
| Arduino Uno ($ 15) | ATmega328P | 5V | 32 KB | 2 KB | 16 MHz | USB, UART, I2C, SPI, GPIO |
| Arduino Mega ($ 25) | ATmega2560 | 5V | 256 KB | 8 KB | 16 MHz | USB, UART, I2C, SPI, GPIO |
| Arduino Nano ($ 15) | ATmega328P | 5V | 32 KB | 2 KB | 16 MHz | USB, UART, I2C, SPI, GPIO |

| | | | | | | |
|---|---|---|---|---|---|---|
| Arduino ESP32 ($ 15) | ESP32 de Espressif Systems (Doble núcleo a 240 MHz) | 3.3V | 4 MB y 16 MB | 520 KB | Hasta 240 MHz | USB, UART, I2C, SPI, GPIO. Wi-Fi y Bluetooth integrados (SIM[10] en unos casos) |

*Nota.* Adaptado de (Arduino, 2018)

Cada tipo de placa tiene sus propias características y ventajas, lo que permite a los creadores seleccionar la más adecuada para sus proyectos específicos.

Existe una amplia gama de placas Arduino para adaptarse a cualquier proyecto. Una vez seleccionada la placa, el siguiente paso es conectarle diferentes sensores y dispositivos para interactuar con el entorno.

En la siguiente sección se exponen detalles específicos de la antena GPS, componente central para el desarrollo del presente proyecto. La integración de la antena GPS con placas Arduino que tengan el soporte para GPS permite construir sistemas electrónicos capaces de percibir ubicación, logrando el desarrollo de soluciones innovadoras.

#### *2.4.1.1 Antena GPS*

La antena GPS es un dispositivo que utiliza una red de satélites para determinar la ubicación geográfica de un objeto, persona o vehículo.

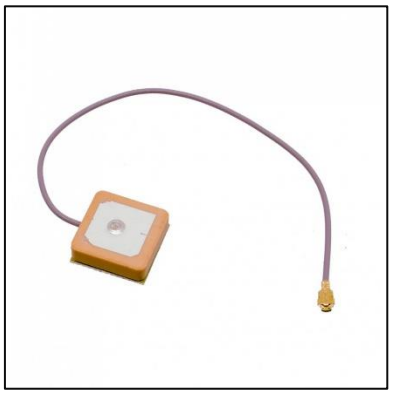
El GPS funciona midiendo el tiempo que tarda en llegar la señal de radio de cada satélite al sensor. Conociendo las distancias a los satélites, se puede calcular la longitud, latitud y altitud del objeto en cuestión de segundo (naylampmechatronics, 2016).

En la Figura 10 se puede visualizar una antena GPS la cual es utilizada en el presente trabajo para determinar la ubicación geográfica del carro recolector de basura.

[10] Subscriber Identity Module (Módulo de Identificación de Suscriptor).

**Figura 10**

*Antena GPS*



*Nota.* Adaptado de (naylampmechatronics, 2016)

Existen módulos diferentes módulos GPS que pueden interconectarse fácilmente con microcontroladores como Arduino o bien este traerlos ya integrados como en algunas soluciones existentes. La Tabla 6 resume las especificaciones técnicas básicas de un módulo Arduino GPS.

**Tabla 6**

*Características técnicas generales de los módulos Arduino GPS*

| **Características técnicas generales de los módulos Arduino GPS** | |
|---|---|
| Alimentación | 3.3V o 5V |
| Interfaz | UART, I2C o SPI |
| Rango de medición | Hasta 10,000 metros |
| Corriente de alimentación | 50 mA |
| Frecuencia de trabajo | 1 Hz - 10 Hz |
| Tipo de sistema/señal | GPS, GLONASS o Galileo |
| Dimensiones del módulo | 25 mm x 25 mm |
| Precisión | ±2.5 metros |
| Tiempo de adquisición GPS | Menos de 60 segundos |
| Sensibilidad | -165 dBm |
| Temperatura de operación | -20°C a 70°C |

*Nota.* Adaptado de (naylampmechatronics, 2016)

## 2.5 Componentes de Software del sistema de recolección de RSU

Después de analizar los conceptos clave sobre hardware y, en particular, las placas Arduino que brindan soporte físico a este proyecto, se realiza un estudio sobre el software que permite crear las funcionalidades de la aplicación móvil y su integración con el hardware, logrando una solución óptima.

El software es un componente fundamental en el desarrollo de soluciones tecnológicas, ya que proporciona las instrucciones y lógica necesarias para que los dispositivos de hardware puedan realizar tareas específicas. En el contexto del sistema de recolección de RSU, el software desempeña un papel crucial en la gestión de datos, la interacción con el usuario y la integración con los componentes de hardware (Ciset ES, 2021).

En el desarrollo de aplicaciones móviles, existen dos plataformas principales: Android e iOS. Estas plataformas ofrecen diferentes entornos de desarrollo, lenguajes de programación y herramientas para crear aplicaciones nativas para cada sistema operativo móvil.

### *2.5.1 Sistema Operativo Android*

Android es un sistema operativo móvil desarrollado por Google, basado en el núcleo Linux. Proporciona un conjunto de herramientas de desarrollo de software (SDK, por sus siglas en inglés) que permiten a los desarrolladores crear aplicaciones nativas utilizando principalmente los lenguajes de programación Java y Kotlin(Android Developers, 2023).

Para el desarrollo de aplicaciones Android existe el software “Android Studio IDE”, este es un software gratuito que proporciona herramientas para escribir código, depurar aplicaciones, realizar pruebas unitarias y de interfaz de usuario, y emular dispositivos Android para probar las aplicaciones antes de implementarlas. Además, Android Studio ofrece una amplia gama de características, como el editor de diseño visual, la integración con herramientas de control de versiones, el análisis de código estático y la compatibilidad con diferentes versiones de Android (Android Developers, 2023).Si bien Android Studio está diseñado

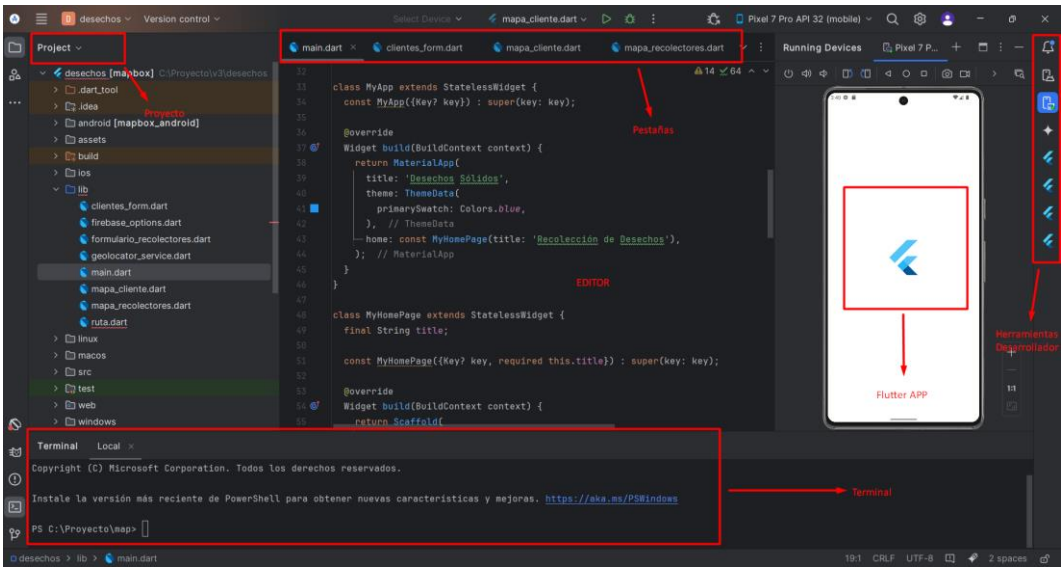
principalmente para el desarrollo nativo de aplicaciones Android, también es compatible con otros frameworks[11] multiplataforma como Flutter, que se aborda en la siguiente sección (Android Developers, 2023).

En la Figura 11, se puede observar el entorno de desarrollo de Android Studio utilizado para el desarrollo de una aplicación basada en Flutter. La parte izquierda muestra la estructura del proyecto actual, con sus respectivas dependencias y archivos .dart. En la parte superior, se encuentran las pestañas que conforman la estructura del proyecto, con diferentes archivos fuente en Dart.

En el panel derecho, se aprecia un dispositivo simulado que permite visualizar la aplicación Flutter en ejecución, junto con las herramientas de desarrollo de Flutter. En la parte inferior, se encuentra una ventana de línea de comandos que permite configurar o ejecutar el proyecto de forma manual, por último, el panel central contiene el Editor de código, donde se puede codificar y modificar los diferentes componentes de la aplicación.

**Figura 11**

*Partes básicas de la interfaz de Arduino Studio IDE con aplicación Flutter*




[11] Estructura de software predefinida que proporciona herramientas, bibliotecas y patrones de diseño para facilitar el desarrollo rápido y eficiente de aplicaciones.

### *2.5.2 Flutter*

Flutter es un framework de código abierto desarrollado por Google que permite crear aplicaciones nativas para Android e iOS[12] utilizando un solo código base escrito en el lenguaje de programación Dart. A pesar de que Flutter tiene su propio entorno de desarrollo integrado (IDE), también se puede utilizar dentro de Android Studio, lo que permite aprovechar las herramientas y características de este último para el desarrollo de aplicaciones Flutter(Flutter, 2023).

Una de las principales ventajas de Flutter es su arquitectura, que permite que la interfaz de usuario se actualice automáticamente ante cambios, lo que facilita el desarrollo de aplicaciones en tiempo real (Flutter, 2023)

Flutter, al ser un framework de código abierto desarrollado por Google, se integra de manera nativa con otras plataformas y servicios de Google, lo que facilita la creación de aplicaciones móviles completas y escalables. Una de las principales ventajas de utilizar Flutter en este proyecto es su capacidad para comunicarse y trabajar en conjunto con servicios en la nube, como Firebase (Flutter, 2023).

La Figura 12 muestra la arquitectura de una aplicación Flutter y cómo se integra con las plataformas móviles iOS y Android. En la parte central se encuentra la aplicación Flutter (cliente), que alberga el estado de la aplicación. Esta aplicación utiliza un canal MethodChannel para comunicarse con los hosts nativos de iOS y Android.

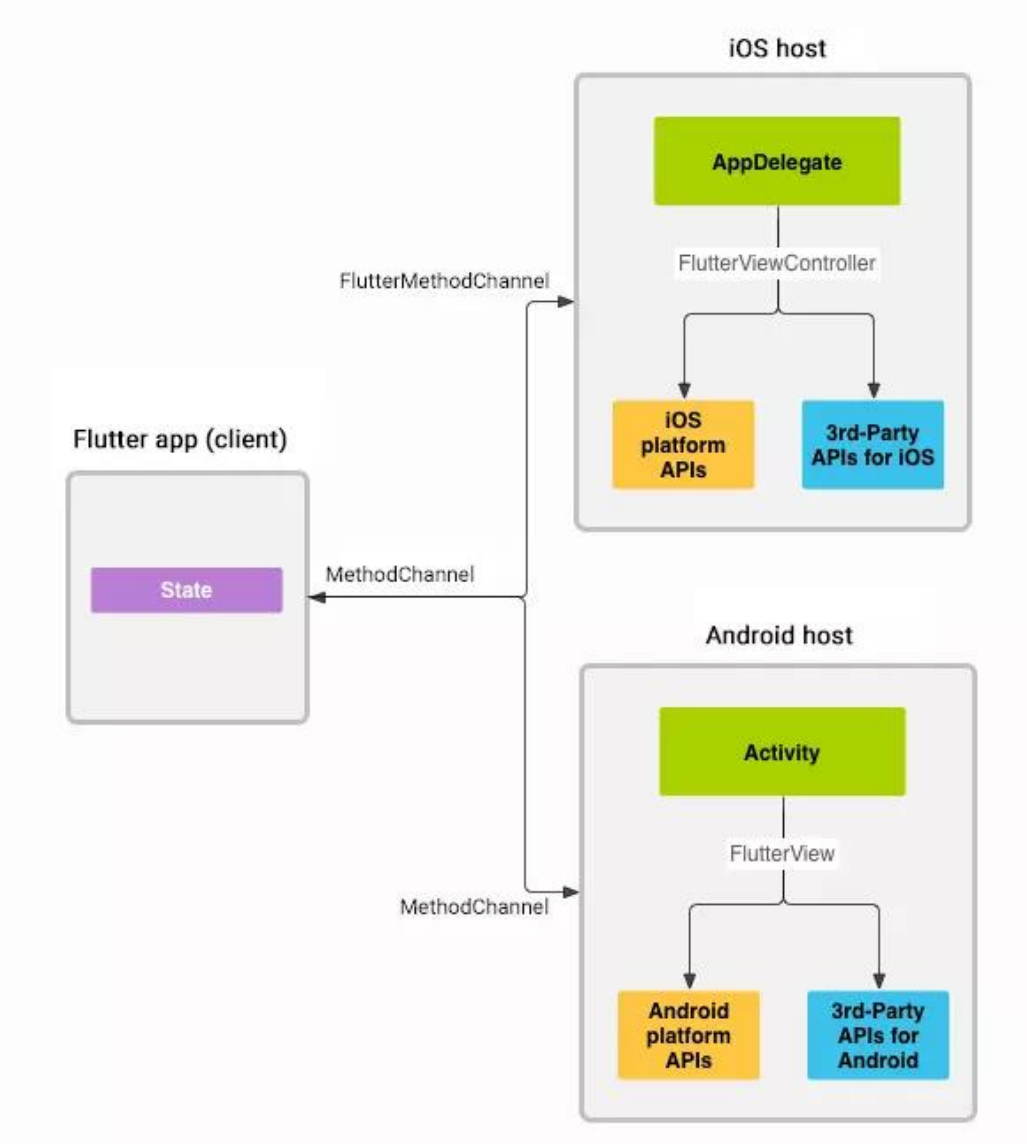
Por un lado, en el host de iOS, la aplicación Flutter interactúa con los componentes FlutterViewController y AppDelegate. Además, tiene acceso a las APIs nativas de la plataforma iOS y a APIs de terceros desarrolladas específicamente para iOS. Por otro lado, en el host de Android, la aplicación Flutter se comunica con los componentes FlutterView y

[12] Sistema operativo móvil desarrollado por Apple Inc.

Activity, y también puede acceder a las APIs nativas de la plataforma Android y APIs de terceros diseñadas para Android. Esta arquitectura permite que una aplicación Flutter se integre y aproveche las funcionalidades nativas de cada plataforma móvil, manteniendo una base de código compartida para ambas plataformas, lo que facilita el desarrollo multiplataforma(Bizzotto, 2018).

**Figura 12**

*Arquitectura de las aplicaciones desarrolladas con Flutter*




*Nota.* Tomado de (Bizzotto, 2018).

### *2.5.3 Firebase*

Firebase es una plataforma integral con planes gratuitos y de paga dependiendo del uso de servicios en la nube para el desarrollo de aplicaciones móviles y web, es desarrollada por Google y ofrece una variedad de herramientas y funcionalidades que simplifican el proceso de desarrollo, incluyendo bases de datos en tiempo real, autenticación de usuarios, hospedaje web, análisis de aplicaciones y notificaciones (Firebase, 2023).

En el contexto de este proyecto, se utilizan dos servicios principales de Firebase: Realtime Database y Cloud Firestore; Realtime Database es una base de datos alojada en la nube que permite almacenar y sincronizar datos en tiempo real. Los datos se almacenan como objetos JSON[13] y se actualizan automáticamente en todos los clientes conectados a la base de datos, lo que la convierte en una opción ideal para aplicaciones que requieren actualizaciones en tiempo real (Firebase, 2023).

Por otro lado, Cloud Firestore es otra base de datos en la nube, pero con un modelo de datos más flexible y escalable. Firestore organiza los datos en colecciones y documentos, lo que facilita el almacenamiento y la consulta de datos estructurados de manera eficiente (FTP Cloud, 2021). Tanto Realtime Database como Cloud Firestore se integran de manera nativa con Flutter, lo que simplifica el proceso de almacenamiento y recuperación de datos en una aplicación móvil.

Firebase, al ser una plataforma integral de servicios en la nube, no solo brinda soluciones para el almacenamiento de datos y la autenticación de usuarios, sino que también facilita la integración con diversas APIs[14] y servicios de terceros, tal como se muestra en la Figura 13, en donde se representa una arquitectura básica de Firebase y como sus diferentes

[13] Acrónimo de JavaScript Object Notation, es un formato de texto ligero utilizado para el intercambio de datos estructurados.. Es comúnmente utilizado para transmitir datos entre un servidor y una aplicación.

[14] Interfaz de programación de aplicaciones compuesta de un conjunto de definiciones y protocolos que permite diseñar e integrar el software de las aplicaciones.

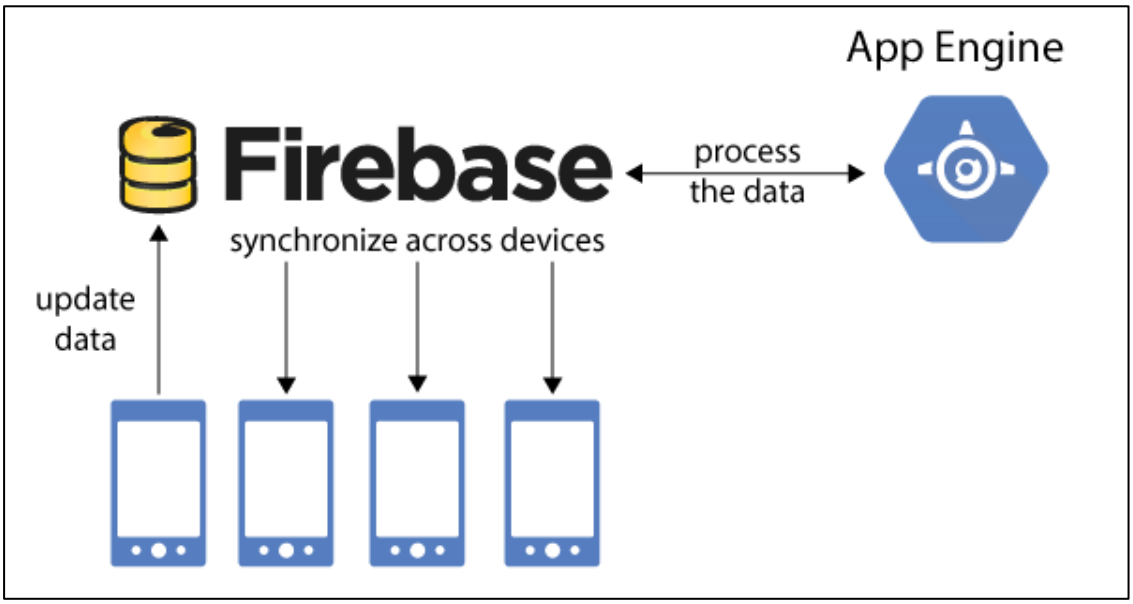
componentes interactúan para brindar una solución eficiente. En el centro, se muestra Firebase, que actúa como una base de datos en la nube. Su función principal es sincronizar los datos desde y hacia diferentes dispositivos móviles(FTP Cloud, 2021).

Firebase, a su vez, procesa esos datos y los envía al App Engine, esta a su vez puede realizar diversos procesamientos y operaciones con los datos recibidos de Firebase.

Esta arquitectura es útil para aplicaciones móviles que requieren compartir datos en tiempo real entre diferentes dispositivos y realizar procesamientos adicionales en el backend con App Engine(Google Cloud Platform Blog, 2015).

**Figura 13**

*Arquitectura Básica de Firebase*




*Nota.* Tomado de (Google Cloud Platform Blog, 2015).

### *2.5.4 Mapbox*

Mapbox es una plataforma de mapas y servicios de ubicación en la nube que ofrece una API de uso gratuito con limitaciones para integrar mapas, geocodificación y enrutamiento en aplicaciones móviles y web. La API de Mapbox proporciona diversas funcionalidades clave,

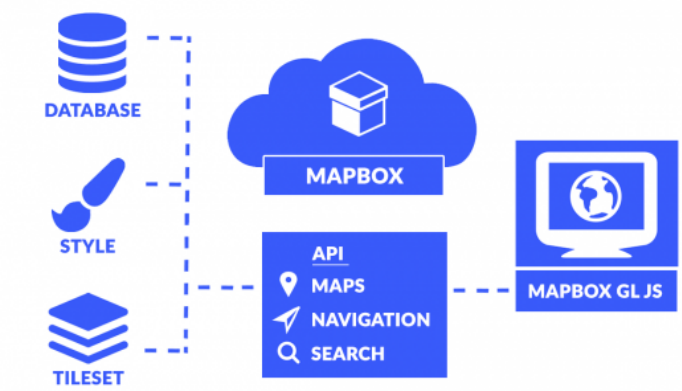
como la visualización de mapas interactivos, la búsqueda de direcciones, el cálculo de rutas optimizadas y la capacidad de agregar capas personalizadas (Mapbox, 2023).

La Figura 14 presenta la arquitectura de Mapbox, en donde se tiene un núcleo que aloja una variedad de servicios y APIs relacionados con mapas. La API de Mapbox ofrece una gama diversa de servicios, desde Maps, que proporciona mapas base y funcionalidades de renderizado, hasta Navigation, que ofrece capacidades de navegación y enrutamiento, y Search, que permite búsquedas de ubicaciones y lugares. Mapbox interactúa con diversos recursos y fuentes de datos, como una base de datos para almacenar información geográfica, conjuntos de tiles pre-renderizados conocidos como Tilesets, y estilos y configuraciones para personalizar la apariencia de los mapas(Mapbox, 2023).

En el lado del cliente, se muestra "Mapbox GL JS", una biblioteca de JavaScript que facilita la integración y el renderizado de mapas de Mapbox en aplicaciones web. Esta arquitectura permite a Mapbox ofrecer una amplia gama de servicios de mapas, navegación y búsqueda a través de su API en la nube (GisGeography, 2020).

**Figura 14**

*Arquitectura básica de Mapbox*




*Nota.* Tomado de (GisGeography, 2020).

En el caso particular del presente proyecto donde la visualización de mapas y el cálculo de rutas optimizadas son componentes esenciales, se hace necesaria la integración con una API

de mapas y servicios de ubicación. Es aquí donde la API de Mapbox es una solución ideal, al ofrecer una amplia gama de funcionalidades respecto a servicios de mapas, las cuales, combinadas con las capacidades de Firebase y Flutter, permiten desarrollar una aplicación móvil completa y eficiente para la gestión de la recolección de residuos sólidos urbanos.

De acuerdo con Mapbox (2023), en el contexto de optimización de rutas es importante considerar que la elección de uno o varios algoritmos puede variar según el escenario específico, y Mapbox probablemente emplea una combinación estratégica de algoritmos de código abierto como también propios, adaptándose dinámicamente a diversas situaciones y requerimientos específicos de cada escenario de ruteo. La Tabla 7 presenta una comparación detallada de los principales algoritmos para optimización de rutas que posiblemente utilice Mapbox, destacando sus características clave y aplicaciones óptimas. Para cada uno, se define su funcionamiento básico, escenarios de aplicación ideales, velocidad para encontrar una solución, precisión de resultados, necesidad de procesamiento y dificultad de implementación.

**Tabla 7**

*Comparación de Algoritmos para la Optimización de Rutas*

| Algoritmo | ¿Qué hace? | Eficiente para | Velocidad | Precisión | Necesidad procesamiento | Dificultad implementación |
|---|---|---|---|---|---|---|
| **Dijkstra** | Encuentra la ruta más corta desde un nodo origen a todos los demás. | Rutas en áreas urbanas donde las distancias no son extremadamente largas. | Media | Media | Alta | Media |
| **A* (A star)** | Busca la ruta más eficiente usando una estimación inteligente. | Rutas largas y complejas | Alta | Alta | Alta | Media |

| | | | | | | |
|---|---|---|---|---|---|---|
| **Vecino más cercano** | Elige el siguiente punto más cercano de cada punto | Rutas simples con muchas paradas | Alta | Baja | Baja | Baja |
| **Algoritmo Genético** | Prueba muchas rutas posibles y las mejora gradualmente | Problemas complejos con muchas variables | Baja | Media | Alta | Alta |
| **Búsqueda Local** | Mejora una ruta existente intercambiando partes de ella | Mejorar rutas ya existentes | Alta | Media | Media | Media |

## 2.6 Metodología en Cascada

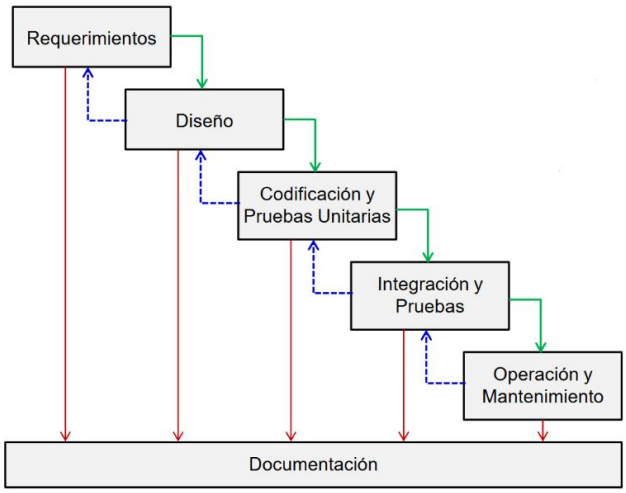
Una vez analizados los aspectos de hardware y software, es importante enmarcar el proyecto dentro de una metodología que permita un desarrollo secuencial. Esto posibilita el desarrollo exitoso de una solución óptima para el proceso de recolección de Residuos Sólidos Urbanos (RSU).

La metodología en cascada, también conocida como Waterfall, es una forma sencilla de gestionar un proyecto dividiéndolo en distintas fases que se desarrollan de manera secuencial, una después de la otra, como si fueran una cascada, es decir, se completa una fase y luego se pasa a la siguiente. Esto significa que cada fase depende de que la anterior se haya finalizado satisfactoriamente(Instituto Europeo de Postgrado - IEP, 2022).

Las principales fases se pueden visualizar en la Figura 15, que además permite entender el proceso ordenado que sigue esta metodología en donde existen además flechas de color verde que muestran el flujo secuencial normal entre las etapas y las flechas azules discontinuas que representan posibles iteraciones o retroalimentación entre algunas fases si fuera necesario. (Tapias, 2014).

**Figura 15**

*Fases de la metodología en Cascada*




*Nota.* Tomado de (Instituto Europeo de Postgrado - IEP, 2022; Tapias, 2014)

En el presente trabajo de titulación, primero se plantean los requisitos del sistema analizando los requerimientos específicos de los usuarios del sistema. Luego se realiza el diseño, dividiendo el trabajo en partes separadas tanto para software como hardware. Después, se codifica y se realiza las validaciones y pruebas por separado. A continuación, se integran todas las partes y se realiza las pruebas y verificaciones al sistema completo. Finalmente, en la etapa de mantenimiento se realizan cambios necesarios, como corregir errores o adaptaciones del sistema a las características del entorno. Cada fase está documentada, de esta forma se evidencia el seguimiento secuencial de proyecto

Uno de los aspectos más importantes es planificación inicial es clave, ya que un retraso en una fase impacta en cascada al resto y retrasa todo el proyecto. Una ventaja de la metodología en cascada es que, al definir los requisitos al comienzo, queda claro el alcance, las tareas y se puede estimar tiempo con precisión. Sin embargo, también esta metodología es menos flexible a cambios posteriores, que resultan más lentos y costosos de implementar. Si una fase se retrasa, se retrasa todo el proyecto (Instituto Europeo de Postgrado - IEP, 2022).

# CAPÍTULO III: DISEÑO DEL SISTEMA DE RECOLECCIÓN DE RSU

En este capítulo, se detalla el proceso para el diseño del sistema de recolección de residuos sólidos urbanos (RSU) del barrio Centenario, siguiendo la metodología en cascada para llevar un proceso ordenado, por tanto, se desarrolla la fase I sobre los requerimientos, en la cual se realiza un análisis de las características del barrio Centenario con relación a la recolección de los RSU para identificar los actores involucrados (stakeholders) en el proyecto y sus necesidades más demandadas para mejorar este proceso, esto permite establecer los componentes idóneos de hardware/software para el sistema y sus funcionalidades.

Una vez desarrollado el apartado de requerimientos en la fase I de la metodología en cascada, se procede a la fase II, que comprende el Diseño del sistema de recolección de RSU. Este diseño integra adecuadamente los componentes de hardware y software seleccionados durante el desarrollo de la fase I de la metodología en cascada, cada elemento trabaja de manera complementaria para crear una solución integral y eficiente, que permita gestionar eficazmente la recolección de residuos sólidos urbanos en el sector del barrio Centenario.

## 3.1 Análisis de Requisitos del sistema de recolección de RSU

En este apartado se establece un análisis detallado de los requisitos necesarios para el desarrollo exitoso del proyecto, se considera importante la realización de un análisis del entorno para verificar las características del Barrio Centenario con relación a la recolección de RSU para asegurar que el proyecto cumpla con los estándares de diseño y dar solución a la problemática del sector.

Esta etapa es crucial debido a que permite establecer las bases para el desarrollo de las siguientes fases del proyecto, a continuación, se describe la situación actual del proceso de recolección de RSU en el barrio Centenario.

### *3.1.1 Situación Actual*

La situación actual es esencial para comprender la problemática que se tiene en el barrio Centenario, perteneciente al cantón Montufar, con relación al proceso de recolección de RSU, a partir de la examinación de su situación se plantea una solución óptima a través del diseño de un sistema para mejorar el proceso de recolección en el sector.

El barrio Centenario, es un sector urbano de la ciudad de San Gabriel, perteneciente al Cantón Montufar, este en los últimos años ha estado en constante crecimiento, de acuerdo con GADM Montúfar (2020), actualmente existe un aproximado de 600 predios en el barrio Centenario, en los cuales están construidas casas en su mayoría de gran extensión, que sirven como viviendas de arriendo para personas que se ven atraídas por el bajo costo del alquiler y las oportunidades de trabajo en el sector agrícola, por tanto, entre más pobladores se genera más cantidad de desechos; considerando el continuo crecimiento de la generación de RSU en el sector, se ha implementado un sistema de gestión de RSU, este incluye un proceso de recolección que al pasar de los días se ha vuelto ineficiente y ha creado nuevas problemáticas en el sector, a diario se puede observar desechos sobre las aceras o calles ocasionando contaminación ambiental/visual, por tal motivo, es necesario implementar nuevas soluciones para realizar un proceso de recolección más adecuado en el barrio Centenario, en donde pueda existir una interacción directa entre recolector y habitantes del sector para evitar que los residuos sean dejados afuera de los domicilios durante tiempos extensos u horarios incorrectos.

La contaminación ambiental/visual se origina debido a que la mayoría de los residentes del sector sacan sus desechos antes que el recolector pase por sus viviendas ocasionando que estos se salgan de sus recipientes o fundas y se esparzan por el lugar por la acción del viento o por causa de animales (perros y gatos), los cuales rompen el empaque en busca de comida como

se puede visualizar en la Figura 16a y que además genera proliferación de plagas[15],por tanto, es necesario un sistema para que los usuarios conozcan el horario adecuado para sacar sus residuos.

**Figura 16**

*Contaminación Visual/Ambiental en el Barrio Centenario en la noche*

**(a)** **(b)**

De acuerdo con los habitantes del sector, el recolector de basura ingresa al barrio Centenario aproximadamente a las 8:00 am de martes a domingo, después de recoger los desechos casa por casa en el barrio colindante. Este proceso tiene una duración que va acorde con el tamaño de barrio, en el barrio Centenario tarda alrededor de 2 horas, aunque no todos los habitantes del sector sacan su basura a diario, otros irrespetan los horarios, mientras que pocos lo hacen cuando escuchan la cercanía del carro recolector mediante la notificación sonora que este posee.

Según los moradores, en algunos casos la recolección se realiza con vehículos inadecuados que no cuentan con la notificación sonora, debido a esto, los moradores sacan sus residuos minutos u horas antes que el recolector pase por sus domicilios (Figura 16b), la

[15] Crecimiento descontrolado de plagas (insectos, roedores, hongos, bacterias, virus, etc.)

mayoría se ven obligados a realizar estas acciones incorrectas, ya que en los horarios de recolección se encuentran trabajando fuera de su domicilio; existen alta cantidad de moradores que evitan sacar sus residuos todos los días y lo hacen en sus días libres. Por lo tanto, es común que el recolector pase varias veces por una misma vivienda a la semana sin que se realice recolección alguna.

Por lo que, si se conocieran previamente los lugares y horarios en los que se realizará la recolección, los moradores podrían sacar sus residuos a la hora correcta, y el recolector podría llegar a puntos específicos. De esta forma, se evitaría que los desechos permanezcan expuestos en la calle por tiempos prolongados, generando contaminación. Además, se lograría optimizar el tiempo de recolección de residuos sólidos urbanos en el sector, para luego dirigirse al siguiente barrio.

Tras describir y analizar la situación actual del barrio Centenario a partir de las experiencias relatadas por los moradores, se evidencia la necesidad de implementar mecanismos para mejorar el proceso de recolección, considerando horarios y notificaciones eficientes. Los vecinos requieren un sistema que les permita notificar previamente al inicio de la recolección que tienen residuos listos para ser recogidos, y a su vez, ser notificados del tiempo estimado en que el recolector llegará a su domicilio. De esta forma, evitarían sacar la basura antes de tiempo, previniendo la contaminación. Asimismo, el recolector podría seguir una ruta que contemple únicamente los puntos de recolección solicitados, sin realizar recorridos innecesarios, ahorrando así tiempo en el proceso.

### *3.1.2 Encuesta*

Luego de establecer la situación actual del entorno donde se aplica el proyecto, se evidencia la falta de datos estadísticos que respalden la información proporcionada por los moradores. Por lo tanto, se implementa una técnica que permite obtener datos cuantitativos para fundamentar el desarrollo del proyecto.

La encuesta es una técnica que permite recopilar datos importantes para la investigación. Esta técnica permite obtener información real y actualizada sobre la problemática en dicho sector.

Además, el diseño de una solución óptima requiere consultar directamente a los usuarios e identificar sus necesidades. Por tanto, es fundamental obtener su retroalimentación para asegurar que el diseño satisfaga sus requerimientos. En la Tabla 8 se especifica el método y formato empleado para obtener los datos necesarios que sustenten el diseño de un sistema de recolección de Residuos Sólidos Urbanos (RSU) en el barrio Centenario, con el fin de generar un impacto social y ambiental positivo para los habitantes.

**Tabla 8**

*Método y formato para obtención de datos para la sustentación del proyecto*

| | **Método y Formato para la obtención de datos** |
|---|---|
| **Método:** | Se utiliza el método de investigación descriptiva mediante una encuesta para recolectar información. Esta técnica ha sido seleccionada con el fin de obtener datos reales sobre el proceso de recolección de Residuos Sólidos Urbanos (RSU) en el barrio Centenario y definir funcionalidades importantes para el diseño del sistema de recolección. Las encuestas se aplican a los propietarios de inmuebles en dicho barrio. |
| **Formato:** | La encuesta consta de 13 preguntas. Las primeras 12 son cerradas, ofreciendo opciones para que el usuario elija, mientras que la última es abierta, permitiéndole brindar sugerencias para el diseño de un sistema óptimo de recolección de Residuos Sólidos Urbanos (RSU).<br>El objetivo es recopilar datos estadísticos sobre la generación de residuos en hogares, la frecuencia de recolección, la percepción del impacto de la contaminación, la necesidad de un sistema de recolección de RSU, la disponibilidad de conexión a internet para su implementación, así como las funcionalidades específicas requeridas. La encuesta completa se adjunta en el Anexo 1. |

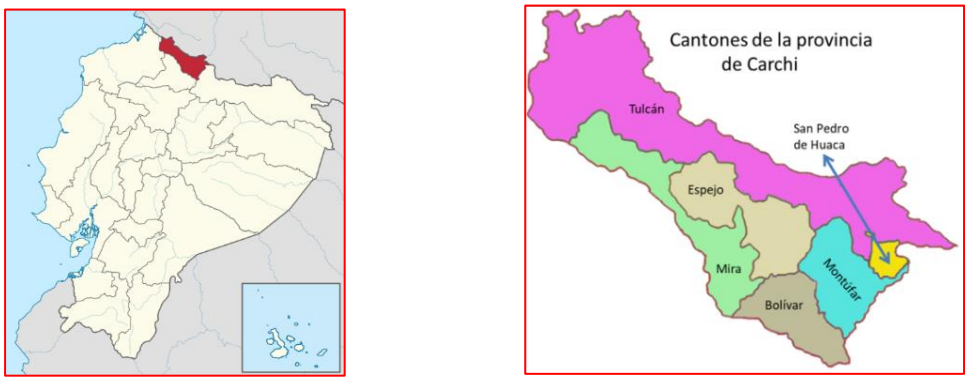
Con el fin de obtener resultados que reflejen fielmente la situación actual y las necesidades de los habitantes del barrio Centenario en relación con el proceso de recolección de Residuos Sólidos Urbanos (RSU), es fundamental establecer el tamaño de la población de este sector. Dado que este trabajo de titulación se centra en la optimización de la recolección domiciliaria de RSU, es crucial conocer el número de viviendas que conforman el barrio. Para ello, es necesario primero ubicar geográficamente el sector, como se detalla en el siguiente apartado.

#### *3.1.2.1 Localización del Barrio Centenario*

Como se muestra en la Figura 17a , el cantón Montúfar se ubica en la región norte del Ecuador, es parte de la provincia del Carchi, integrante de la zona 1 de la planificación a nivel nacional. Su territorio limita con la cuenca del Valle del Chota y conforma las cuencas alta y media del rio Apaquí, sus límites territoriales son (ver Figura 17b): al Norte el Cantón Tulcán; al Sur el Cantón Bolívar; al Este Cantón San Pedro de Huaca y al Oeste el Cantón Espejo(GAD-Montúfar, 2020).

**Figura 17**

*Macro localización del cantón Montufar, ciudad San Gabriel*



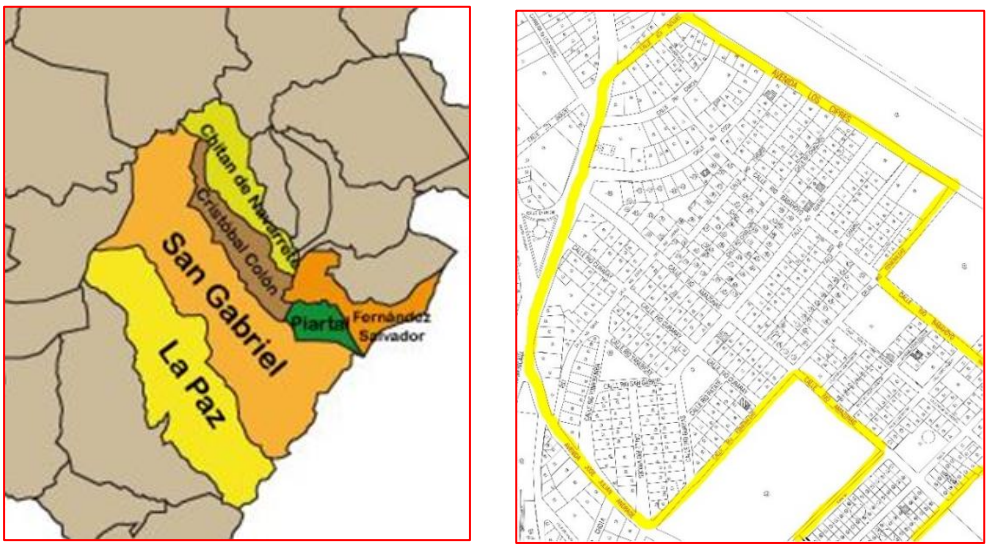

**(a)** **(b)**

*Nota.* Adaptado de (GADM Montúfar, 2020)

El cantón Montúfar está conformado por las parroquias La Paz, San Gabriel, Cristóbal Colón, Chitán de Navarretes, Piartal y Fernández Salvador, como se muestra en la Figura 18a. El barrio Centenario se ubica en la parte sur de la ciudad de San Gabriel y es uno de los sectores más extensos de la ciudad. De acuerdo con el mapa catastral del barrio Centenario, que se muestra en la Figura 18b, se contabilizan 600 residencias en este sector.

**Figura 18**

*Micro localización del Barrio Centenario*




**(a)** **(b)**

*Nota.* Adaptado de (GADM Montúfar, 2020)

### *3.1.2.2 Cálculo muestral*

La investigación se lleva a cabo en el barrio Centenario, ubicado en el cantón Montúfar. Para recopilar información, se toma una muestra de la cantidad de domicilios del sector, de acuerdo con el mapa catastral proporcionado por el GADM Montúfar (ver Figura 18b). Las encuestas son aplicadas a los propietarios de cada inmueble. Considerando que existen 600 predios en el barrio Centenario, este número representa el tamaño de la población para el

estudio. El tamaño de la muestra se calcula utilizando la Ec. 1, propuesta por los autores Murray y Larry (2005).

$$n = \frac{NZ^2\sigma^2}{(N-1)e^2 + Z^2\sigma^2} \qquad \textbf{Ec. 1}$$

Donde:

- $\boldsymbol{n}$ = Representa el tamaño de la muestra.
- $\boldsymbol{\sigma}$ = Es la proporción de individuos que tienen en la población la característica de estudio. Este dato es habitualmente incógnito y se suele suponer que 0.5 que es la elección más convincente.
- $\boldsymbol{N}$ = Representa el tamaño de la población.
- $\boldsymbol{Z}$ = Representa el nivel de confianza, es decir, un valor constante y generalmente se presenta con el 95%, equivalente a 1,96 siendo el valor mínimo aceptable.
- $\boldsymbol{e}$ = Significa el límite aceptable de error muestral que, habitualmente suele usarse un valor entre el 1% (0,1) y 9% (0,9), se selecciona el valor mínimo de error aceptable para obtener una respuesta más acertada.

Por tanto, se considera los siguientes datos: $\boldsymbol{\sigma}$ = 0,5; $\boldsymbol{N}$ = 600; $\boldsymbol{Z}$ = 1,96 y $\boldsymbol{e}$ = 0,09

$$n = \frac{500(1{,}96)^2(0{,}5)^2}{(500-1)(0{,}09)^2 + (1{,}96)^2(0{,}5)^2} = 95{,}99 \cong 96\ Encuestas$$

Luego de reemplazar los datos en la Ec.1 y realizar los cálculos matemáticos correspondiente, se determina que se debe aplicar un total de 96 encuestas de acuerdo con el formato de la Tabla 8, lo que permite obtener resultados con un nivel de confianza del 95%.

#### *3.1.2.3 Resultados de la encuesta aplicada*

Tras aplicar la encuesta detallada en el **Anexo 1** a 96 propietarios de inmuebles en el barrio Centenario, se analizan los resultados obtenidos según lo expuesto en la tabulación de la encuesta presente en el **Anexo 2**. Estos resultados evidencian la necesidad de llevar a cabo este proyecto para solucionar la problemática existente en el sector, demostrando que es una solución viable para el barrio Centenario. Los principales hallazgos son los siguientes:

- **Pregunta 1:** ¿Cuántas personas viven en su domicilio?

  Los resultados de la encuesta revelan que, alrededor del 75% de los encuestados viven con 3 o más personas, lo que implica un tamaño familiar mediano a grande, por tanto, se apunta a una mayor generación de residuos, esto resalta la importancia de contar con un buen servicio de recolección.

- **Pregunta 2:** Aproximadamente, ¿Qué cantidad de residuos sólidos urbanos genera su hogar a la semana? (*Considérese como medida un costal*)

  Los resultados de la encuesta demuestran que, cerca del 45% de los encuestados generan entre 1 y 2 costales de residuos semanalmente, mientras que aproximadamente el 28% generan más de 2 costales. Estas altas cantidades de desechos generados evidencian la necesidad de implementar técnicas adecuadas para el manejo de residuos a nivel doméstico.

- **Pregunta 3:** ¿Con qué frecuencia saca sus residuos sólidos para recolección?

  Los resultados de la encuesta determinan que, un 48.45% de los encuestados sacan la basura 2 veces por semana, sugiriendo una frecuencia baja. Un 23.71% la sacan 3 veces o más, esto indica que el recolector actualmente pasa por domicilios que no requieren recolección diaria, por tanto, se evidencia la necesidad de implementar un sistema óptimo de recolección para evitar recorridos innecesarios.

- **Pregunta 4:** Respecto a la infraestructura y recursos disponibles (*camiones recolectores, alarma sonora, personal*) del GADM Montufar para la recolección de los residuos sólidos urbanos, usted los considera:

  Los resultados de la encuesta señalan que, aproximadamente el 75% de los encuestados califican la infraestructura actual como buena, y alrededor del 14% la consideran muy buena, estos resultados demuestran la necesidad de seguir trabajando en mejoras continuas para garantizar una mejor percepción.

- **Pregunta 5:** ¿Han existido demoras en los horarios de recolección actual de sus residuos sólidos?

  De acuerdo con los resultados de la encuesta, aproximadamente el 67% de los encuestados indican que rara vez experimentan demoras, mientras que alrededor del 22% mencionan que estas son frecuentes. Esto evidencia que, existe un porcentaje considerable de usuarios que enfrentan este problema. La implementación de un sistema para optimizar las rutas podría reducir las demoras.

- **Pregunta 6:** ¿Está de acuerdo con los horarios actuales de recolección de los residuos sólidos urbanos?

  Los resultados de la encuesta determinan que, alrededor del 80% de los encuestados están de acuerdo con los horarios actuales de recolección, aunque es importante considerar las preferencias de la minoría para mejorar la satisfacción general.

- **Pregunta 7:** ¿Cómo percibe el impacto de las bolsas de basura expuestas en los exteriores de los hogares y la acción de la fauna urbana en romper las fundas de residuos, generando contaminación visual/ambiental?

Los resultados determinan que, más del 90% de los encuestados perciben como muy perjudicial o algo perjudicial el impacto de las bolsas de basura expuestas en los exteriores de los hogares, lo que destaca la importancia de implementar soluciones efectivas para mitigar este problema.

- **Pregunta 8:** ¿Cómo considera que es el sistema actual de recolección de residuos sólidos?

  De acuerdo con los resultados de la encuesta se sugiere que, aunque el 72% de los encuestados lo consideran bueno y el 16% muy bueno al sistema actual de recolección actual, es necesario mantener una mejora continua para mantener altos estándares de servicio.

- **Pregunta 9:** ¿ Cree que sería útil la implementación de un sistema de recolección de residuos sólidos urbanos, que cuente con una aplicación móvil para solicitar y ser notificado sobre la recolección de los residuos?

  Los resultados de la encuesta demuestran que, alrededor del 70% de encuestados cree que sería muy útil la implementación de un sistema de recolección con una aplicación móvil para solicitar y ser notificado, lo que permite demostrar la aceptación hacia esta solución tecnológica.

- **Pregunta 10:** ¿Dispone de conexión a internet en su hogar?

  Los resultados de la encuesta demuestran que, aproximadamente el 96% de los encuestados disponen de conexión a internet, lo que indica la alta accesibilidad a internet y su capacidad para utilizar un sistema en línea para la recolección de sus residuos.

- **Pregunta 11:** ¿ Qué tipo de conexión a internet utilizan en su hogar? (Puede elegir más de una opción)

  Los resultados de la encuesta reflejan que, la mayoría (67%) prefiere utilizar Wi-Fi para acceder a Internet, 18% prefiere los datos móviles y un 14% tiene la flexibilidad de utilizar cualquiera de las dos tecnologías, evidenciando una alta disponibilidad de conexión a internet por parte de los usuarios y la viabilidad para implementar un sistema que requiera acceso a Internet.

- **Pregunta 12:** Defina su criterio de utilidad respecto a las siguientes funciones de la aplicación del sistema de recolección de residuos sólidos urbanos

**Opción A** - Visualizar en tiempo real la ubicación del camión recolector mediante la aplicación.

**Opción B** - Visualizar el tiempo aproximado de llegada del carro recolector al domicilio.

**Opción C** - Emitir una notificación cuando el camión recolector este próximo a llegar al domicilio.

**Opción D** - Emitir una notificación de llegada del camión relector.

De acuerdo con los resultados de la encuesta se refleja que, las funcionalidades más valoradas son visualizar la ubicación en tiempo real, el tiempo aproximado de llegada del camión recolector y la emisión de notificaciones cuando esté próximo a llegar, cerca del 90% de los encuestados las considera útiles o muy útiles, esto demuestra la importancia de la información y notificaciones en tiempo real.

- **Pregunta 13:** Tiene alguna sugerencia que le gustaría que se considere en el sistema de recolección de residuos sólidos.

  Los resultados de la encuesta sugieren que, debe de existir un aumento en la cantidad de camiones recolectores, mejorar horarios de recolección,

implementar políticas para la separación de residuos, desplegar contenedores de residuos y control de animales. Esto indica que, aunque el servicio actual es aceptable, existe aún un margen amplio para optimizarlo y atender las necesidades específicas de ciertos grupos de moradores.

### *3.1.3 Descripción general del sistema de recolección de RSU*

En esta sección se detalla el funcionamiento general del sistema de recolección de RSU, el cual integra diferentes tecnologías de hardware y software para crear un sistema eficiente. La Figura 19 muestra cómo se compone e interactúa el sistema de recolección de RSU, se observa la forma de obtención de datos del GPS del recolector mediante un prototipo constituido por una placa de procesamiento que incluye los módulos GPRS/GSM y GPS con sus respectivas antenas. Este prototipo se ubica estratégicamente en el carro recolector sin interferir con el espacio del conductor.

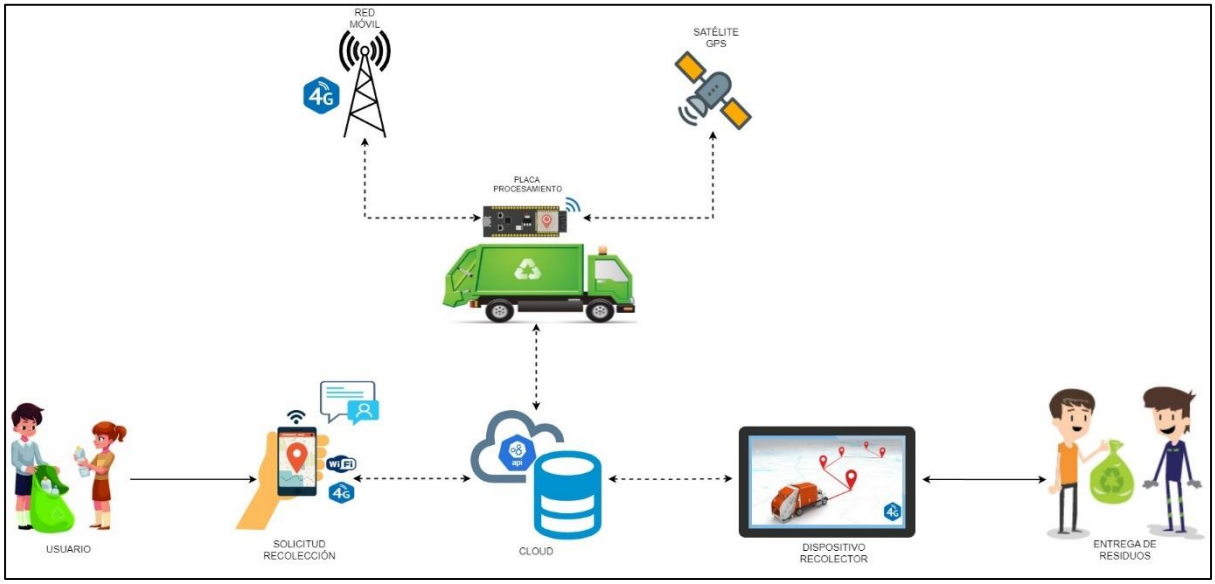
La placa de procesamiento integra el módulo GPRS/GSM y lo activa mediante la SIM de una operadora móvil para conectarse a la red móvil, luego envía los datos del GPS al cloud para su almacenamiento y procesamiento. Esta placa se alimenta energéticamente mediante una fuente de batería externa, que debe recargarse periódicamente para garantizar la disponibilidad del sistema.

El sistema contempla una aplicación móvil con dos modos de ingreso; uno para conductor del recolector en donde se puede visualizar en interfaces los puntos de recolección y la ruta de recolección, mientras que para el modo de cliente se visualiza la interfaz para llenar los datos de la solicitud de recolección y una segunda interfaz para verificar el tiempo aproximado de llegada, distancia y ruta del recolector respecto a la ubicación del cliente, además de recibir notificaciones importantes sobre la recolección de sus residuos, esto permite que el cliente y el recolector interactúen en tiempo real para mejorar el sistema de recolección de residuos en el sector.

El cliente cuenta con una restricción de horario para enviar las solicitudes de recolección, si se encuentra dentro del horario correcto, la solicitud se envía con los datos del cliente y su ubicación, esta información se almacena en una base de datos y luego es procesada con herramientas adecuadas para generar la ruta de recolección.

**Figura 19**

*Representación gráfica de la composición e interacción del Sistema de recolección de RSU con los usuarios.*




Luego de definir como se compone e interactúa el Sistema de recolección de RSU, es necesario realizar un análisis detallado de los requerimientos para su correcta operación. Este análisis permite identificar las necesidades específicas y garantizar que el sistema cumpla con los objetivos propuestos de manera efectiva. En la siguiente sección, se abordan los requerimientos del Sistema de recolección de RSU, considerando los diferentes componentes y actores involucrados.

### *3.1.4 Requerimientos del sistema de recolección de RSU*

Basado en los fundamentos teóricos presentado en el capítulo 2, además del análisis de la situación actual y los resultados de la encuesta aplicada descritos en el presente capítulo, se

obtienen los datos para establecer las características esenciales que el sistema debe poseer, así como las limitaciones que deben ser consideradas durante su diseño.

Para definir de los requerimientos del sistema, es fundamental identificar a los stakeholders o partes interesadas involucradas en el proyecto. Estos desempeñan un papel crucial en la definición de los requisitos y en la evaluación del desempeño del sistema.

#### *3.1.4.1 Determinación de los Stakeholders*

En el marco de este proyecto, los stakeholders o partes interesadas engloban a aquellos individuos o grupos que poseen un alto interés en el resultado exitoso del mismo. Estos pueden beneficiarse de manera directa o indirecta del proyecto, por lo que es importante identificar y comprender las necesidades, expectativas y roles de cada uno de estos.

La Tabla 9 detalla los involucrados en el proyecto, a quienes se debe tener en cuenta en todas las fases de desarrollo para contribuir al éxito general del Sistema de recolección de RSU.

**Tabla 9**

*Descripción de Stakeholders*

| Stakeholders | Función |
|---|---|
| Usuarios del sistema | Habitantes del barrio Centenario. |
| GADM Montufar | Administradores del territorio |
| Ing. Jaime Michilena, MSc. | Director del trabajo de titulación |
| Ing. Mauricio Domínguez, MSc. | Asesor del trabajo de titulación |
| Cristian Quelal | Desarrollador del trabajo de titulación |

En el diseño del sistema cada requerimiento es evaluado rigurosamente con el fin de cumplir los objetivos planteados. Se validan tres aspectos fundamentales: los requerimientos de los actores involucrados (stakeholders), los requerimientos del sistema en sí mismo y los requerimientos de arquitectura. El desarrollo del proyecto se sustenta en estos tres pilares

esenciales, los cuales se identifican mediante abreviaturas específicas, como se muestra en la Tabla 10.

**Tabla 10**

*Abreviatura de los requerimientos*

| Abreviatura | Requerimiento |
|---|---|
| **RDST** | Stakeholders |
| **RDS** | Sistema |
| **RDA** | Arquitectura |

Para determinar el grado de importancia de los requerimientos es importante asignar niveles de prioridad. La asignación de niveles de prioridad se fundamenta en un análisis de diversos factores clave, entre los que se encuentran: el grado de influencia en el logro de los objetivos propuestos para el proyecto, la contribución al adecuado desempeño y operatividad del sistema, y el nivel de impacto en la satisfacción de los requerimientos planteados por los clientes o usuarios finales del sistema.

La asignación de prioridades se detalla en la Tabla 11 y facilita la toma de decisiones durante el desarrollo del proyecto, permitiendo una gestión eficiente de los recursos del proyecto.

**Tabla 11**

*Prioridad de los requerimientos*

| Prioridad | Descripción |
|---|---|
| Alta=**Rojo** | Los requerimientos de alta prioridad son fundamentales e imprescindibles para el correcto funcionamiento del sistema. Su cumplimiento es crucial para alcanzar los objetivos principales del proyecto. |

| | |
|---|---|
| Media=**Verde** | Los requerimientos de prioridad media son importantes, pero no críticos. Su implementación contribuye al óptimo desempeño del sistema, aunque su omisión no impediría su operación básica. |
| Baja=**Azul** | Los requerimientos de baja prioridad tienen un impacto menor en el sistema. Su ausencia no afecta al cumplimiento de los objetivos principales, pero pueden aportar mejoras o funcionalidades adicionales. |

#### *3.1.4.2 Requerimientos de Stakeholders*

Los requerimientos de los actores involucrados o stakeholders son fundamentales para el desarrollo exitoso del sistema, ya que definen el conjunto de requisitos que sirven como base y guía durante todo el proceso. Al abordar los requerimientos de los distintos actores clave, es posible diseñar un sistema que cumpla con los objetivos del presente trabajo de titulación y brinde una solución integral a la problemática identificada.

En la Tabla 12 se establecen los requerimientos de los stakeholders para el diseño del sistema con la asignación respectiva de su prioridad. Estos requerimientos iniciales sirven para asegurar que la construcción del sistema cumpla con las necesidades de los actores involucrados.

**Tabla 12**

*Requerimientos de Stakeholders*

| **Requerimientos de Stakeholders RDST** | | | | |
|---|---|---|---|---|
| **#** | **Requerimiento** | **Prioridad** | | |
| | | **Alta** | **Media** | **Baja** |
| **RDST1** | Se debe tener conexión a internet para transmitir datos. | | | |
| **RDST2** | Es necesario una aplicación móvil para el sistema. | | | |
| **RDST3** | La aplicación móvil debe tener dos modos de ingreso: cliente y recolector. | | | |

| | |
|---|---|
| **RDST4** | La aplicación móvil debe permitir a los clientes solicitar la recolección de residuos en un horario predefinido y recibir notificaciones sobre el proceso de recolección. |
| **RDST5** | La aplicación móvil debe permitir al cliente visualizar la ubicación del recolector. |
| **RDST6** | La aplicación móvil debe notificar al cliente sobre el tiempo y distancia aproximada de llegada. |
| **RDST7** | La aplicación móvil debe notificar al usuario sobre la proximidad del recolector al punto de recolección. |
| **RDST8** | La aplicación móvil debe de notificar al usuario sobre la llegada del recolector al punto de recolección. |
| **RDST9** | La aplicación móvil debe permitir al conductor visualizar todos los puntos de recolección. |
| **RDST10** | La aplicación móvil debe permitir al conductor visualizar la ruta recolección. |
| **RDST11** | El dispositivo de recolección de datos debe incluir un compartimiento o ranura para alojar una batería, de modo que pueda obtener su alimentación de energía de forma autónoma. |

### *3.1.4.3 Requerimientos de Sistema*

Los requerimientos de sistema son fundamentales para guiar el diseño y desarrollo del sistema de recolección de RSU. Estos requerimientos definen las características técnicas que el sistema debe cumplir para satisfacer las necesidades de los stakeholders y abordar de manera efectiva las problemáticas identificadas.

En la Tabla 13 se presentan los requerimientos de sistema clasificados en diferentes categorías, cada uno de estos requerimientos ha sido cuidadosamente analizado y definido para garantizar el funcionamiento optimo del sistema.

**Tabla 13**

*Requerimientos del Sistema*

| **Requerimientos de Sistema RDS** | | | | |
|---|---|---|---|---|
| **#** | **Requerimiento** | **Prioridad** | | |
| | | **Alta** | **Media** | **Baja** |
| **RDS1** | El microcontrolador debe utilizar tecnología 4G LTE para transmitir datos a internet. | | | |
| **RDS2** | La aplicación móvil debe tener conexión a internet mediante WiFi o datos móviles para transmitir y recibir datos. | | | |
| **RDS3** | El sistema debe transmitir y procesar los datos de forma constante durante el proceso de recolección | | | |
| **RDS4** | La aplicación debe tener configurado las funcionalidades requeridas por el cliente. | | | |
| **RDS5** | El sistema debe contar con la restricción para envió de solicitudes de recolección de martes a domingo en el horario de 07:00 a 08:00 am | | | |
| **RDS6** | La aplicación debe tener configurado las funcionalidades requeridas para el conductor del recolector. | | | |
| **RDS7** | El microcontrolador del debe tener una fuente de alimentación entre 3.3 V a 4.2V | | | |
| | **Requerimiento de Performance** | | | |
| **RDS8** | El microcontrolador debe priorizar un envio de datos mediante 4G LTE para asegurar altas velocidades de transmisión de datos. | | | |

| | Requerimientos de modos/estados | | |
|---|---|---|---|
| **RDS9** | El dispositivo debe de prenderse o estar activo cuando el recolector empieza los recorridos de recolección. | | |
| **RDS11** | El dispositivo debe de apagarse cuando el recolector finalice los recorridos de recolección para ahorrar batería | | |

### *3.1.4.4 Requerimientos de Arquitectura*

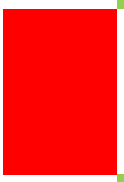
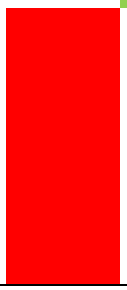
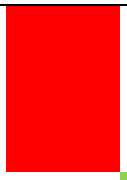
Después de establecer los requerimientos de stakeholders y los requerimientos necesarios para un correcto funcionamiento del sistema, es importante definir los requerimientos de arquitectura.

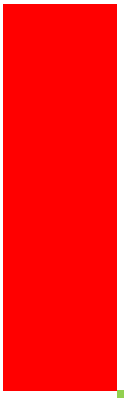
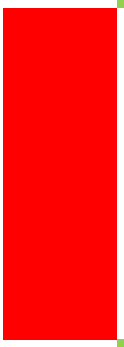
En la Tabla 14 se detallan los requerimientos de arquitectura que se deben considerar para el diseño del sistema. Estos requerimientos abordan aspectos técnicos y de diseño que permitan una implementación robusta y de alta calidad.

**Tabla 14**

*Requerimientos de Arquitectura*

| Requerimientos de Arquitectura RDA | | | | |
|---|---|---|---|---|
| **#** | **Requerimiento** | **Prioridad** | | |
| | | **Alta** | **Media** | **Baja** |
| | **Requerimientos de software** | | | |
| **RDA1** | El lenguaje de programación debe ser compatible con el microcontrolador. | | | |
| **RDA2** | El lenguaje de programación debe ser de código abierto. | | | |
| **RDA3** | El framework de desarrollo de la aplicación móvil debe de ser multiplataforma. | | | |
| **RDA4** | El software debe ser fácil de instalar y usar. | | | |

| | | |
|---|---|---|
| **RDA5** | La base de datos debe de estar alojada en la nube y ser confiable. |  |
| **RDA6** | La base de datos debe permitir visualizar los datos almacenados de forma fácil. |  |
| **RDA7** | La base de datos debe de ser compatible con el framework de desarrollo de la aplicación móvil. |  |
| **RDA8** | Los servicios de mapas deben ser personalizables y alojados en la nube. |  |
| **RDA9** | Debe existir disponibilidad de librerías para la configuración de características. |  |
| **RDA10** | Debe soportar escalabilidad para manejar aumento en el número de usuarios y solicitudes de recolección. | |
| **Requerimientos de diseño** | | |
| **RDA11** | El dispositivo debe de protegerse con una estructura para no estar expuesto posibles daños. |  |
| **RDA12** | El dispositivo debe de ubicarse en un lugar estratégico en el carro recolector. |  |
| **Requerimientos de hardware** | | |
| **RDA13** | El microcontrolador debe tener integrado un módulo de comunicación GPRS/GSM. |  |
| **RDA14** | El microcontrolador debe tener integrado un módulo GPS. | |
| **RDA15** | El microcontrolador debe de ser compatible con tecnología móvil 4G LTE. |  |
| **RDA16** | El microcontrolador debe tener disponibilidad de ranura para SIM de operadora. |  |
| **RDA17** | El microcontrolador debe contar con interfaces para antena GPS. |  |

| | | |
|---|---|---|
| **RDA18** | El microcontrolador debe contar con interfaces para antena 4G LTE |  |
| **RDA19** | Se debe contar con una antena 4G LTE. | |
| **RDA20** | Se debe contar con una antena GPS. | |
| **RDA21** | Se debe contar con una SIM de una operadora telefónica con cobertura 4G LTE en el barrio Centenario. | |
| **RDA22** | La SIM debe contar con datos móviles con un plan de bajo costo. |  |
| **RDA23** | La operadora móvil propietaria de la SIM debe brindar una buena experiencia de usuario. |  |
| **RDA24** | La batería debe tener la capacidad de alimentar al microcontrolador por un tiempo mínimo de 2 horas. | |
| **RDA25** | La batería debe ser recargable con un cargador externo |  |

## 3.2 Selección de Hardware y Software para el Sistema de recolección de RSU

Una vez definidos los requerimientos que guían el desarrollo del sistema de recolección de residuos sólidos urbanos, es fundamental proceder con la elección de hardware y software adecuado para el proyecto.

Esta etapa del proyecto implica un análisis exhaustivo de las opciones disponibles en el mercado, evaluando su compatibilidad con los requerimientos definidos y su capacidad para cumplir con los objetivos establecidos. Una elección acertada de estos elementos garantiza un rendimiento óptimo, escalabilidad, seguridad y mantenimiento en el sistema.

### *3.2.1 Selección de Hardware del Sistema de recolección de RSU*

La selección adecuada del hardware es fundamental para garantizar el correcto funcionamiento del sistema de recolección de residuos sólidos urbanos. Cada componente de

hardware desempeña un papel crucial en la recopilación, procesamiento y transmisión de datos, así como en la alimentación y protección del sistema.

La selección cuidadosa de estos componentes asegura una solución robusta y confiable, capaz de adaptarse a las necesidades futuras del sistema.

#### *3.2.1.1 Placa de procesamiento*

Este componente es responsable de recopilar y procesar los datos del GPS, así como de gestionar la comunicación con la red y la transmisión de los datos del GPS a hacia la base de datos. Por lo tanto, la selección de la placa de procesamiento adecuada es importante para el correcto funcionamiento del sistema.

En la Tabla 15, se detalla el análisis y valoración en base a los requerimientos definidos previamente, de esta forma se realiza una comparación entre las posibles placas que se ajuste con los requerimientos del proyecto.

**Tabla 15**

*Selección de la placa de procesamiento*

| **Selección de placa de procesamiento** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Hardware** | **Requerimientos** | | | | | | | | |
| | **RDST11** | **RDS1** | **RD A13** | **RD A14** | **RD A15** | **RD A16** | **RD A17** | **RD A18** | **Valor total** |
| Arduino Mega | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| ESP32 T-SIM7600 X | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 8 |
| Raspberry Pi 4 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| | | | | | “1” Cumple | | “0” No cumple | | |

Después de un análisis de las diferentes opciones disponibles en el mercado, se determina que la placa de procesamiento ESP32 T-SIM7600X es la alternativa más adecuada

para cumplir con los requerimientos establecidos para la placa de procesamiento del sistema de recolección de RSU. Como se muestra en la Tabla 15, esta placa obtuvo una puntuación de 8/8, superando ampliamente a otras opciones como la Raspberry Pi 4 y el Arduino Mega, que no logran satisfacer los criterios de selección.

Esta placa destaca por su conectividad inalámbrica avanzada, al integrar un módulo GPRS/GSM SIM7600 que permite la conexión a redes móviles LTE, incluyendo ranura para tarjeta SIM física. Además, cuenta con compatibilidad nativa con tecnologías GPS y soporte para baterías, aspectos importantes para el sistema de recolección de RSU(Xinyuan, 2023).

En la Tabla 16 se muestran sus características principales desde una perspectiva técnica en base al datasheet del **Anexo 3,** que hace referencia al módulo ESP32 T-SIM7600NA, la cuál ha sido la placa elegida del conjunto de ESP32 T-SIM7600X, debido a la compatibilidad de soporte para bandas de frecuencia LTE de Ecuador (Xinyuan, 2023).

**Tabla 16**

*Características Técnicas ESP32 T-SIM7600NA*

| Característica | Descripción |
|---|---|
| Procesador | Procesador de doble núcleo Xtensa LX6 de 32 bits, con frecuencia de reloj de hasta 240 MHz, Fash 4MB y PSRAM 8MB |
| Módulo LTE | Módulo SIM7600NA, compatible con redes celulares 4G LTE, con soporte para bandas de frecuencia de Norteamérica(similares a Ecuador) por sus siglas NA. |
| Comunicación | Soporte para comunicación inalámbrica mediante LTE, incluyendo LTE, NB-IoT, Wi-Fi 802.11 b/g/n y Buetooth: v4.2 BR/EDR and BLE |
| Velocidad LTE | Optimización de la velocidad de descarga de 10(DL)/5(UL) a 150(DL)/50(UL) |

| | |
|---|---|
| GNSS | Receptor GNSS incorporado compatible con GPS, GLONASS, BeiDou y Galileo, para posicionamiento global |
| Interfaces | Interfaces UART, SPI, I2C, SDIO, I2S y CAN disponibles para conexión de periféricos |
| Potencia | Rango de voltaje de operación de 3.3V a 5V, compatible con diversas fuentes de alimentación |
| Dimensiones | Dimensiones físicas aproximadas de 30.0*30.0*2.9mm |
| Batería y carga solar | Integración de soporte para batería 18650 y puerto de carga solar. |

Después de haber revisado las características importantes de la placa ESP32 T-SIM7600NA, es importante considerar la elección adecuada de la tarjeta SIM para garantizar una conectividad inalámbrica óptima a internet. En la siguiente sección se realiza la evaluación y elección de la SIM adecuada a partir de los requerimientos definidos previamente.

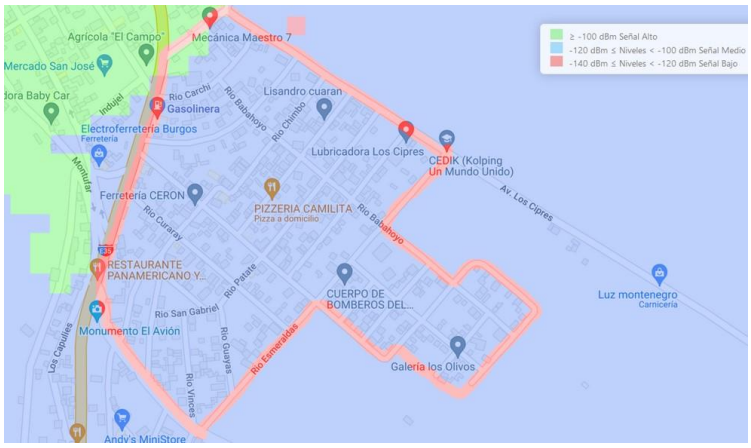
#### *3.2.1.2 Tarjeta SIM*

La tarjeta SIM desempeña un papel fundamental en la comunicación inalámbrica, facilitando la identificación del dispositivo en la red y proporcionando acceso a los servicios de datos móviles. Para el presente proyecto es importante que la tarjeta SIM pueda hacer uso de tecnología 4G LTE con una cobertura estable en el barrio Centenario, para esto se verifica que operadoras prestan el servicio en el sector y de esta forma lograr establecer una comparativa en base a los requerimientos prestablecidos

Las tres operadoras móviles que tienen mayor presencia en el sector son: Claro, Movistar y CNT. La Figura 20 muestra la cobertura 4G LTE de Claro en el sector, donde se verifica que si existe cobertura para esta tecnología móvil en un rango medio entre -120 dBm y -100 dBm.

**Figura 20**

*Mapa de cobertura 4G LTE de la operadora Claro en el barrio Centenario.*




*Nota.* Adaptada de (Geodata, 2024)

A continuación, en la Figura 21 se visualiza la cobertura 4G LTE de la operadora Movistar en el sector, que al igual que la operadora Claro se encuentra en un rango medio entre -120 dBm y -100 dBm, aunque existen mínimos espacios con cobertura en un rango mayor a -100 dBm, lo que indica una mejor cobertura en esas zonas específicas.

**Figura 21**

*Mapa de cobertura 4G LTE de la operadora Movistar en el barrio Centenario.*


*Nota.* Adaptada de (CNT AppGeoportal, 2024)

Finalmente, la Figura 22 muestra la cobertura 4G LTE de la operadora CNT en el sector. A diferencia de Claro y Movistar, se verifica que CNT dispone de una mejor cobertura en el

área. Aproximadamente la mitad de la extensión del barrio cuenta con un rango de cobertura superior a -100 dBm, lo que indica mejores niveles de señal. Por tanto, queda evidenciado que la operadora CNT brinda la cobertura 4G LTE con mayor nivel de potencia en el barrio Centenario.

**Figura 22**

*Mapa de cobertura 4G LTE de la operadora CNT en el barrio Centenario.*


*Nota.* Adaptada de (CNT AppGeoportal, 2024)

Después de analizar la cobertura de las principales operadoras móviles en el sector, se verifica que las tres cumplen con el requisito de brindar cobertura para tecnología 4G LTE. CNT se destaca al ofrecer una cobertura con mayor potencia en el área, lo que puede influir positivamente en la transmisión de los datos a internet. No obstante, Claro y Movistar también presentan rangos estables para conectarse a la red móvil mediante 4G LTE. Por lo tanto, la cobertura no es un factor diferencial que permita decidir entre una u otra operadora.

El reporte estadístico mensual de enero de 2024 de Arcotel (2024) destaca que Claro es la operadora con mayor participación del mercado en el servicio móvil avanzado a nivel nacional, seguida por Movistar y CNT en tercer lugar. Esto demuestra la mayor aceptación de Claro entre los usuarios ecuatorianos.

Respaldando esta información a nivel local, el reporte sobre la experiencia de usuario en redes móviles de (SpeedChecker, 2022) señala que, Claro lidera como la operadora móvil con mayor preferencia de los usuarios en la provincia del Carchi, donde se ubica la ciudad de San Gabriel y el barrio Centenario, área de aplicación de este proyecto. Claro se posiciona como la operadora líder al ofrecer un servicio con menor latencia, registrando 43 milisegundos (ms), por debajo de los 49 ms de CNT y los 106 ms de Movistar.

La Tabla 17 presenta una comparación de las características de costo y experiencia de usuario de las principales operadoras móviles en el barrio Centenario. En cuanto al costo de los planes de datos móviles, se observa que tanto Claro como Movistar ofrecen planes a $10.25, mientras que CNT tiene el plan más económico a $10.00, aunque no es una diferencia significativa. Sin embargo, al evaluar la experiencia de usuario, Claro se destaca como la única operadora con una calificación positiva, en contraste con Movistar y CNT, que presentan una experiencia de usuario negativa según los reportes analizados.

**Tabla 17**

*Comparación de características de costo y experiencia de usuario de las principales operadoras móviles en barrio Centenario.*

| **Operadora** | **Costo de plan de datos móviles(30 días)** | **Experiencia de usuario** |
|---|---|---|
| **Claro** | $10.25 | Positiva |
| **Movistar** | $10.25 | Negativa |
| **CNT** | $10.00 | Negativa |

Una vez analizadas las características de las diferentes operadoras en el sector, se puede evaluarlas de acuerdo con los requerimientos que se alinean para elegir la mejor opción para el

proyecto. En la Tabla 18 se realiza la selección de acuerdo con los criterios de cada requerimiento alineado a la elección de la tarjeta SIM.

**Tabla 18**

*Selección de Tarjeta SIM*

| Selección de Tarjeta SIM | | | | | |
|---|---|---|---|---|---|
| **Operadora** | **Requerimientos** | | | | |
| | **RDST1** | **RDA21** | **RDA22** | **RDA23** | **Valor total** |
| **Claro** | 1 | 1 | 1 | 1 | 4 |
| **Movistar** | 1 | 1 | 1 | 0 | 3 |
| **CNT** | 1 | 1 | 1 | 0 | 3 |

Como se muestra en la Tabla 18, la evaluación de las opciones de tarjeta SIM para el proyecto, destaca a Claro como la elección más sobresaliente al cumplir con todos los requisitos necesarios, obteniendo un puntaje total de 4 sobre 4. Por otro lado, tanto Movistar como CNT alcanzaron un puntaje ligeramente inferior de 3 sobre 4. En consecuencia, basándose en esta evaluación comparativa, Claro es la opción más adecuada y confiable.

#### *3.2.1.3 Batería*

Tras seleccionar la tarjeta SIM de Claro, es fundamental abordar otro componente crítico: la batería. La elección de una batería apropiada es esencial para garantizar un suministro de energía continuo y estable a la placa, asegurando así su correcto funcionamiento y autonomía.

La batería debe ser capaz de proporcionar la corriente y el voltaje necesarios para alimentar el microcontrolador. Para ello, se considera el datasheet del **Anexo 3** correspondiente al microcontrolador ESP32 T-SIM7600, que indica este dispositivo integra soporte para baterías ICR(Lithium-Ion Cylindrical Rechargeable) 18650 (Xinyuan, 2023).

Según la Tabla 19, el consumo de corriente de trabajo del sistema se encuentra alrededor de los 200 mAh, considerando el funcionamiento del módulo 4G y GPS que el microcontrolador integra. Sin embargo, es esencial considerar que factores externos, como la temperatura de operación, podrían incrementar el consumo normal.

**Tabla 19**

*Consumo energético de la placa.*

| Consumo Energético | | | |
|---|---|---|---|
| **Hardware** | **Consumo** | **Voltaje** | **Temperatura** |
| **ESP32 T-SIM7600NA** | 200 mAh | 3.3V – 4.2V | -40 °C + 85°C |

Para satisfacer los requerimientos de consumo energético (RDA24 y RDA25), es crucial seleccionar una batería recargable con una capacidad adecuada que garantice el funcionamiento prolongado y confiable del sistema. Dado que el recolector demora aproximadamente 2 horas en completar la recolección en todo el barrio, se elige una batería con una capacidad mínima para alimentar el sistema durante ese tiempo.

Para calcular la capacidad necesaria de la batería en mAh(miliamperio-hora), se utiliza la Ec. 2. Esta fórmula requiere dos datos: el consumo del microcontrolador y el tiempo durante el cual se requiere que este operativo. Con estos datos, se determina el valor mínimo de la capacidad de la batería requerida para el microcontrolador, como se muestra a continuación:

$$Capacidad\ Bateria\ (mAh) = Consumo\ x\ Tiempo \qquad \textbf{Ec. 2}$$

$$Capacidad\ Bateria\ (mAh) = 200\ mAh\ x\ 2\ horas$$

$$Capacidad\ Bateria\ (mAh) = 400\ mAh$$

Luego de remplazar los datos correspondientes en Ec. 2, se verifica que, es necesaria una batería con una capacidad mínima de 400 mAh para garantizar la alimentación energética del microcontrolador durante las 2 horas requeridas.

Por tanto, se define que la batería para el microcontrolador es una batería ICR 18650 de 3.3V - 4.2V y 400 mAh mínimo de capacidad.

### *3.2.2 Selección de Software para el sistema de recolección de RSU*

Una vez seleccionados los componentes de hardware, es fundamental abordar la elección del software adecuado para el desarrollo del sistema de recolección de residuos sólidos urbanos.

El software desempeña un papel crucial en la programación y configuración de los componentes de hardware, así como en la gestión de la comunicación, el procesamiento de datos y la integración con la plataforma móvil. En este apartado se realiza la comparativa en base a los requerimientos que comprende este sistema.

#### *3.2.2.1 Selección de Software para la programación del microcontrolador*

En esta sección, se lleva a cabo la selección del software más adecuado para programar la placa de procesamiento ESP32-SIM7600NA. Si bien esta placa es compatible con diversos softwares, algunos podrían simplificar ciertas características o la implementación de librerías específicas para su óptimo funcionamiento. Por esta razón, se analizan y evalúan diferentes opciones de software en base a los requerimientos establecidos.

En la Tabla 20, se presenta una comparativa en base a los requerimientos específicos para las opciones de software consideradas para la programación de la placa de procesamiento. Esta evaluación permite seleccionar el software más adecuado para establecer un desarrollo efectivo, que contribuya al logro de los objetivos planteados.

**Tabla 20**

*Selección de software para la programación de la placa de procesamiento*

| Software | RDA1 | RDA2 | RDA4 | RDA9 | Valor total |
|---|---|---|---|---|---|
| **Arduino IDE** | 1 | 1 | 1 | 1 | 4 |
| **Micropython** | 1 | 1 | 0 | 0 | 2 |
| **PlatformIO** | 1 | 1 | 0 | 0 | 2 |

**"1" Cumple "0" No cumple**

El software que se destaca en la Tabla 20 por cumplir con los requisitos necesarios para el desarrollo del sistema es Arduino IDE, obteniendo una puntuación perfecta de 4/4, superando así a los otros dos softwares evaluados que alcanzan una calificación de 2/4 en términos de cumplimiento de los requisitos.

Dado que el proyecto utiliza una placa ESP32, Arduino IDE resulta ideal debido a su compatibilidad con las librerías necesarias para esta plataforma. Además, su lenguaje de programación C++ ofrece una sintaxis más accesible y estructurada en comparación con otros softwares, facilitando la comprensión y escritura del código. La disponibilidad de bibliotecas especializadas para la adquisición de datos del GPS y el envío de datos a través de Internet contribuye aún más a simplificar el proceso de programación con este software.

#### *3.2.2.2 Selección de software para el desarrollo de la aplicación móvil*

Para el desarrollo de aplicaciones móviles, hay una amplia gama de software disponibles, cada uno con sus propias funcionalidades y requisitos de habilidades. La selección del software adecuado depende en gran medida de las necesidades específicas del proyecto.

En la Tabla 21 se presenta la comparativa entre diferentes softwares, junto con la calificación de cumplimiento en relación con los requerimientos asociados a esta selección. Es importante tener en cuenta que algunos softwares son más completos en sí mismos, mientras que otros pueden requerir el uso de herramientas adicionales para lograr ciertas funcionalidades.

**Tabla 21**

*Selección del software de desarrollo de app móvil*

| **Selección del software de desarrollo de app móvil** | | | | | |
|---|---|---|---|---|---|
| **Software** | | | | | |
| | **RDA2** | **RDA3** | **RDA4** | **RDA9** | **Valor total** |
| **Android Studio** | 1 | 0 | 1 | 1 | 3 |
| **Xcode** | 0 | 0 | 0 | 0 | 0 |
| **Ionic Studio** | 0 | 1 | 0 | 1 | 2 |
| **Visual Studio** | 1 | 0 | 1 | 0 | 2 |
| **Flutter** | 1 | 1 | 0 | 1 | 3 |
| **“1” Cumple “0” No cumple** | | | | | |

Después del análisis y calificación de las opciones disponibles, se obtiene que la mejor opción para el proyecto es utilizar Android Studio combinado con Flutter para el desarrollo de la aplicación móvil. Ambos son desarrollados por Google, lo que garantiza una integración fluida y un soporte completo. Además, ambos son gratuitos, lo que permite acceder a una amplia gama de herramientas sin incurrir en costos adicionales.

Android Studio proporciona un entorno de desarrollo robusto con un emulador de teléfonos integrado, permitiendo probar aplicaciones en diversas configuraciones de dispositivos Android. Su potente depurador y herramientas de solución de errores simplifican

la identificación y corrección de problemas durante el desarrollo. Estas características hacen de Android Studio una plataforma ideal para asegurar la compatibilidad y el rendimiento óptimo en una variedad de dispositivos.

Flutter, por su parte con su lenguaje de programación Dart, permite la creación de aplicaciones multiplataforma con una sola base de código, soportando tanto Android como iOS. Esto optimiza el tiempo y los recursos necesarios para el desarrollo, eliminando la necesidad de mantener bases de código separadas para cada plataforma. La integración con Android Studio es sencilla y efectiva, gracias a herramientas específicas que facilitan el desarrollo y la depuración de aplicaciones Flutter, garantizando una excelente experiencia de desarrollo.

### *3.2.3 Selección de Servidores para el sistema de recolección de RSU*

La selección de servidores es un proceso estratégico que requiere un análisis detallado de los requisitos del proyecto y una evaluación minuciosa de las opciones disponibles en el mercado. Al elegir los servidores adecuados, se sientan las bases para un sistema robusto y confiable que pueda manejar de manera eficiente las cargas de trabajo esperadas y proporcionar un rendimiento óptimo en todo momento.

En esta sección, se realiza la selección de dos componentes críticos para la aplicación: la base de datos y la API de mapas. Se elige las opciones más adecuadas para garantizar que nuestra aplicación cumpla de manera óptima con su propósito.

#### *3.2.3.1 Elección de la Base de Datos*

Para seleccionar la base de datos más apropiada, se hará referencia en los requerimientos del sistema especificados en este capítulo. Esto garantiza que se cumplan los estándares necesarios para el óptimo funcionamiento del sistema. La comparación y elección de la base de datos se detalla en la Tabla 22, donde se califican las diversas opciones disponibles.

**Tabla 22**

*Selección de base de datos para el sistema de recolección de RSU*

| Software | Selección de base de datos para el sistema de recolección de RSU | | | | |
|---|---|---|---|---|---|
| | **RDS5** | **RDA6** | **RDA7** | **RDA10** | **Valor total** |
| **MySQL** | 0 | 0 | 1 | 1 | 2 |
| **Azure CosmoDB** | 1 | 1 | 0 | 1 | 3 |
| **Firebase** | 1 | 1 | 1 | 1 | 4 |
| **Oracle** | 1 | 1 | 0 | 1 | 3 |
| | **“1” Cumple** | **“0” No cumple** | | | |

En este análisis, se observa que Firebase sobresale como la opción que mejor cumple los requisitos establecidos. Aunque Azure Cosmos DB y Oracle son alternativas sólidas, su aprendizaje es más complejo en comparación con Firebase. Al utilizar Firebase como base de datos, se respalda por sus herramientas integradas que afianzarán el proyecto, además de que, se integra perfectamente con plataformas previamente definidas como Android Studio y Flutter.

#### *3.2.3.2 Elección de la API de Mapas*

El sistema requiere el uso de mapas para monitorear la ubicación del recolector en tiempo real. Para abordar estas necesidades, se recurre a las APIs de Mapas, que simplifican la integración de soluciones complejas en las aplicaciones móviles. Entre las características clave se destaca la optimización de rutas.

Es importante que la API seleccionada tenga la capacidad de ejecutar las acciones especificadas en los requerimientos y garantice un rendimiento óptimo. En la Tabla 23, se elige

entre las mejores opciones del mercado para el proyecto y posteriormente se integra en el sistema con los demás componentes previamente establecidos.

**Tabla 23**

*Selección de API de Mapas para el sistema de recolección de RSU*

| | **Selección de API de Mapas para el sistema de recolección de RSU** | | | | | |
|---|---|---|---|---|---|---|
| **Software** | | | | | | |
| | **RDS3** | **RDA4** | **RDA8** | **RDA9** | **RD10** | **Valor total** |
| **OpenStreetMap** | 0 | 0 | 1 | 0 | 0 | 1 |
| **Mapbox** | 1 | 1 | 1 | 1 | 1 | 5 |
| **Google Maps** | 1 | 0 | 1 | 1 | 1 | 4 |
| | "1" Cumple "0" No cumple | | | | | |

Como se puede ver en la Tabla 23, la elección final se ha inclinado a favor de Mapbox, ya que cumple con los criterios de selección, obteniendo una puntuación de 5/5. Mapbox es una opción robusta que ofrece una amplia gama de capacidades y un rendimiento superior. Esta plataforma ha sido elegida debido a su capacidad para brindar soluciones de mapeo de alta calidad. Su facilidad de integración con Firebase y Flutter garantiza una implementación sin problemas.

La capacidad de personalización de Mapbox ofrece la flexibilidad necesaria para adaptarse a los requisitos específicos del proyecto, mientras que su escalabilidad garantiza que pueda crecer junto con las necesidades del sistema. La plataforma también ofrece una amplia variedad de mapas, así como datos de tráfico en tiempo real, lo que ayuda a mejorar aún más su utilidad para el seguimiento de vehículos en tiempo real y la planificación de rutas eficientes.

## 3.3 Diseño del sistema de recolección de RSU

Luego de definir todos los componentes esenciales para el sistema, se puede abordar el diseño del sistema, integrando los elementos seleccionados previamente. Este paso ayuda a desarrollar de forma ordenada el diseño del Sistema del sistema en base a los requerimientos establecidos permitiendo cumplir los objetivos planteados.

En primer lugar, se necesita establecer una arquitectura del sistema de recolección de RSU, ya que permite integrar los diferentes componentes, asegurando su compatibilidad y capacidad para trabajar de manera coordinada, brindando así una solución óptima al proceso de recolección de RSU en el barrio Centenario.

### *3.3.1 Arquitectura del sistema de recolección de RSU*

Para obtener una arquitectura robusta y acorde con los estándares de diseño e ingeniería, se ha optado por emplear el modelo de referencia de Internet de las Cosas (IoT, por sus siglas en inglés). Como se muestra en la Figura 23, este modelo permite desglosar y organizar los componentes del sistema en capas, facilitando la comprensión de su funcionamiento y la interacción entre sus distintos elementos (Rueda, 2017), a continuación, se describe cada capa de la arquitectura del sistema de recolección de RSU:

- **Capa de detección de objetos:** En esta capa se encuentra el Nodo Recolector, el cual integra un microcontrolador Arduino ESP32 T-SIM7600NA con antenas GPS y LTE. Este microcontrolador cuenta con un módulo SIM de la operadora Claro, lo que le permite conectarse a Internet a través de la red móvil y enviar datos cada 5 segundos.

- **Capa de red:** Esta capa se encarga de la comunicación del Nodo Recolector con Internet mediante la red móvil LTE. Luego, los datos enviados por el nodo son redirigidos a los servidores correspondientes para su almacenamiento y procesamiento.

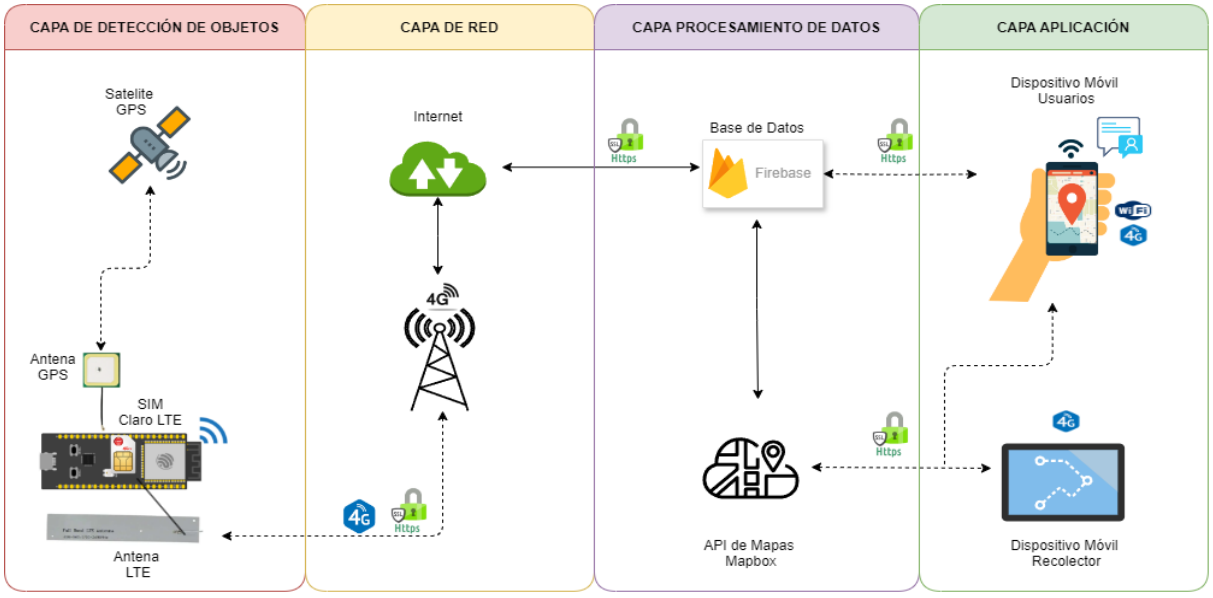
- **Capa de procesamiento de datos:** En esta capa se utilizan los servicios de Firebase(Firebase Realtime Database y Firestore), los cuales se encargan del almacenamiento

y procesamiento de la información proveniente del Nodo Recolector. Además, se emplea la API de Mapbox, que utiliza los datos almacenados en Firebase para generar mapas con ubicaciones y rutas de recolección. Todo esto se logra gracias al desarrollo de la aplicación móvil en Flutter, aprovechando las características y funcionalidades que este framework ofrece, como la integración con servicios en la nube y APIs externas, así como la capacidad de personalizar interfaces de usuario y configurar características adicionales, como notificaciones.

• **Capa de aplicación:** En esta capa se envía, recepta y visualiza los datos a través del protocolo seguro HTTPS[16]. Estos datos pueden ser visualizados en tiempo real en las interfaces de la aplicación móvil del sistema de recolección de RSU.

**Figura 23**

*Modelo de referencia IoT aplicado al Sistema de recolección de RSU en el barrio Centenario*




[16] Protocolo de Transferencia de Hipertexto Seguro

### *3.3.2 Diagrama de bloques general del sistema de recolección de RSU*

El funcionamiento del sistema de recolección de residuos sólidos se representa mediante un diagrama de bloques, que está compuesto por cuatro bloques principales que operan de forma coordinada, como se ilustra en la Figura 24.

El primer bloque, denominado "Bloque de obtención de datos ", se encarga de adquirir datos de localización del vehículo recolector, incorpora el microcontrolador Arduino ESP32 T-SIM7600NA alimentado por la batería ICR 18650; este microcontrolador integra dos módulos, un GPS para determinar la ubicación en tiempo real y un GPRS/GSM para establecer una conexión inalámbrica con la red celular. Mediante las antenas GPS, LTE y una tarjeta SIM de la operadora Claro, el microcontrolador Arduino envía periódicamente (cada 5 segundos) un archivo JSON (JavaScript Object Notation) que contiene los datos de localización GPS al bloque de transmisión, utilizando el protocolo seguro HTTPS.

El segundo bloque, denominado "Bloque de transmisión", actúa como intermediario entre el bloque de obtención de datos y el bloque de almacenamiento/procesamiento. Este bloque comprende la infraestructura de la operadora de telecomunicaciones y los servidores de la red de internet. El microcontrolador instalado en el vehículo recolector, a través del módulo de comunicación con la antena LTE establece una conexión inalámbrica con las estaciones base de la red móvil de la operadora Claro. De esta manera, los datos recolectados son transmitidos de forma segura a través de la red de la operadora hacia Internet, para luego ser redirigidos hacia los servidores del siguiente bloque.

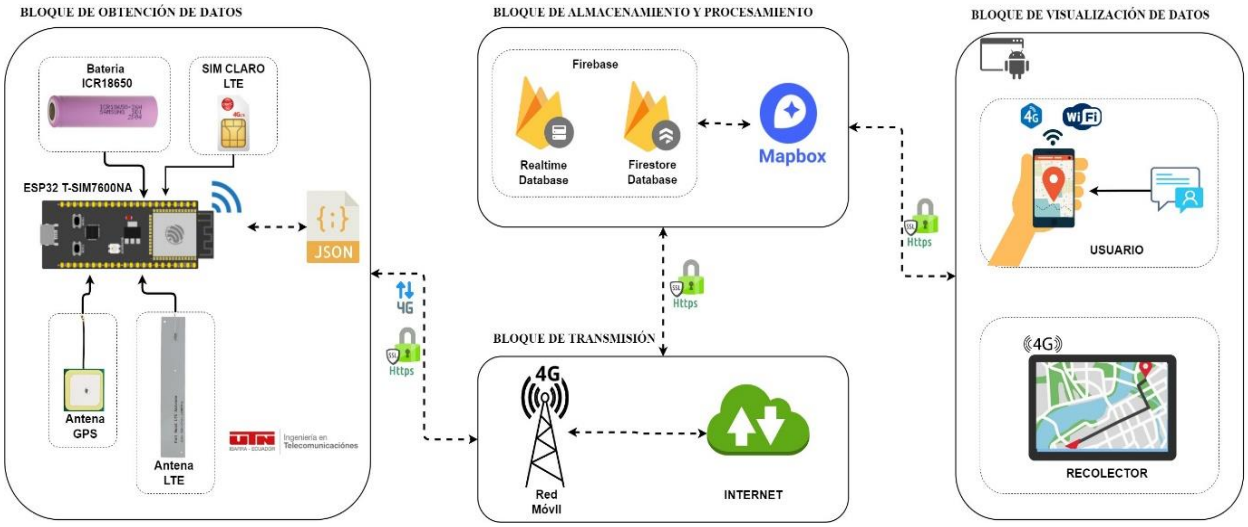
El tercer bloque, denominado "Bloque de almacenamiento y procesamiento", alberga una plataforma en la nube llamada Firebase, donde se encuentran las bases de datos de las cuales se usan dos tipos: Realtime Database para almacenar en tiempo real los datos de localización GPS del vehículo recolector, y Firestore Database para almacenar los datos de las

solicitudes de recolección de residuos de los clientes. Los datos extraídos de estas bases de datos son procesados por la API de Mapbox, para luego ser visualizados en el siguiente bloque.

Finalmente, el cuarto bloque, denominado "Bloque de visualización de datos", comprende una aplicación móvil que permite a los usuarios visualizar los datos recolectados y procesados. Desde la App, los recolectores pueden ver los puntos de recolección y la ruta óptima que pasa por cada uno, mientras que los clientes visualizan la ubicación en tiempo real del vehículo, la distancia y el tiempo estimado de llegada. Además, la aplicación genera notificaciones oportunas para el cliente relacionadas con el proceso de recolección.

**Figura 24**

*Diagrama de bloques del Sistema de recolección de RSU*




### *3.3.3 Diagrama de flujo del sistema de recolección de RSU*

El diagrama de flujo de la Figura 25, representa detalladamente el proceso funcional del sistema de recolección de residuos sólidos, desde la obtención de datos GPS hasta la visualización de rutas óptimas y notificaciones en la aplicación móvil, pasando por el procesamiento de datos en la nube y la integración con diferentes tecnologías y servicios.

El diagrama contempla puntos de control y validación, como verificación de conexiones, elección del tipo de usuario y disponibilidad del botón de solicitud para clientes según un horario predeterminado.

Para iniciar, se enciende el módulo ESP32 T-SIM7600NA conectado a las antenas GPS, LTE y el SIM de la operadora. Una vez encendido, se establece conexión con la red móvil de Claro a través del módulo GPRS/GSM. Si la conexión es exitosa, se habilita el receptor GPS para obtener datos de localización.

Cuando se obtienen los datos GPS, se preparan en formato JSON y se envían a la Realtime Database de Firebase a través de HTTPS.

Paralelamente, un usuario ingresa a la aplicación móvil. Si es cliente, el sistema verifica el horario para habilitar el botón de solicitud de recolección. Si el horario es el correcto, el cliente ingresa su nombre y apellido, y al aceptar, se capturan y envían sus coordenadas GPS a la Firestore Database de Firebase.

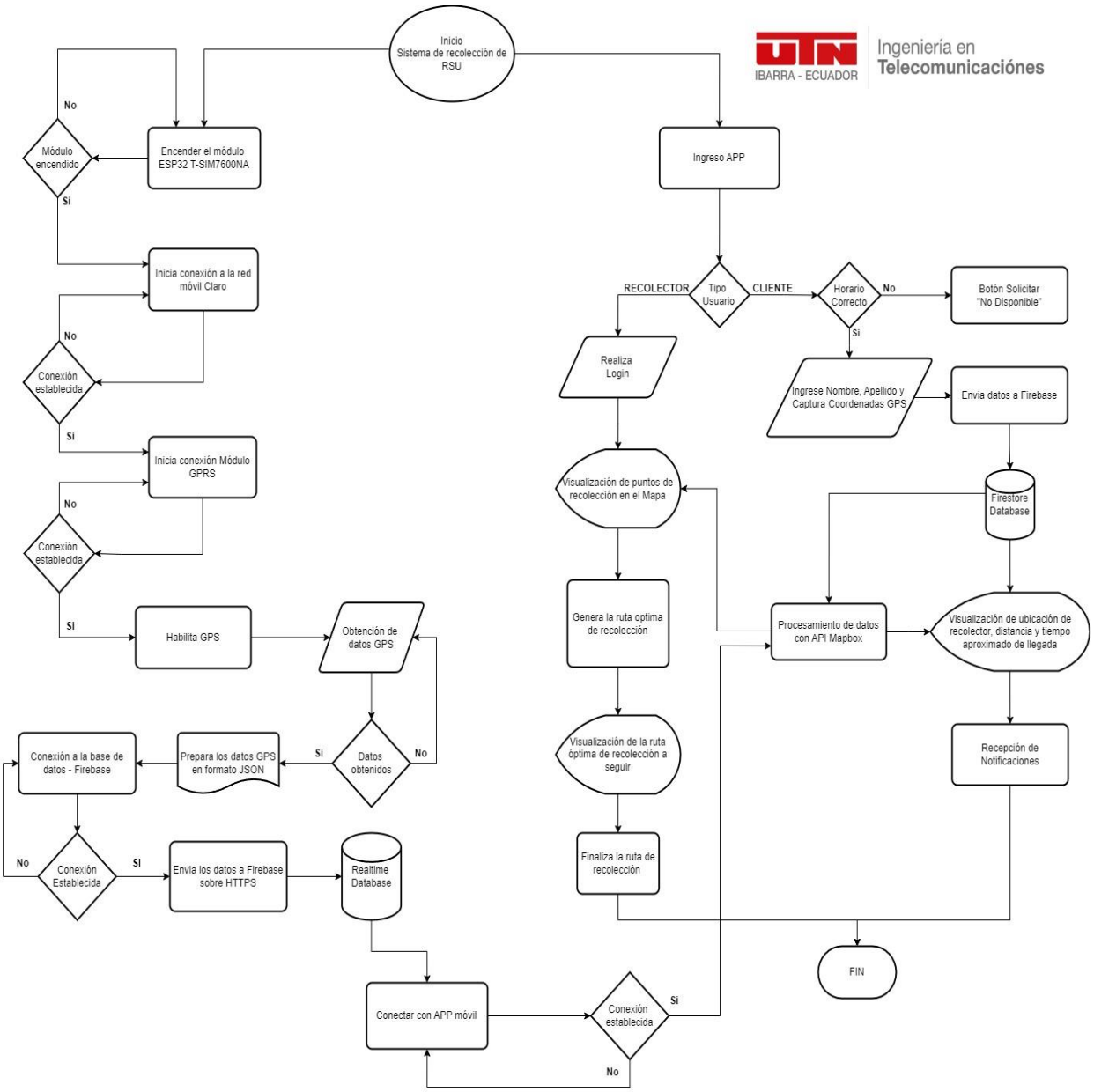
Si es recolector, inicia sesión con las credenciales dadas por el administrador del sistema, una vez realizado el login puede visualizar los puntos de recolección en un mapa y generar la ruta óptima, la cual se muestra en la aplicación para seguirla durante el proceso.

Durante la recolección, los clientes visualizan la ubicación del recolector, la distancia aproximada en metros y el tiempo estimado de llegada en minutos. Además, reciben notificaciones relacionadas con el proceso.

Cuando finaliza la ruta, el sistema completa su ciclo y se detiene hasta el próximo horario establecido.

**Figura 25**

*Diagrama de flujo del Sistema de recolección de RSU*



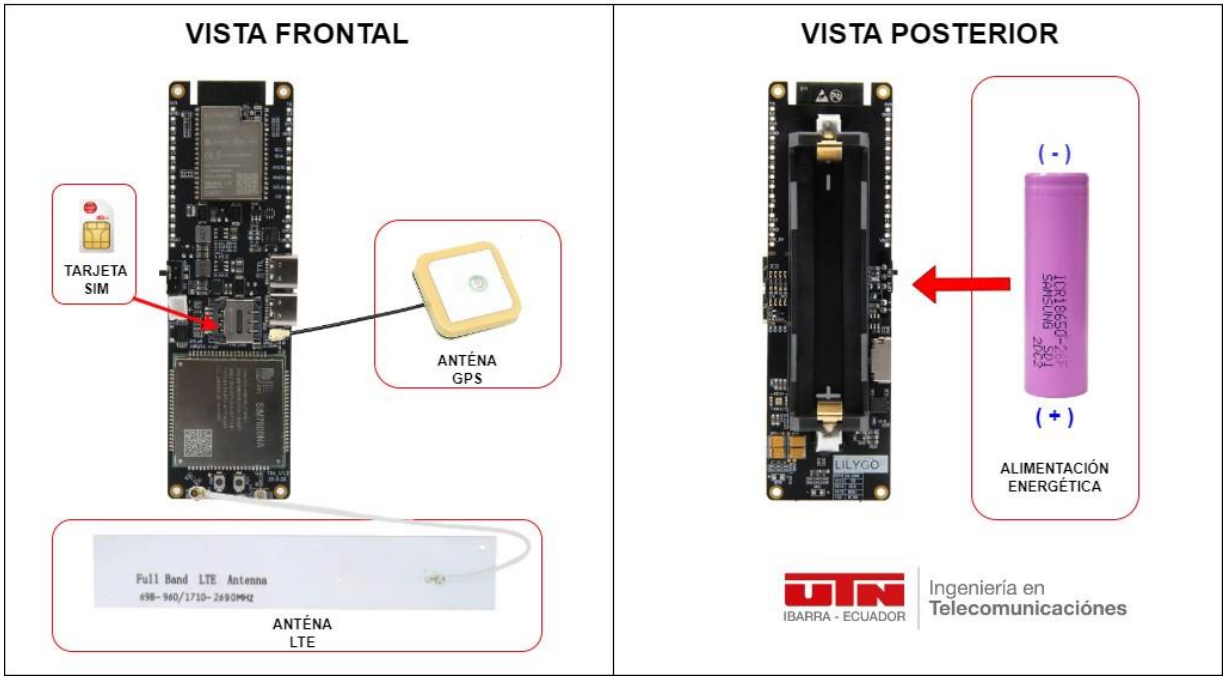

### *3.3.4 Estructura y programación del bloque de obtención de datos del sistema de recolección de RSU*

El bloque de obtención de datos del sistema consta de tres etapas fundamentales: suministro de energía, obtención de datos y transmisión de información.

La Figura 26 ilustra el esquema de conexiones de este bloque, que comprende el ESP32 con el módulo GSM/GPRS T-SIM7600NA, antenas LTE/GPS y tarjeta SIM en la parte frontal, y un compartimento posterior para alojar la batería de suministro de energía.

**Figura 26**

*Esquema de conexiones del nodo recolector*

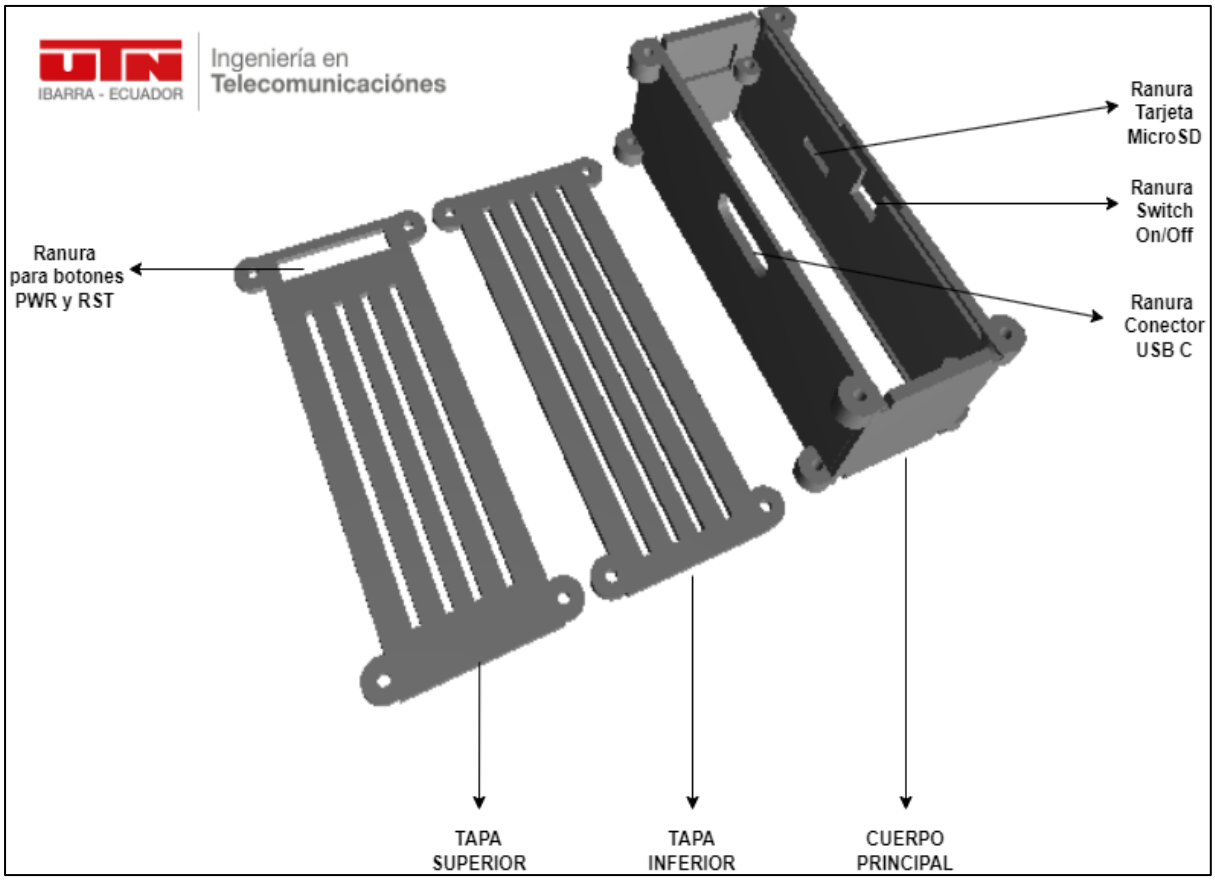



Es importante considerar que el microcontrolador y sus componentes deben estar protegidos con un shield para evitar manipulaciones indebidas. Para esto, se ha diseñado el shield adecuado (ver Figura 27) que consta de un cuerpo principal con ranuras para las conexiones de cable tipo C y el switch para encender o apagar la placa y dos tapas removibles, una superior y una inferior, aseguradas mediante tornillos.

La tapa superior se puede remover para que el administrador del sistema verifique las conexiones de las antenas GPS y LTE en el microcontrolador, mientras que la tapa inferior facilita el cambio de batería.

**Figura 27**

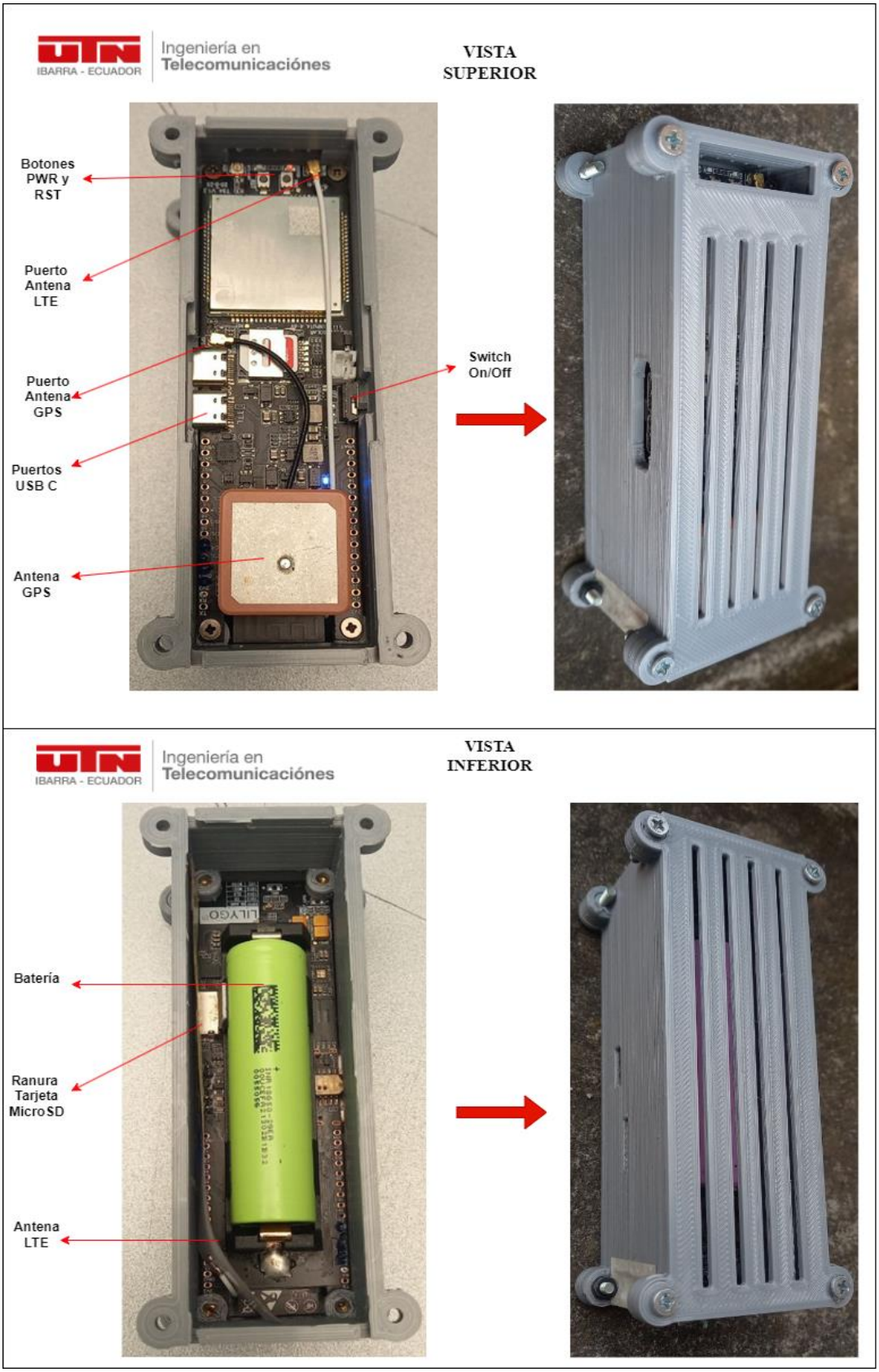
*Diseño de shield para el microcontrolador y sus componentes del nodo recolector del Sistema de recolección de RSU*




En la Figura 28 se ilustra la disposición final de los componentes en el shield tras ser impreso en 3D. Las tapas incluyen las ranuras necesarias para disipar el calor y mantener el sistema en rangos de temperatura de operación óptimos.

**Figura 28**

*Disposición del microcontrolador y sus componentes para el nodo recolector en el shield*



#### *3.3.4.1 Etapa de suministro de energía al nodo recolector*

En la etapa de suministro de energía, se emplea una batería de ion de litio ICR18650 de 3,7V y un mínimo 400 mAh de acuerdo con el datasheet (Anexo 3) del microcontrolador ESP32 T-SIM7600NA y los requerimientos establecidos en este capítulo.

En la Tabla 24, se detallan los valores teóricos sobre el consumo energético del microcontrolador ESP32 T-SIM7600NA en modo de trabajo normal y en modo de máxima potencia. Este último consumo puede darse debido a factores como el funcionamiento a máxima capacidad de los módulos GPS y LTE, o condiciones como la temperatura de operación. Es necesario considerar estos valores para proporcionar la autonomía adecuada al sistema.

**Tabla 24**

*Consumo de energía del microcontrolador ESP32 T-SIM7600NA*

| Hardware | Consumo Normal (mAh) | Consumo Máximo (mAh) |
|---|---|---|
| **ESP32 T-SIM7600NA** | 200 | 600-800 |

Según los parámetros operacionales del sistema, se requiere una batería con una capacidad mínima de 400 mAh. Sin embargo, con esta capacidad, la batería tendría que cargarse diariamente después de las dos horas de activación. Para evitar este proceso de carga diaria, se utiliza una batería de 2600 mAh mínimo. Con esta capacidad, se asegura que la batería dure aproximadamente 6.5 días sin necesidad de recargarse. De esta forma, el administrador del sistema podría recargar la batería cada vez que el sistema complete una semana de trabajo, es decir, cada seis días (de martes a domingo) facilitando la gestión del sistema.

#### *3.3.4.2 Etapa de obtención de datos del nodo recolector*

Para la obtención y transmisión de datos, el nodo recolector integra el microcontrolador ESP32 T-SIM7600NA, este obtiene los datos de GPS y los transmite a la base de datos a través de la red móvil LTE. Este microcontrolador al tener integrado los modulo GPS y GSM/GPRS, es necesario llamar a librerías específicas para la programación de las características requeridas

como se ilustra en la Figura 29, cada una de estas librerías cumple un rol determinado que se detalla en la Tabla 25.

**Figura 29**

*Librerías necesarias para la obtención y transmisión de datos del ESP32 T-SIM7600NA*

```
//Librerias
#include "SSLClient.h"
#include "certs.h"
#include <TinyGsmClient.h>
#include <ArduinoHttpClient.h>
#include <ArduinoJson.h>
```

**Tabla 25**

*Funciones de las librerías del código de programación del ESP32 T-SIM7600NA*

| Librería | Función |
|---|---|
| **SSLClient.h** | Proporciona la funcionalidad para establecer conexiones seguras mediante SSL/TLS[17]. |
| **certs.h** | Contiene los certificados SSL necesarios para la comunicación segura. |
| **TinyGSMClient.h** | Permite la comunicación con el módulo GSM para el envío y recepción de datos. |
| **ArduinoHttpClient.h** | Facilita el envío de solicitudes HTTP[18] y la recepción de respuestas desde un servidor. |
| **ArduinoJson.h** | Ofrece herramientas para crear, enviar y procesar datos JSON de manera sencilla y eficiente. |

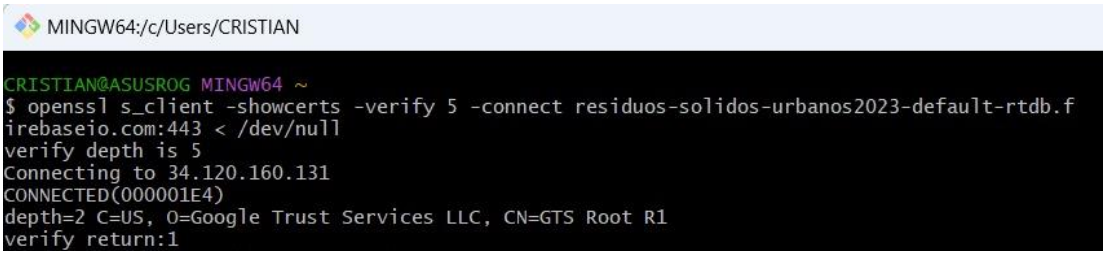
Como se detalla en la Tabla 25, se hace uso de la librería SSLClient.h debido a que, es necesario establecer una conexión segura con los servidores de Firebase para almacenar los datos, además, se hace uso del recurso adicional certs.h, en el cual se contiene una cadena de caracteres que representa el certificado Root CA[19] para la autenticación con Firebase.

[17] Secure Sockets Layer/Transport Layer Security
[18] Hypertext Transfer Protocol
[19] Certificate Authority

Para obtener el certificado Root CA necesario para establecer una conexión segura con los servidores de Firebase, se puede utilizar una consola de comandos y digitar el comando que se muestra en la Figura 30. Este comando se utiliza para conectarse a un servidor SSL/TLS (en este caso, el servidor de Firebase del proyecto) y obtener los certificados que se envían durante el proceso de negociación SSL/TLS.

**Figura 30**

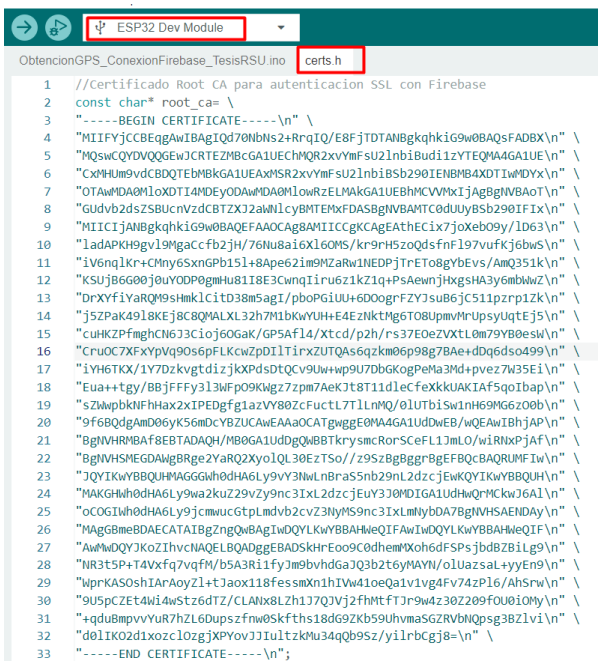
*Comando para obtener el certificado Root para autenticación con Firebase*




Después de obtener el certificado Root CA mediante el comando indicado anteriormente, es necesario agregar este certificado a la pestaña certs.h, como se ilustra en la Figura 31.

**Figura 31**

*Certificado Root para conexión segura con los servidores de Firebase*



Una vez importadas las librerías necesarias y añadido el certificado Root CA para el establecimiento de la conexión segura, se estructura la programación para que el microcontrolador obtenga y transmita los datos GPS. Inicialmente, se determina los pines importantes y la velocidad de comunicación serial, como se muestra en la Figura 32.

**Figura 32**

*Definiciones de pines y la velocidad de la comunicación serial para el microcontrolador ESP32 T-SIM7600NA*

```
// Definiciones de pines y la velocidad de la comunicación serial
#define UART_BAUD           115200          //Velocidad de baudios para el puerto serial
#define MODEM_TX            27              //Pin TX del módulo GSM
#define MODEM_RX            26              //Pin RX del módulo GSM
#define MODEM_PWRKEY        4               //Pin para encender/apagar el módulo GSM
#define MODEM_FLIGHT        25              //Pin de control de modo vuelo del módulo GSM
#define LED_PIN             12              //Pin para el LED indicador
```

De igual forma, se deben definir las configuraciones del módulo GSM, como se ilustra en la Figura 33, aquí se especifica la versión del módem que se está utilizando y los puertos seriales para la comunicación con la consola de monitoreo y la salida de depuración de los mensajes de la librería TinyGSM.

**Figura 33**

*Definición de la configuración del modem y puertos seriales importantes para el microcontrolador ESP30 T-SIM7600NA*

```
#define TINY_GSM_MODEM_SIM7600          //Define el módulo GSM utilizado como SIM7600
#define SerialMon Serial                //Define el puerto serial para la comunicación con la consola de monitoreo
#define SerialAT Serial1                //Define el puerto serial para la comunicación con el módulo GSM
#define TINY_GSM_DEBUG SerialMon        //Define el puerto serial para la salida de depuración de TinyGSM
```

De manera similar, se definen y habilita los Tests correspondientes para que el microcontrolador realice las funciones requeridas. Como se muestra en la Figura 34, los Tests habilitados corresponden a: `GPRS` para establecer la comunicación con la red móvil, `GPS` para obtener los datos de ubicación, `TCP` para establecer la comunicación segura, `Temperatura`

y `Time` para obtener parámetros importantes que permiten el monitoreo de la operación del microcontrolador.

**Figura 34**

*Definición de las pruebas necesarias para la operación adecuada del microcontrolador del nodo recolector*

```
//Definiciones de las pruebas habilitadas de conectividad con TinyGSM
#define TINY_GSM_TEST_GPRS          true
#define TINY_GSM_TEST_GPS           true
#define TINY_GSM_TEST_TCP           true
#define TINY_GSM_TEST_TEMPERATURE   true
#define TINY_GSM_TEST_TIME          true
```

Luego de definir las configuraciones para la operación del microcontrolador del nodo recolector, es necesario especificar los parámetros para establecer la conexión con la red móvil predeterminada. En este caso, se utiliza la red de la operadora Claro. Por lo tanto, se definen las credenciales GPRS, como se muestra en la Figura 35, se declaran tres variables constantes de tipo arreglo de caracteres (`const char[]`): `apn, gprsUser, y gprsPass`. La variable `apn` almacena el nombre del punto de acceso `internet.claro.com.ec`, mientras que `gprsUser` y `gprsPass` contienen el nombre de usuario y la contraseña, ambos con el valor `claro`. Los corchetes vacíos `[]` en las declaraciones permiten que el compilador determine automáticamente el tamaño de los arreglos basado en las cadenas asignadas.

**Figura 35**

*Definición de credenciales GPRS para el establecimiento de conexión con la red móvil Claro Ecuador*

```
// Credenciales GPRS para la conexion con la red movil Claro
const char apn[] = "internet.claro.com.ec";
const char gprsUser[] = "claro";
const char gprsPass[] = "claro";
```

Asimismo, para establecer la conexión con los servidores de la base de datos de Firebase, es necesario definir varios parámetros, como se muestra en la Figura 36, la primera

línea define server como un arreglo de caracteres constante `const char[]` que almacena la URL del servidor Firebase. La segunda línea declara `resource` como una cadena de texto constante `const String` que contiene el token de autenticación para acceder a la base de datos. La tercera línea establece `port` como un entero constante `const int` con el valor `443`, que es el puerto estándar para conexiones HTTPS. La última línea define `var` como una cadena de texto constante `const String` con el valor `/`, representando el directorio raíz en la base de datos. Es importante destacar que en la figura, los valores de `server` y `resource` están ocultos por razones de seguridad, ya que son datos sensibles y únicos del proyecto.

**Figura 36**

*Credenciales para el establecimiento de conexión con los servidores de BDD de Firebase*

```
// Credenciales para la conexion con la BDD FIREBASE
const char server[]  = "[redacted]-rtdb.firebaseio.com";
const String resource  = "[redacted]";
const int port = 443;
const String var  = "/";
```

Una vez definido los parámetros necesarios para el establecimiento de conexión con Firebase y posterior envío y recepción de datos, se presentan las secciones clave del algoritmo de operación del microcontrolador ESP32 T-SIM7600NA del nodo recolector. Para una revisión completa, se recomienda consultar el **Anexo 4**, que contiene el código implementado con comentarios detallados.

Por tanto, como se puede visualizar en la Figura 37, al inicializar el módem, se verifica si el proceso de inicialización fue exitoso.

Luego, se configura el modo de red mediante la función `setNetworkMode()`. Esta función permite seleccionar el tipo de conexión a la red móvil a utilizar. En este caso, se establece el modo `38`, que corresponde a `LTE Only`, lo que significa que el módem se conectará únicamente a redes LTE. Consecutivamente, se programa un mensaje con un valor de retorno para verificar que la conexión por LTE fue exitoso (1) o error (0).

**Figura 37**

*Programación de inicialización de modem y selección de modo de red LTE para la conexión*

```
//Inicio de Modem
if (!modem.init()) {
    DBG("Inicializando Modem.........");
    delay(5000);
  }else{
    DBG("Modem inicializado exitosamente");
  }

#if TINY_GSM_TEST_GPRS
  /*  Modo de seleccion de conexion a red:
      2 - Automatico
      13 - GSM Only
      14 - WCDMA Only
      38 - LTE Only
      19 - GSM+WCDMA Only
      48 - Any but LTE
      39 - GSM+WCDMA+LTE Only
      51 - GSM+LTE Only
      54 - WCDMA+LTE Only
 */
    ret = modem.setNetworkMode(38);
    DBG("Modo de Red:", ret);
```

Al igual que en el apartado anterior, es necesario definir la configuración del modo de GNSS (Sistema Global de Navegación por Satélite), como se muestra en la Figura 38, aquí se obtiene y muestra el modo de GNSS actual utilizando la función `getGNSSMode()`.

Luego, se establece el modo de GNSS deseado a través de la `función setGNSSMode(0, 0).` Se selecciona el modo GLONASS (`valor 0`) debido a que, es la mejor alternativa ante GPS por su cobertura global y precisión. Además, se deshabilita el modo de ahorro de energía (`valor 0`) para obtener datos de ubicación más precisos y confiables, ya que este modo puede afectar la precisión del posicionamiento.

**Figura 38**

*Configuración del modo GNSS para la obtención de datos GPS*

```
uint8_t mode = modem.getGNSSMode();
DBG("Modo de GNSS :", mode);

/**
CGNSSMODE: <gnss_mode>,<dpo_mode>
Configuracion del modo de GNSS
gnss_mode:
    0 : GLONASS
    1 : BEIDOU
    2 : GALILEO
    3 : QZSS
dpo_mode (ahorro de energia) :
    0 deshabilitado
    1 habilitado
*/
modem.setGNSSMode(0, 0);
```

Una vez configurados los modos de red y GNSS, se establece la conexión con la red móvil (ver Figura 39). Se utilizan condicionales `if` para verificar si el módem se conectó exitosamente. El método `!modem.waitForNetwork(600000L)` espera hasta 600,000 milisegundos (10 minutos) para completar la conexión. Si no se logra la conexión en este tiempo, se reintenta después de 5 segundos. En caso de éxito, el método `modem.isNetworkConnected()` verifica y muestra un mensaje confirmando la conexión a la red móvil.

**Figura 39**

*Configuración para establecer la conexión a la red móvil.*

```
//Conexion a la red movil
DBG("Esperando la conexión a la red movil..");
if (!modem.waitForNetwork(600000L)) {
DBG("Red movil no disponible, volviendo a intentar en 5s");
delay(5000);
return;
}

//Verificacion de que la red movil este conectada
if (modem.isNetworkConnected()) {
DBG("Red movil conectada");
}
```

Luego, mediante la salida del método `modem.isGprsConnected()` se verifica si la conexión GPRS está establecida correctamente. Como se muestra en la Figura 40, se obtiene y guarda el resultado del método en la variable `res` y se muestra el estado de la conexión GPRS (conectado o no conectado).

Además, mediante los diferentes métodos se recopila y se muestra información relevante del módem y la tarjeta SIM, como el número CCID(Integrated Circuit Card Indentifier), IMEI(International Mobile Equipment Identity), IMSI(International Mobile Subscriber Identity), el nombre del operador de red, la dirección IP local asignada y la potencia de la señal de la red móvil en dBm. Aunque el módulo permite obtener la calidad de la señal en csq(Signal Quality), se ha optado por transformarlo a dBm, una medida más utilizada en la medición de potencia de redes inalámbricas.

Inicialmente, los módulos GSM/GPRS/UMTS/LTE para obtener la calidad de señal(CSQ) utilizan el comando `AT+CSQ`, esto devuelve un valor entre 0-31 que representa la calidad de la señal, aunque existe el caso particular de que se devuelve el valor de 99 que indica que la señal no es conocida o no detectable.

Para convertir los valores de CSQ a dBm se utiliza la Ec. 3, de esta forma se proporciona una medida estándar de la intensidad de la señal de la red móvil percibida por el módulo. En la Ec. 3, la variable N corresponde al valor de CSQ devuelto por el comando antes mencionado.

$$dBm = -113 + N * 2 \qquad \textbf{Ec. 3}$$

La obtención correcta de estos datos, permite verificar que todo esté configurado y funcionando adecuadamente.

**Figura 40**

*Verificación de Conexión GPRS y Recopilación de Información del Módem y la Tarjeta SIM*

```
// Verificar si la conexión GPRS está establecida
res = modem.isGprsConnected();
DBG("GPRS estado:", res ? "Conectado" : "No Conectado");
DBG("Conexion establecida con", apn);

//Obtener información de la tarjeta SIM
String ccid = modem.getSimCCID();
DBG("CCID:", ccid);
String imei = modem.getIMEI();
DBG("IMEI:", imei);
String imsi = modem.getIMSI();
DBG("IMSI:", imsi);
String cop = modem.getOperator();
DBG("Operador:", cop);
IPAddress local = modem.localIP();
//Obtener la dirección IP local
DBG("Direccion IP Local asignada:", local);
//Obtener la calidad de la señal
int csq = modem.getSignalQuality();
int signalStrength;
if (csq == 99) {
  signalStrength = -999; // Señal no conocida o no detectable
} else {
  signalStrength = -113 + 2 * csq;
}
DBG("Potencia de la señal de la red móvil:", signalStrength, " dBm");
```

Tras la verificación de la conexión correcta, se procede a extraer los datos del GPS, como se muestra en la Figura 41. Primeramente, se habilita el modem GPS mediante el método `modem.enableGPS()` y espera 2 segundos para su inicialización adecuada.

Inmediatamente, se inicializan las diferentes variables para almacenar los datos de GPS, se definen las variables de diferente tipo como `float` para obtener decimales, asi como también de tipo `int` para obtener valores enteros; por tanto, las variables que se obtienen son: latitud, longitud, velocidad, altitud, precisión y los componentes de la fecha y hora, y en un bucle indefinido `for(;;)` con el método `modem.getGPS()` se intenta obtener los datos del

GPS. Se acompaña de una función `digitalWrite,` que permite alternar el modo encendido/apagado del led del microcontrolador, de esta forma se establece un indicador visual para identificar cada vez que se estén obteniendo datos GPS dentro del bucle.

**Figura 41**

*Habilitación de GPS, definición de variables GPS y obtención de datos GPS.*

```
modem.enableGPS();
delay(2000);
// Se inicializan las variables para almacenar la latitud, longitud, velocidad,
float lat2      = 0;
float lon2      = 0;
float speed2    = 0;
float speed_kmh = 0;
float alt2      = 0;
int   vsat2     = 0;
int   usat2     = 0;
float accuracy2 = 0;
int   year2     = 0;
int   month2    = 0;
int   day2      = 0;
int   hour2     = 0;
int   min2      = 0;
int   sec2      = 0;

DBG("Obteniendo coordenadas GPS.....");
for (;;) {                                                    //Bucle infini
  digitalWrite(LED_PIN, !digitalRead(LED_PIN));               //Alternar el

    if (modem.getGPS(&lat2, &lon2, &speed2, &alt2, &vsat2, &usat2, &accuracy2,
                     &year2, &month2, &day2, &hour2, &min2, &sec2)) {      //Se
```

Si los datos del GPS se obtienen con éxito, estos se imprimirán en el monitor serial. Por el contrario, si el proceso falla, se reintentará automáticamente hasta obtener los datos.

### *3.3.4.3 Etapa de transmisión de los datos del nodo recolector*

En esta etapa, se implementa la lógica de programación para la transmisión de la información obtenida en el proceso previo, por tanto, se utiliza el módulo GSM/GPRS del microcontrolador, la tarjeta SIM de la operadora Claro y la antena LTE conectada a su respectivo puerto. Estos componentes permiten la transmisión de los datos recolectados a través de la red móvil hacia los servidores de almacenamiento y procesamiento.

Como se muestra en la Figura 43, una vez obtenidos los datos del GPS, estos son convertidos al formato JSON, el cual es utilizado por el servidor de Firebase para recibir y enviar datos. Para esto, se hace uso de las funciones de la librería `ArduinoJSON`, previamente definida. Inicialmente, se define el tamaño del búfer para el objeto JSON mediante la variable `StaticJsonDocument<256> jsonBuffer`, que determina un tamaño de 256 bits, y crea el objeto JSON. Luego, se asignan los valores de cada variable que integran el documento JSON. Finalmente, el documento JSON se serializa mediante `serializeJson(buffer, data)` `para convertirse` una cadena de texto para enviarlo en una solicitud HTTP; así tambien se construye la URL con las variables previamente definidas (`var` y `resource`) y se especifica el tipo de contenido `application/json.`

**Figura 42**

*Preparación y conversión de datos GPS obtenidos a formato JSON para envío a Firebase*

```
StaticJsonDocument<256> jsonBuffer; //Se define el tamaño del búfer para almacenar el objeto JSON
// Se asignan los valores de las variables de coordenadas y tiempo al objeto JSON
jsonBuffer["lat"] = lat2;
jsonBuffer["lng"] = lon2;
jsonBuffer["speed"] = speed_kmh;
jsonBuffer["altura"] = alt2;
jsonBuffer["precicion"] = accuracy2;
jsonBuffer["fecha"] = String(day3) + "/" + String(month3) + "/" + String(year3);
jsonBuffer["hora"] = String(hour3) + ":" + String(min3) + ":" + String(sec3);

//Convertir el objeto JSON a una cadena de texto
String postData;
serializeJson(jsonBuffer, postData);      //Se serializa el objeto JSON en una cadena de texto para poder

//Se construye la URL para la solicitud HTTP, incluyendo la variable "var" y el recurso de autenticación.
  String URL = var + ".json?auth=" + resource;
//Se define el tipo de contenido de la solicitud HTTP como "application/json".
  String contentType = "application/json";
```

Una vez preparados los datos GPS en formato JSON, se envía los datos al servidor de Firebase usando el método `client.put()`, que contiene la `URL` construida previamente, el tipo de contenido `application/json` y los datos (`postData`) del objeto JSON como se

muestra en la Figura 44. Posteriormente, se obtiene el código de respuesta de la solicitud HTTP mediante el método `client.responsesStatusCode()` y se guarda el valor de respuesta en la variable entera `int status_code`, asimismo, mediante el método `client.responseBody()` se extrae la estructura de los datos enviados y se guarda en la variable de texto `String response` para luego ser impresa en la consola de monitoreo, esto permite verificar si la transmisión de datos fue exitosa. Para terminar, se cierra la conexión con el servidor mediante el método `client.stop()`.

**Figura 43**

*Envío de datos GPS a Firebase y obtención de código de respuesta HTTP*

```
//Se envía una solicitud HTTP PUT al servidor Firebase con la URL, el tipo de contenido y los datos
client.put(URL, contentType, postData);

//Se obtiene el código de estado de la respuesta HTTP y el cuerpo de la respuesta.
int status_code = client.responseStatusCode();
String response = client.responseBody();

//Se muestran en el monitor serial el código de estado y la respuesta recibida del servidor.
DBG("Código de Estado: ", status_code);

// Verificar si el código de estado es diferente de 200
if (status_code == 200) {
  DBG("Datos enviados a Firebase: ", response);
  // Se cierra la conexión con el servidor Firebase.
  client.stop();
  break;  // Salir del bucle si el envío fue exitoso
```

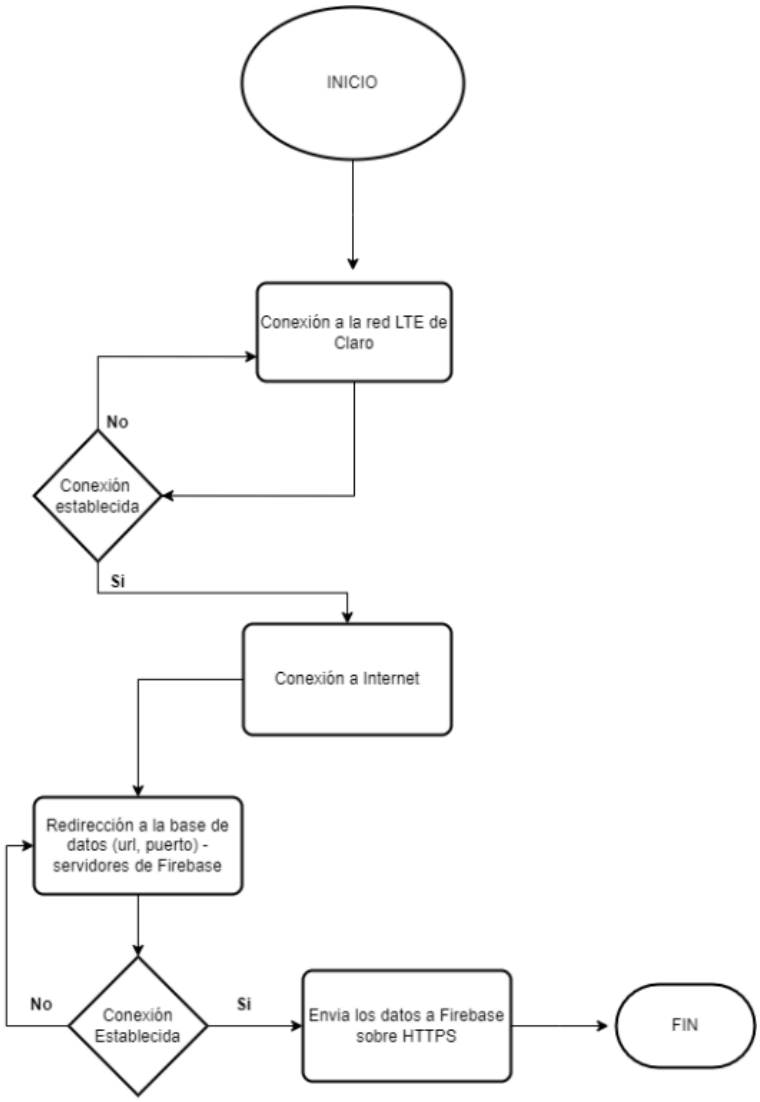
### *3.3.5 Estructura del bloque de transmisión del sistema de recolección*

El bloque de transmisión del sistema de recolección de RSU se encarga de enviar los datos recolectados desde el nodo recolector hacia Internet, utilizando la infraestructura 4G LTE proporcionada por la operadora móvil Claro. En la Figura 45 se muestra el diagrama de flujo del funcionamiento del bloque de transmisión. En este diagrama, se puede observar que inicialmente se establece la conexión a la red LTE de Claro, luego se realiza la conexión a Internet y posteriormente se redirigen los datos a la base de datos en los servidores de Firebase mediante una URL y puerto específico. Una vez establecida la conexión, los datos se envían a

Firebase utilizando el protocolo seguro HTTPS. Este ciclo representa el funcionamiento básico de este bloque.

**Figura 44**

*Diagrama de flujo del funcionamiento básico del bloque de transmisión del Sistema de recolección de RSU*






En esta etapa, no se requiere configuraciones adicionales, ya que la infraestructura 4G LTE es proporcionada por la operadora móvil y el acceso a Internet se realiza a través de los

servidores existentes en la red mundial. Posteriormente, utilizando la URL y el puerto correspondiente, los datos son redirigidos a los servidores de Firebase, lo que proporciona la conectividad necesaria para la transmisión de datos.

### *3.3.6 Estructura y configuración del bloque de almacenamiento y procesamiento de datos del sistema de recolección de RSU*

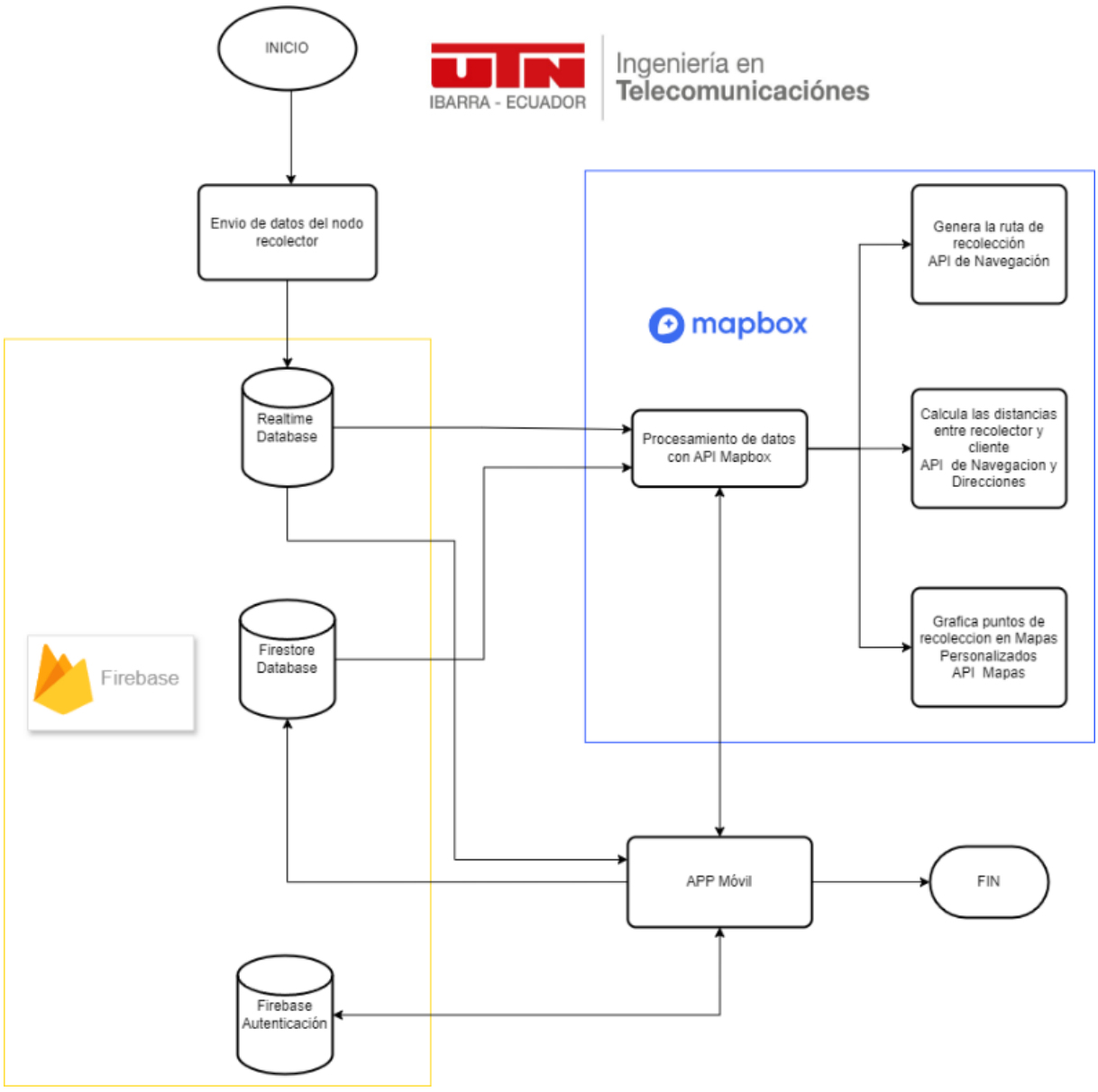
El bloque de almacenamiento y procesamiento de datos del sistema de recolección de RSU consta de varios componentes fundamentales que permiten almacenar y procesar los datos, tanto del nodo recolector como de la aplicación móvil, la cual forma parte del bloque de visualización de datos.

En la Figura 46, se visualiza el diagrama de flujo que permite comprender la operación del bloque de almacenamiento y procesamiento. En el diagrama se puede apreciar que, una vez enviado los datos del nodo recolector, estos llegan al servidor de Firebase, específicamente a Realtime Database; estos se receptan cada 5 segundos para luego ser extraídos por la aplicación móvil y a la API de Mapbox para su procesamiento. Además, se utiliza Firestore Database para guardar los datos de las solicitudes de recolección de los clientes, tales como el nombre/apellido y la ubicación. También se hace uso del servicio de Autenticación de Firebase para gestionar el inicio de sesión del usuario recolector en la aplicación móvil.

Por otro lado, la API de Mapbox a través de solicitudes HTTP a sus APIs integradas (Navegación, Direcciones y Mapas), y utilizando los datos de Realtime Database y Firestore Database, genera rutas de recolección, calcula distancias entre el recolector-cliente y grafica puntos de recolección en mapas personalizados. Estos datos procesados se visualizan en la aplicación móvil, correspondiente al bloque de visualización de datos.

**Figura 45**

*Diagrama de flujo de operación del bloque de almacenamiento y procesamiento del sistema de recolección de RSU*




Después de analizar la estructura y operación general del bloque de almacenamiento y procesamiento, se detalla la configuración de cada uno de sus componentes.

- **Servicio de Autenticación de Firebase**

Al comienzo, es indispensable crear una cuenta en el sitio oficial de Firebase (**https://firebase.google.com/**) y un proyecto específico para obtener acceso a las diferentes bases de datos y servicios ofrecidos. Para este fin, se puede seguir la guía que se encuentra en la página oficial de Firebase.

Al tener una cuenta y proyecto activo, se configura el servicio de autenticación, este es fundamental para asegurar la seguridad y el acceso a funcionalidades destinadas al usuario recolector en la aplicación móvil. Como se muestra en la Figura 47, se configura el método de verificación mediante correo electrónico (identificador) y contraseña, estos son creados manualmente por el administrador en la base de datos. De esta forma, no es posible que otros usuarios generen nuevas credenciales.

**Figura 46**

*Configuración de servicio de Autenticación de Firebase para los usuarios recolectores del Sistema de recolección de RSU.*


- **Realtime Database**

Para el servicio de Realtime Firebase, es importante configurar adecuadamente las reglas de acceso a la base de datos, tal como se muestra en la Figura 48. Se recomienda establecer las reglas de lectura y escritura en `true`. Al habilitar la lectura y escritura, se

garantiza que la base de datos pueda recibir y enviar datos sin restricciones. Esto es fundamental para el correcto funcionamiento del sistema, ya que el nodo recolector necesita enviar datos constantemente, mientras que la aplicación móvil y la API de Mapbox acceden a los datos de GPS almacenados.

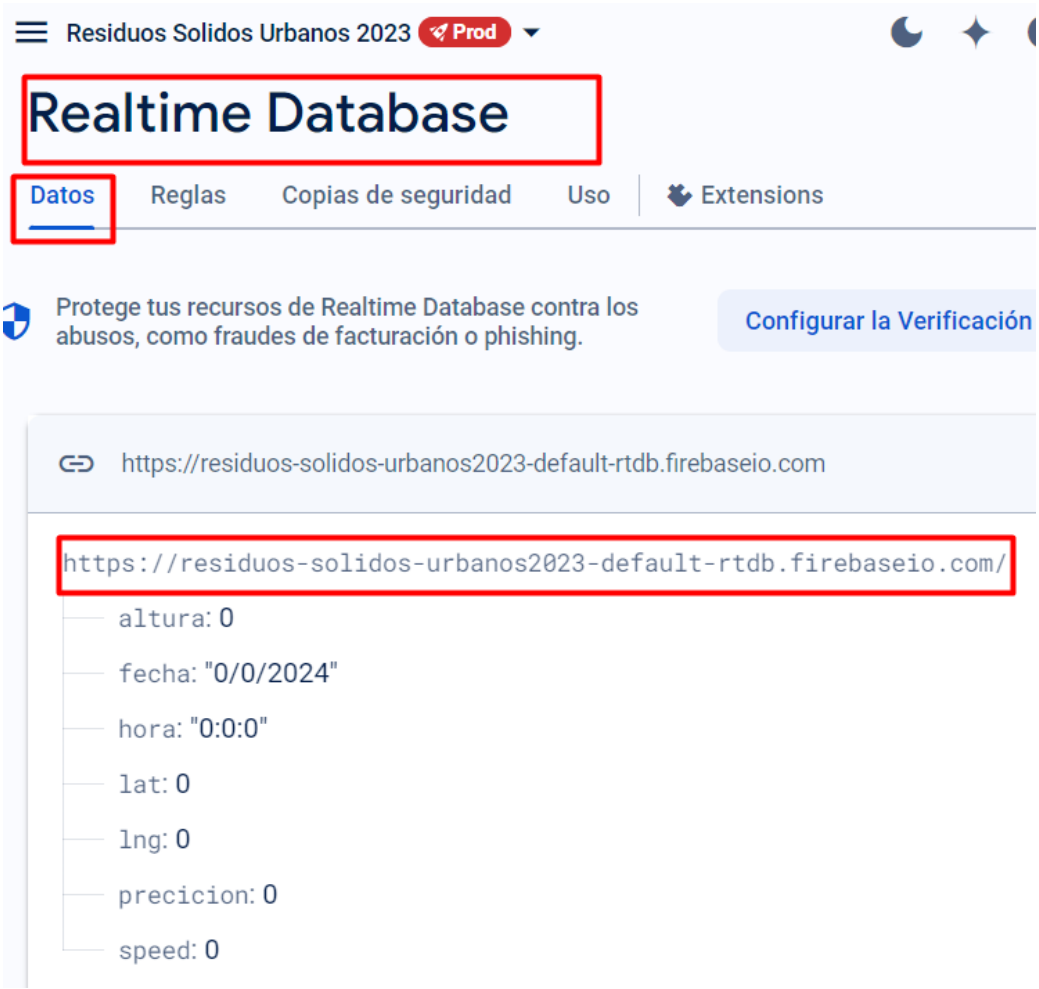
**Figura 47**

*Configuración de reglas de Realtime Database para almacenar los datos del nodo recolector del sistema de recolección de RSU*


La Figura 49 muestra la estructura de almacenamiento de los datos en Realtime Database. Cada dato se representa mediante la variable trasmitida desde el nodo recolector, tal como se describió anteriormente. De esta manera, la información se encuentra organiza y accesible para su visualización y posterior procesamiento.

**Figura 48**

*Visualización de la estructura de datos del GPS del nodo recolector en Realtime Database*




- **Cloud Firestore Database**

Al igual que la base de datos anterior, Cloud Firestore Database se rige por sus propias reglas, como se muestra en la Figura 50, estas reglas permiten controlar la lectura y escritura, para que los datos de los clientes puedan ser almacenados y posteriormente extraídos para procesos específicos del sistema, por tanto, las reglas deben estar definidas en `true`.

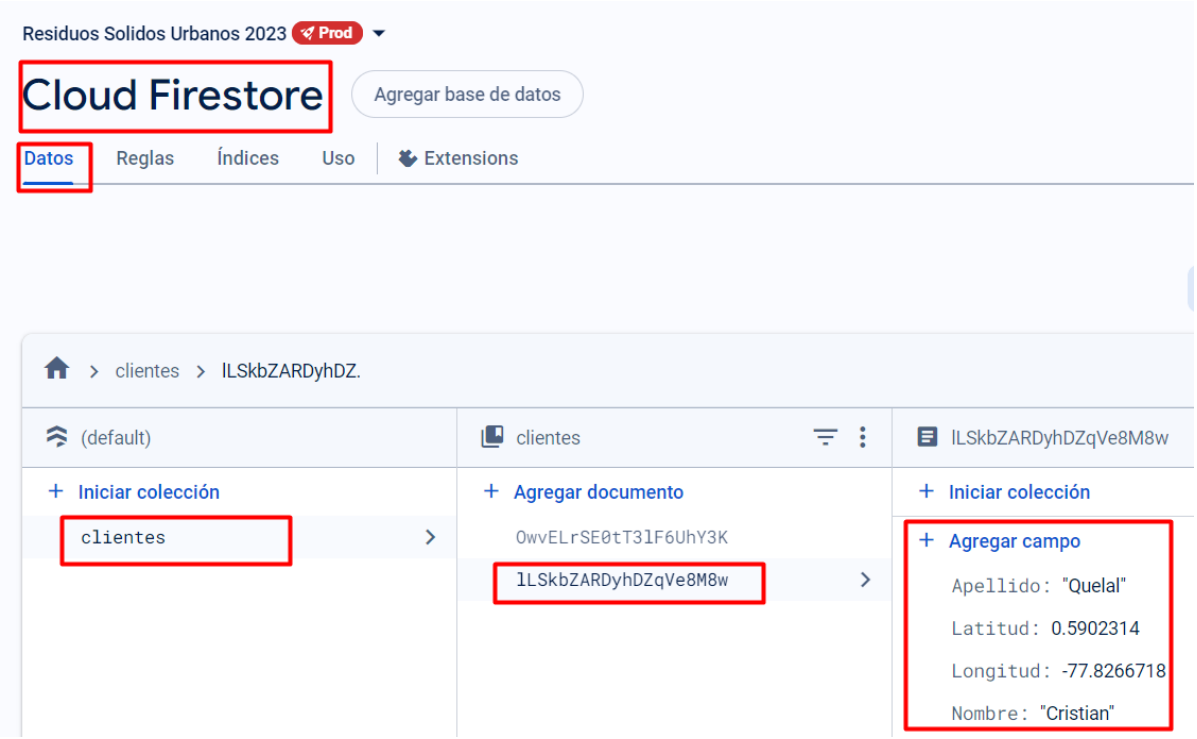
**Figura 49**

*Configuración de reglas de Cloud Firestore Database para la lectura y escritura de los datos de clientes del sistema de recolección de RSU*


La Figura 51 muestra la estructura de los datos almacenados en la base de datos Cloud Firestore. La información se organiza en colecciones, y en este caso en particular, la colección se denomina `clientes`. Dentro de esta colección, se crea un documento por cada cliente, el cual contiene los datos correspondientes a ese cliente específico. Esta estructura permite que la información se encuentre organizada y accesible, facilitando su gestión y posterior procesamiento.

**Figura 50**

*Visualización de la estructura de datos de clientes del sistema de recolección de RSU en Cloud Firestore Database*




- **API Mapbox**

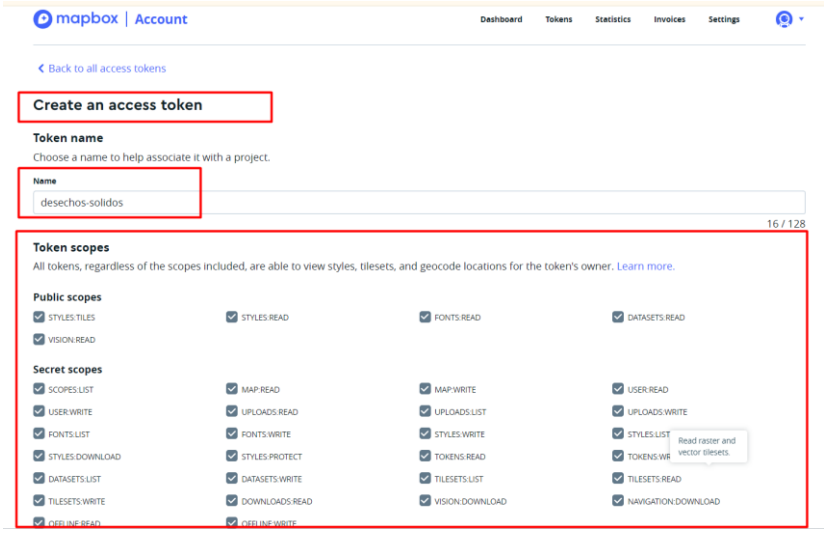
Por último, se configura la API de Mapbox, esta permite procesar los datos de las bases de datos revisadas previamente y crear las rutas de recolección, establecer la distancia entre el recolector - clientes, mostrar los puntos de recolección y la ubicación del recolector en mapas personalizados.

En primer lugar, es necesario crear una cuenta en el sitio oficial de Mapbox (**https://www.mapbox.com/**) para acceder a los servicios de su API. Una vez creada la cuenta, se pueden generar tokens personalizados, como se muestra en la Figura 52.

En este proceso, se asigna un nombre al Token y se definen sus características para que tenga acceso a los recursos de la API de Mapbox. Para obtener un Token que permita acceder a todas las funcionalidades de esta plataforma, es necesario incluir todos los scopes públicos y privados.

**Figura 51**

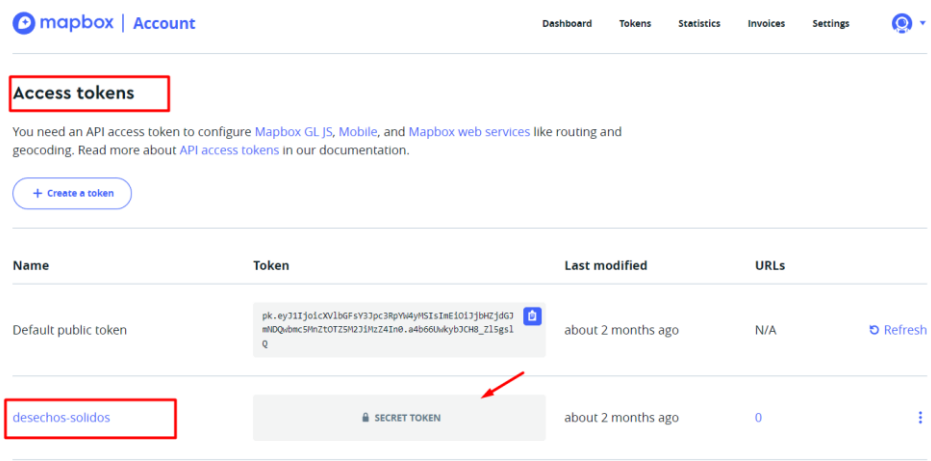
*Creación de Token de API de Mapbox para las funcionalidades de mapas requeridas en el Sistema de recolección de RSU*




Una vez creado el token, sólo se puede acceder al Token por una sola vez, ya que posteriormente permanecerá oculto para garantizar su seguridad, como se muestra en la Figura 53. Es importante tener en cuenta que, se debe crear un Token propio para cada aplicación, ya que el uso del Token público limitaría el acceso a muchas características y podría exponer datos importantes, comprometiendo así la seguridad del sistema.

**Figura 52**

*Token de API de Mapbox generado para implementar las funcionalidades de mapas en el Sistema de recolección de RSU*



Adicionalmente, se puede considerar la creación de mapas personalizados en el apartado "Creación de mapa en estudio". Esta opción, ubicada en la pantalla principal, permite generar mapas personalizados con características específicas para su visualización en la aplicación.

Con estas configuraciones, el bloque de almacenamiento y procesamiento de datos está listo para gestionar los datos recolectados y procesarlos para su posterior visualización en la aplicación móvil.

### *3.3.7 Estructura y configuración del Bloque de Visualización de Datos*

El bloque de visualización de datos del sistema de recolección de RSU es un componente crucial que permite a los usuarios interactuar con la información recolectada y procesada. Este bloque se encarga de presentar de forma clara y accesible los datos importantes para el proceso de recolección de los RSU.

Como se muestra en la Figura 54, el funcionamiento de este bloque involucra diferentes interfaces de usuario, estas se encuentran integradas en la aplicación móvil desarrollada, llamada **ResiRutas** con características específicas para cada una. Los datos procesados por este bloque son extraídos y formateados según los requisitos de visualización del usuario.

De acuerdo con el diagrama de funcionamiento de este bloque se tiene que, la aplicación móvil permite el ingreso de dos tipos de usuarios: clientes y recolectores. Para los **clientes**, solo se puede enviar una solicitud de recolección dentro del horario establecido (martes a domingo de 7:00 a.m. a 8:00 a.m.). Si el usuario está dentro del horario, el botón para enviar la solicitud se habilita. El cliente debe ingresar su nombre y apellido, y al hacer clic en el botón de aceptar, se capturan automáticamente las coordenadas actuales de su dispositivo, siempre que tenga activada la ubicación y haya otorgado los permisos correspondientes a la aplicación durante la instalación.

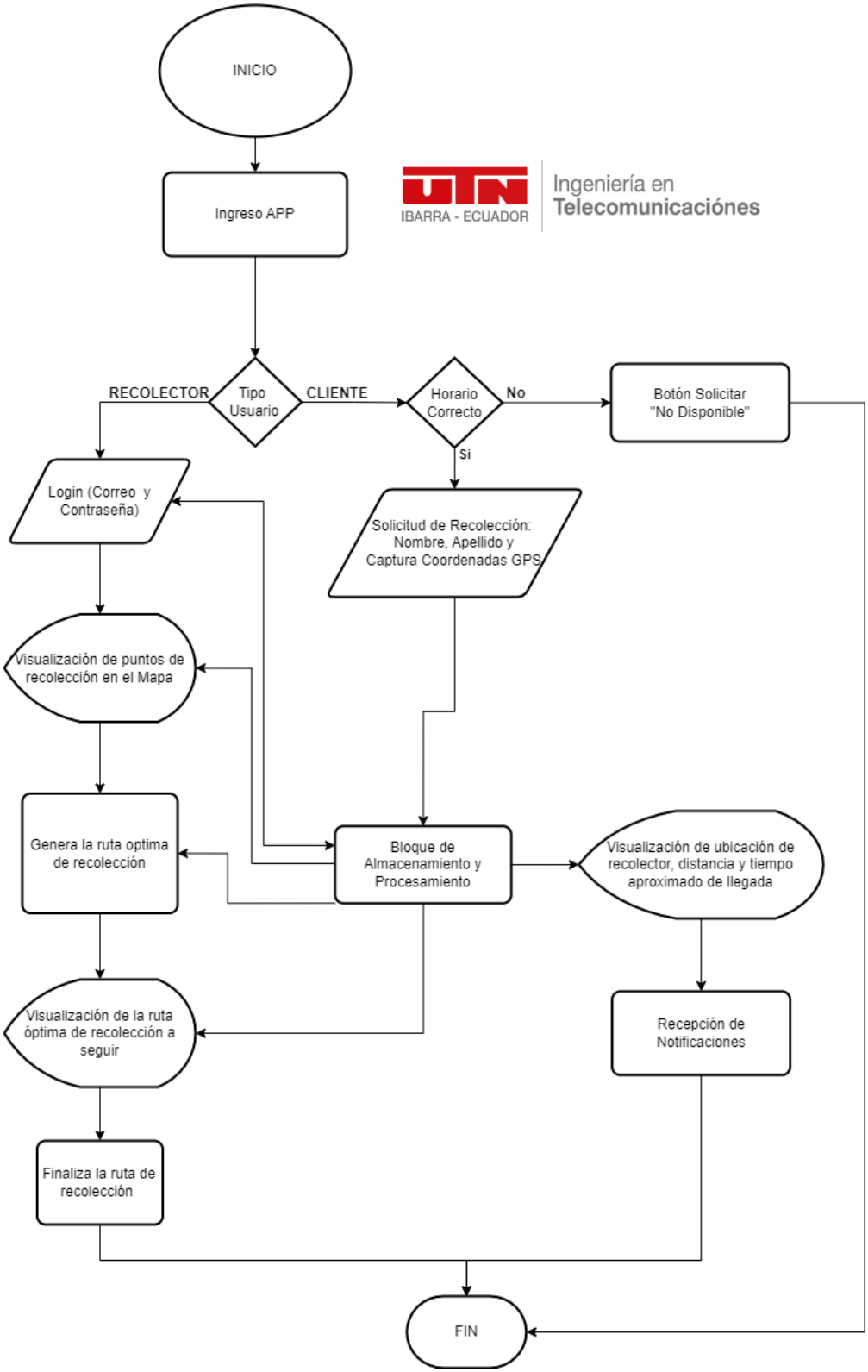
Una vez enviada la solicitud, los datos se envían al bloque de almacenamiento y procesamiento, y luego el cliente puede visualizar en la aplicación un mapa con la ubicación del recolector, la distancia existente entre los dos y el tiempo aproximado de llegada, además, recibe notificaciones importantes sobre el proceso de recolección.

Para los usuarios **recolectores**, es necesario iniciar sesión con las credenciales (correo y contraseña) otorgadas por el administrador del sistema. Una vez autenticado, el recolector puede visualizar en el mapa los puntos de recolección registrados. Luego, puede ir a generar la ruta de recolección a partir del botón de la interfaz. La siguiente interfaz cuenta con un botón para generar la ruta óptima de recolección y otro para finalizar dicha ruta. Al hacer clic en el botón de generar ruta, la aplicación realiza una solicitud al bloque de almacenamiento y procesamiento para obtener las ubicaciones(longitud y latitud) de los puntos de recolección y con esta información, se calcula y visualiza en el mapa la ruta óptima que el recolector debe seguir.

Una vez que el recolector haya completado la ruta generada, debe regresar a la interfaz previa y hacer uso del botón de finalizar la ruta, el cual elimina de la base de datos los registros de los clientes que solicitaron el servicio de recolección durante ese día, dando por terminado el proceso para el recolector.

**Figura 53**

*Diagrama de flujo del funcionamiento del bloque de visualización de datos del Sistema de recolección de RSU*



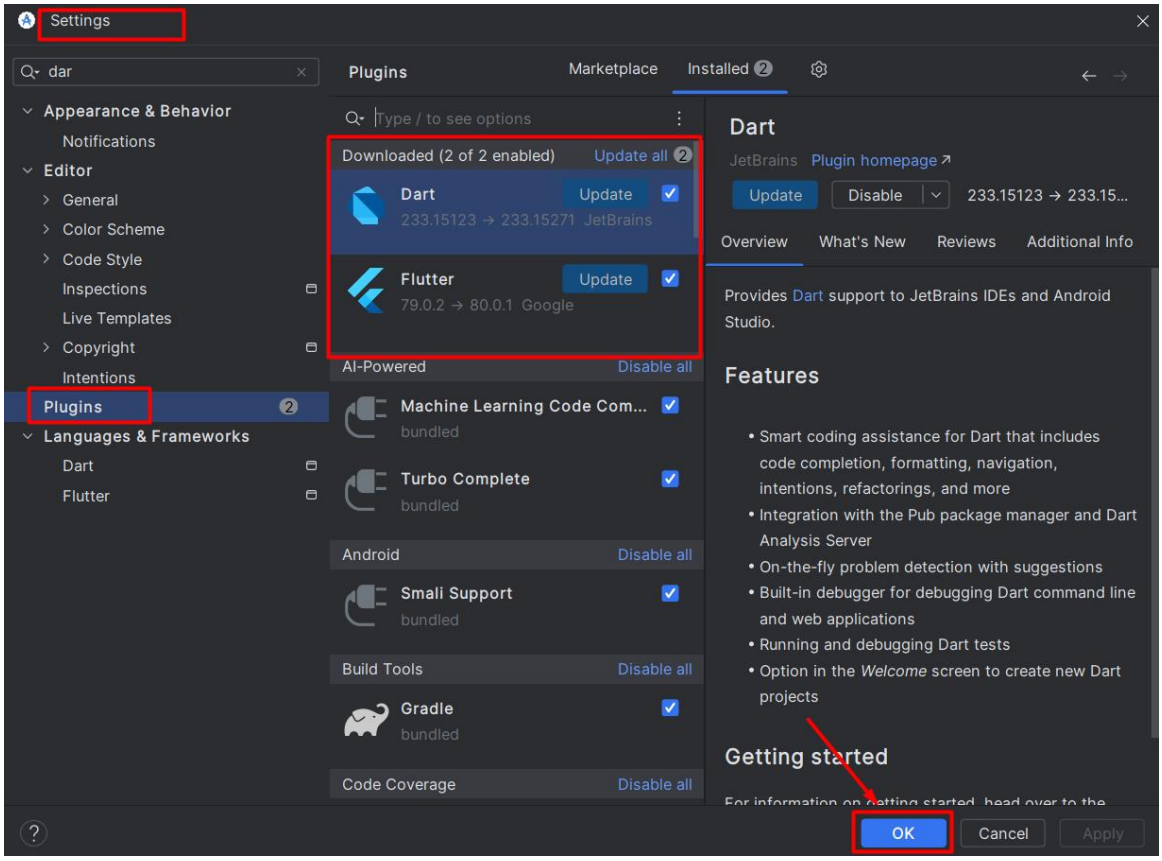
Una vez detallado el funcionamiento del bloque de visualización de datos a partir del diagrama de flujo previo, se detalla la programación de las características clave del sistema de recolección de RSU: la generación de rutas y las notificaciones del proceso. Dado que el código es extenso y consta de múltiples interfaces, se describen únicamente estos aspectos que demuestran el cumplimiento de los objetivos del proyecto. Para revisar detalladamente el algoritmo completo de la aplicación móvil, se puede acceder al código fuente en el **"Anexo 5".**

Antes de empezar, es necesario recordar que para programar las características de la aplicación móvil de recolección de RSU **ResiRutas**, se utilizan las herramientas definidas en los requerimientos: Android Studio con el complemento de Flutter y su lenguaje Dart. Por tanto, el primer paso es instalar Android Studio desde su página web oficial (**https://developer.android.com**). Una vez instalado Android Studio, es necesario integrar Flutter, para lo cual se debe descargar el SDK de Flutter desde su sitio web oficial (**https://docs.flutter.dev/**). Ambos sitios proporcionan guías de instalación específicas según el sistema operativo del usuario.

Con los dos softwares instalados, el siguiente paso es integrar Flutter en Android Studio. Para ello, se debe abrir Android Studio, ir a la sección de Plugins y buscar los complementos de Flutter y Dart, como se muestra en la Figura 55.

**Figura 54**

*Instalación de Plugins de Flutter y Dart en Android Studio para el desarrollo de la aplicación móvil para el sistema de recolección de RSU*

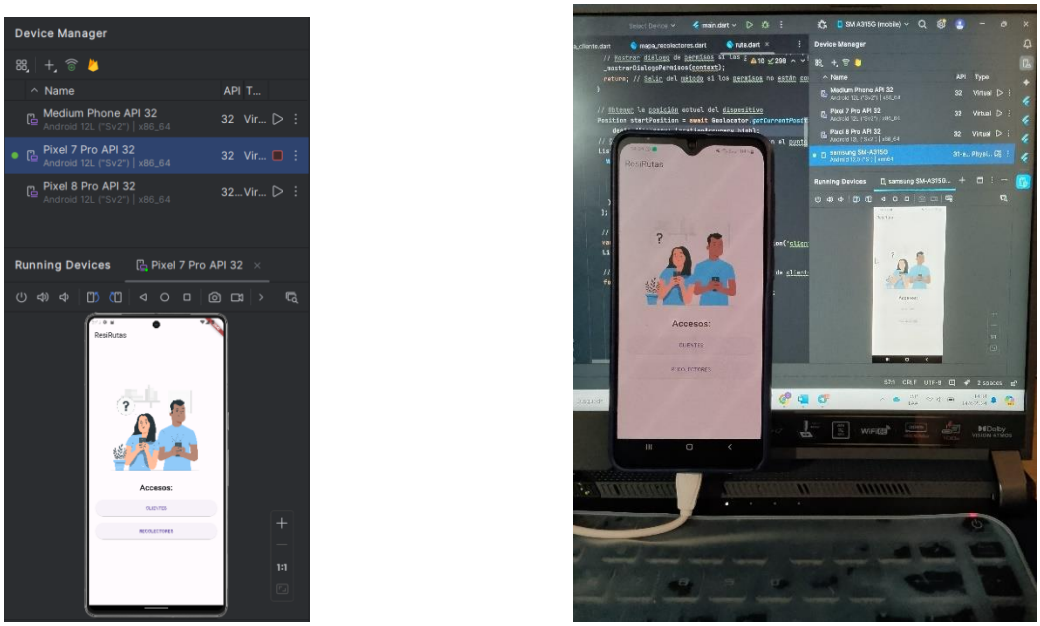



Una vez configurado el entorno de desarrollo, el siguiente paso es preparar uno o varios dispositivos para probar la aplicación de recolección de RSU. Android Studio ofrece la facilidad de utilizar tanto dispositivos virtuales como físicos para este propósito. En el caso de optar por un dispositivo virtual, el software proporciona el AVD (Android Virtual Device) Manager, una herramienta que permite crear y gestionar emuladores con diversas configuraciones de hardware y versiones de Android. Por otro lado, si se prefiere trabajar con un dispositivo físico, basta con conectarlo al computador mediante un cable USB y habilitar las opciones de desarrollador en el dispositivo. La Figura 56 ilustra ambas opciones: la Figura 56a muestra la interfaz del AVD Manager para la creación de dispositivos virtuales y la Figura

56b, un smartphone conectado y reconocido por Android Studio , listo para la instalación y prueba de la aplicación.

**Figura 55**

*Opciones de dispositivos para pruebas en Android Studio: AVD Manager y dispositivo físico*




**(a)** **(b)**

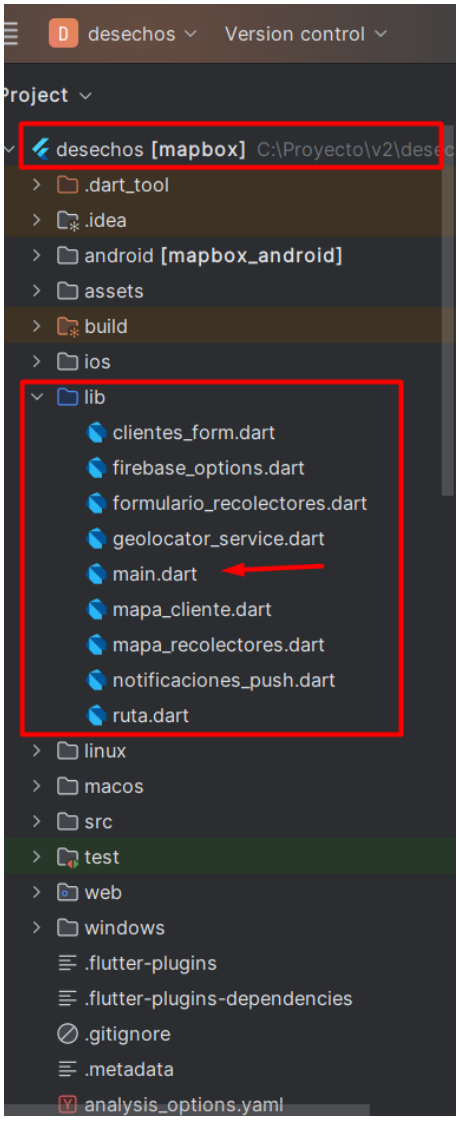
Al completar estos pasos iniciales correctamente, se puede crear, programar y probar aplicaciones móviles desarrolladas con Flutter en Android Studio.

### *3.3.7.1 Diseño y programación de la aplicación móvil para el sistema de recolección de RSU*

Al crear un proyecto de Flutter, se genera automáticamente una estructura organizada de archivos para el desarrollo de una aplicación móvil. El archivo principal, `main.dart,` se ubica en la carpeta `lib,` como se muestra en la Figura 57. Este es el punto de partida donde se comienza a escribir el código de la aplicación móvil. Esta incluye interfaces de usuario tanto para el cliente como para el recolector, por lo que cada interfaz y sus características se programan en archivos `.dart` separados, esto facilita una mejor gestión del diseño de la aplicación.

**Figura 56**

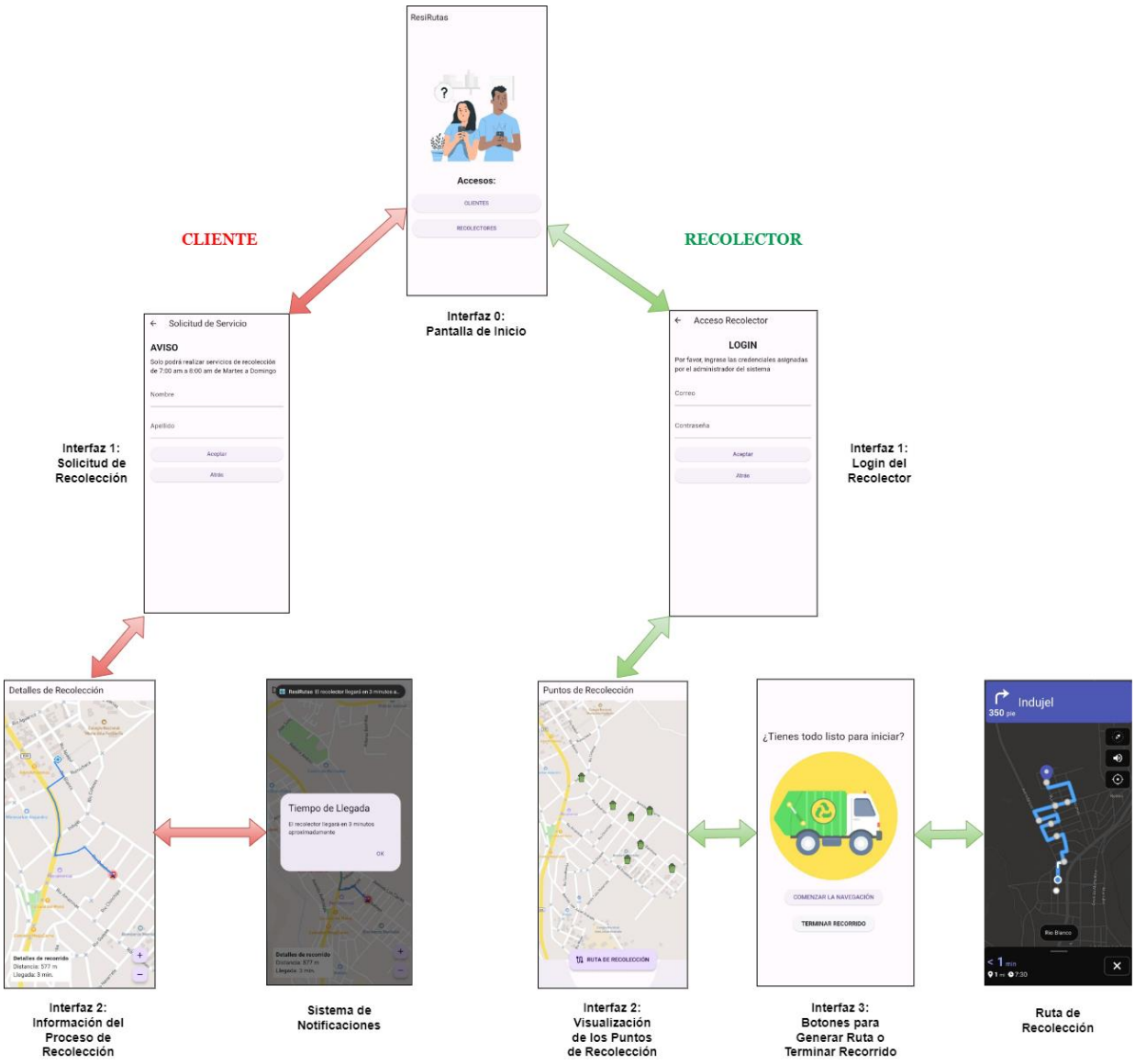
*Estructura de los archivos para el desarrollo de la aplicación móvil del Sistema de recolección de RSU*




La documentación y las librerías necesarias para el desarrollo en Flutter están disponibles en su página oficial y en **www.pub.dev**. Estas fuentes proporcionan guías detalladas sobre cómo importar y utilizar los paquetes necesarios según las características que se deseen implementar. Como se puede apreciar en la Figura 58, el diseño de las interfaces de la aplicación móvil prioriza la facilidad de navegación para los usuarios, de esta forma se garantiza el cumplimiento de los requerimientos establecidos para la aplicación móvil.

**Figura 57**

*Estructura de las interfaces de la aplicación móvil del Sistema de recolección de RSU*

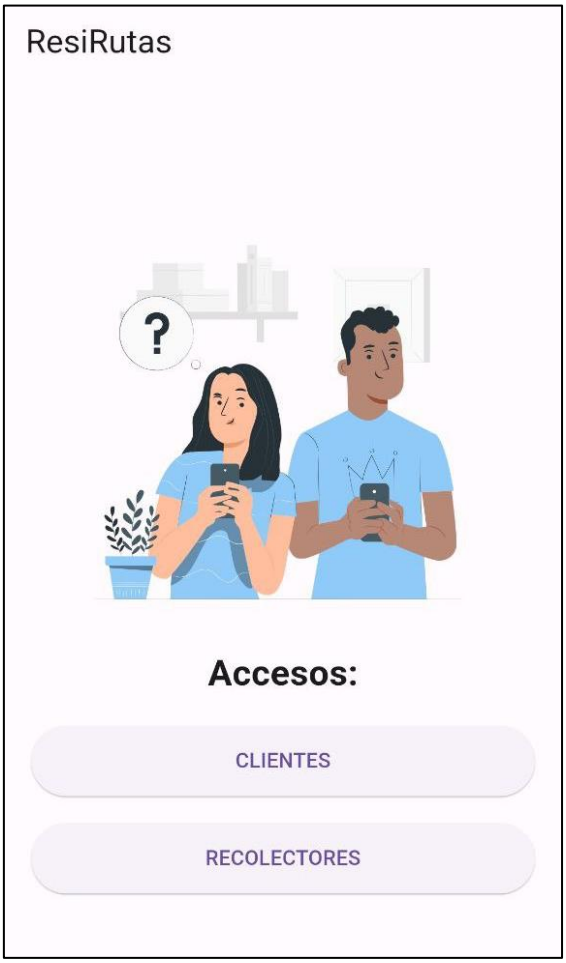



A continuación, se describe cada una de las interfaces, para proporcionar una comprensión detallada de la funcionalidad y la interacción de la información en cada una. Este análisis no solo facilita el entendimiento de la estructura operativa, sino que también permite apreciar cómo cada componente contribuye a la experiencia del usuario. La Figura 59 ilustra la Pantalla Inicial, esta sirve como punto de elección para dos tipos de usuarios. Por un lado, los clientes pueden ingresar directamente para enviar sus solicitudes de recolección. Mientras

que, los recolectores deben autenticarse mediante credenciales dadas por el administrador del sistema.

**Figura 58**

*Interfaz Inicial de la aplicación móvil del Sistema de recolección de RSU*




- **Interfaces para Clientes de la aplicación móvil del Sistema de recolección de RSU**

Al ingresar como **Cliente**, se muestra la interfaz para la solicitud de recolección, como se ilustra en la Figura 60. El botón `Aceptar` se activa únicamente durante el horario establecido, conforme al aviso visible en pantalla. Fuera de este período, el botón permanece deshabilitado, impidiendo el envío de solicitudes.

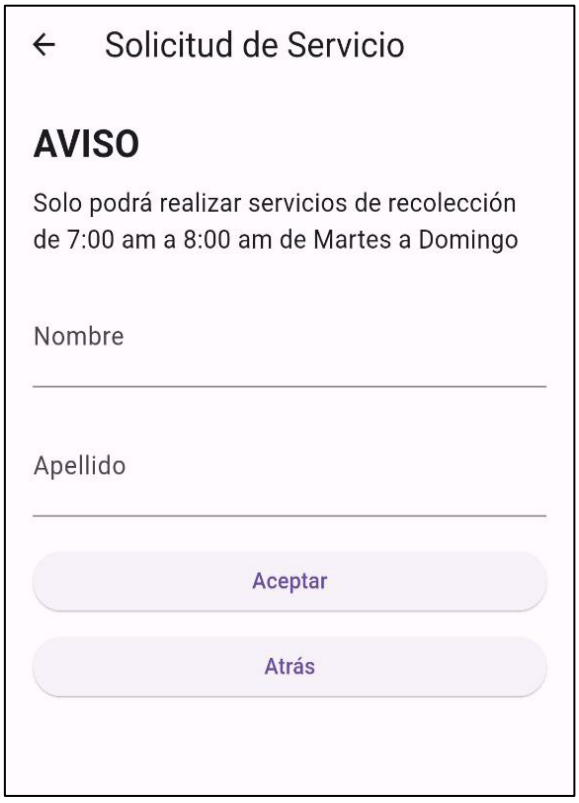
Para enviar una solicitud, el usuario debe proporcionar su nombre y apellido en los campos correspondientes, una vez completados los datos, al pulsar `Aceptar`, la aplicación

captura automáticamente la ubicación del dispositivo, estableciéndola como punto de recolección para la solicitud.

Es importante tener activada la ubicación del dispositivo móvil y haber otorgado los permisos pertinentes a la aplicación. Estos permisos se solicitan al instalar la aplicación y abrirla por primera vez, siendo estos indispensables para el correcto funcionamiento de las características que comprende el Sistema de recolección de RSU.

**Figura 59**

*Interfaz de Solicitud de servicio de recolección para los clientes de la aplicación móvil del Sistema de recolección de RSU*

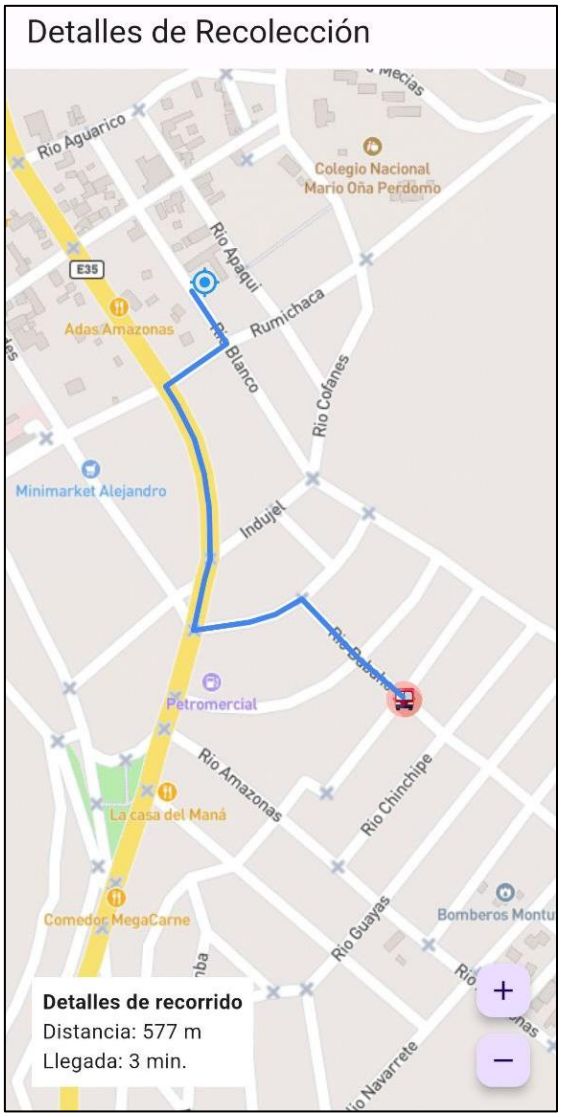



Tras generar la solicitud de servicio de recolección, se redirige automáticamente al cliente a la interfaz de seguimiento del proceso, como se ilustra en la Figura 61. Esta pantalla proporciona información detallada y en tiempo real sobre el estado de la recolección, mostrando la distancia en metros entre el recolector y la ubicación del cliente, así como el tiempo aproximado de llegada en minutos.

Adicionalmente, un mapa interactivo permite visualizar la ubicación actual del recolector y la ruta hasta el punto de recolección. Estos elementos dinámicos ofrecen al cliente una experiencia transparente, facilitando un control visual del progreso de su solicitud.

**Figura 60**

*Interfaz de visualización de los detalles del proceso de recolección para el cliente de la aplicación móvil del Sistema de recolección de RSU*

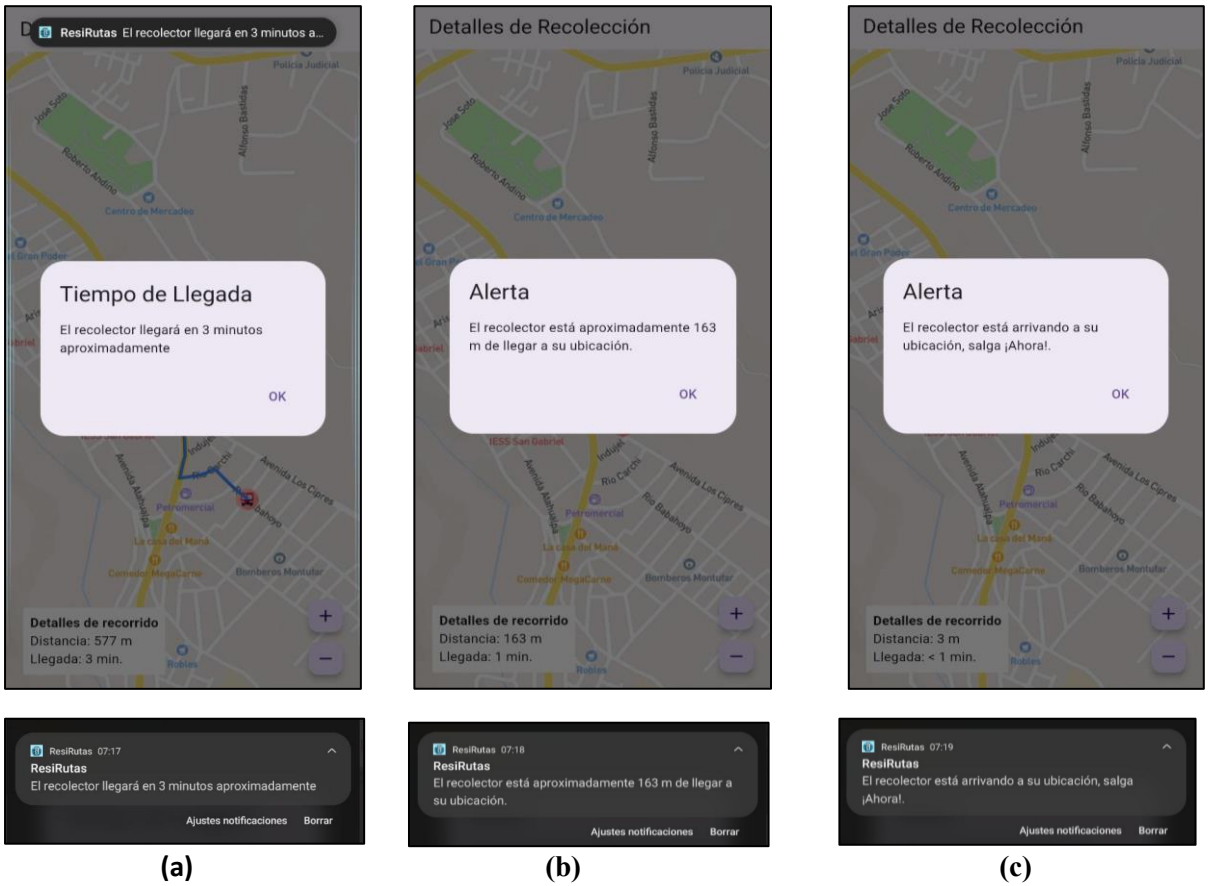



Basándose en los datos del recorrido, se generan notificaciones locales para mantener al usuario informado sobre el progreso de su solicitud como se muestra en la Figura 62. Estas alertas incluyen: el tiempo aproximado de llegada tras generar la solicitud (Figura 62a), la proximidad del recolector (Figura 62b) y el arribo del recolector al punto de recolección (Figura

62c). Además, para asegurar que el usuario reciba estas actualizaciones incluso cuando la aplicación está en segundo plano, se emiten notificaciones push.

**Figura 61**

*Notificaciones locales y push para alertar al cliente del proceso de recolección en la aplicación móvil del Sistema de recolección de RSU*




**(a)** **(b)** **(c)**

Para comprender la implementación de las notificaciones en tiempo real durante el proceso de recolección, es importante examinar la programación completa realizada en el archivo `mapa_cliente.dart` en el **Anexo 5**. Este archivo contiene el código fuente de las características visualizadas en las Figuras 61 y 62, abarcando tanto la interfaz de seguimiento como el sistema de notificaciones.

La configuración de estas funcionalidades requiere la importación de diversos paquetes, como se detalla en la Figura 63, en donde se pueden observar las librerías utilizadas, acompañadas de comentarios que especifican la función de cada una.

**Figura 62**

*Importación de paquetes necesarios para la programación de las características de detalles del recorrido de recolección y notificaciones*

```dart
//Importacion de paquetes necesarios
import 'dart:async'; // Importa el paquete para trabajar con funciones asíncronas y temporizadores
import 'dart:convert'; // Importa el paquete para codificación y decodificación de JSON
import 'package:flutter/material.dart'; // Importa el paquete Flutter para UI
import 'package:flutter_map/flutter_map.dart'; // Importa Flutter Map para integrar mapas
import 'package:latlong2/latlong.dart'; // Importa el paquete para manejar coordenadas geográficas
import 'package:firebase_core/firebase_core.dart'; // Importa Firebase Core para la inicialización de Firebase
import 'package:firebase_database/firebase_database.dart'; // Importa Firebase Realtime Database
import 'package:geolocator/geolocator.dart'; // Importa el paquete para obtener la ubicación del dispositivo
import 'package:http/http.dart' as http; // Importa el paquete HTTP para realizar solicitudes HTTP
import 'package:resirutas/notificaciones_push.dart'; // Importa las notificaciones push
import 'firebase_options.dart'; // Importa las opciones de configuración de Firebase
import 'package:flutter_svg/flutter_svg.dart'; // Importa SVG para mostrar imágenes vectoriales
import 'package:flutter_ringtone_player/flutter_ringtone_player.dart'; // Importa para reproducir tonos
```

Posteriormente, se programa una de las funciones más importantes `cargarMarcadores()` como se muestra en la Figura 64, esta permite implementar la geolocalización y visualización de la posición actual del usuario en un mapa interactivo. Inicialmente, utiliza el método `getCurrentPosition()` de la clase Geolocator para obtener las coordenadas GPS del dispositivo, configurado con `LocationAccuracy.high` para máxima precisión. Estas coordenadas se convierten en un objeto `LatLng`, representando la ubicación actual. Este objeto se utiliza para crear un `Marker` personalizado, configurado con dimensiones de `80x80` píxeles y un icono `Icons.my_location` en color azul, representando visualmente la ubicación del usuario en el mapa. Para finalizar, el método `move()` del `mapController` se invoca con la ubicación actual y un nivel de zoom de `16.0`. Esta operación centra el mapa en la posición del usuario, garantizando que su ubicación sea siempre visible.

**Figura 63**

*Programación de la función cargarMarcadores() para la visualización del marcador de ubicación del cliente en el mapa de la aplicación móvil*

```
void cargarMarcadores() async {
  // Función para cargar los marcadores en el mapa
  Position position = await Geolocator.getCurrentPosition( // Obtiene la posición actual del dispositivo
      desiredAccuracy: LocationAccuracy.high);
  LatLng currentLocation = LatLng(position.latitude, position.longitude); // Crea la ubicación actual

  Marker currentLocationMarker = Marker(
    // Crea un marcador para la ubicación actual
    point: currentLocation,
    width: 80,
    height: 80,
    child: const Icon(Icons.my_location, color: Colors.blue),
  );  // Marker

  mapController.move(currentLocation, 16.0); // Mueve la cámara del mapa a la ubicación actual
```

Siguiendo la metodología empleada para visualizar el marcador del cliente, se programa el marcador del recolector, como se ilustra en la Figura 65. La diferencia clave radica en el origen de los datos de ubicación: mientras que para el cliente se utiliza la geolocalización del dispositivo, para el recolector, la latitud y longitud se extraen de Firebase Realtime Database.

Para ello, se implementa una lógica de programación que establece un modo de escucha constante sobre Firebase, capturando y procesando los cambio en los datos. La Figura 65 muestra cómo se programa la actualización en tiempo real de la ubicación del recolector en el mapa. El código utiliza `dbRef.onValue.listen()` para estar atento a cambios en la base de datos Firebase. Cuando ocurre un cambio, el programa verifica si la pantalla aún está activa mediante el condicional `if` y el parámetro `mounted` para evitar actualizaciones innecesarias. Luego, obtiene los nuevos datos de ubicación de Firebase.

El programa extrae la latitud y longitud de estos datos utilizando `double.tryParse()`. Esta función intenta convertir los valores de texto en números decimales. Si la conversión falla, se usa 0.0 como valor predeterminado. Con estas coordenadas, se crea un nuevo punto en el mapa `LatLng` y se genera un marcador personalizado. Este marcador tiene un tamaño de `27x27` píxeles y usa una imagen

SVG(Scalable Vector Graphics) de un camión para representar al recolector. De esta manera, la posición del recolector se actualiza visualmente en el mapa cada vez que cambia en la base de datos.

**Figura 64**

*Programación de la función cargarMarcadores() para la visualización del marcador de ubicación del recolector en el mapa de la aplicación móvil*

```
dbRef.onValue.listen((event) async {
  // Escucha cambios en la base de datos Firebase
  if (mounted) { // Verifica si el widget aún está montado
    final data = event.snapshot.value as Map<dynamic, dynamic>?; // Obtiene los datos de Firebase
    if (data != null) {
      final lat = double.tryParse(data['lat'].toString()) ??
          0.0; // Obtiene la latitud de los datos
      final lng = double.tryParse(data['lng'].toString()) ??
          0.0; // Obtiene la longitud de los datos
      LatLng currentPointB = LatLng(
          lat, lng); // Crea la ubicación del punto B
      final marker = Marker(
        // Creacion de un marcador para el punto B
        point: currentPointB,
        width: 27,
        height: 27,
        child: SvgPicture.asset(
          'assets/recolectores.svg', // Ruta del archivo SVG del camión
        ),  // SvgPicture.asset
      );  // Marker
```

Para graficar y visualizar la ruta entre el recolector-cliente, se utiliza la API de Direcciones de Mapbox. Este proceso, ilustrado en la Figura 66, muestra cómo se obtiene la ruta entre el cliente y el recolector usando la API de Direcciones de Mapbox. Primero, se construye una `URL` especial que incluye las coordenadas de ambos puntos (longitud y latitud) y un token de acceso único para autenticar la solicitud. Esta `URL` contiene parámetros específicos como `geometries=geojson` para obtener la ruta en un formato compatible con el mapa.

Luego, el programa envía una solicitud a esta URL usando `http.get()` y espera la respuesta. Cuando llega la respuesta, se convierte de formato JSON a un formato que el programa puede usar fácilmente. De esta respuesta, se extraen los puntos que forman la ruta.

Específicamente, el programa accede a la lista de coordenadas dentro de la respuesta JSON y procesa cada par de coordenadas. Cada punto se convierte de las coordenadas que da Mapbox (longitud, latitud) al formato `LatLng` que usa el mapa en la aplicación, invirtiendo el orden de las coordenadas para que coincida con el formato requerido. El programa realiza esta conversión para cada punto de la ruta y crea una nueva lista con todos estos puntos convertidos. El resultado es una lista completa de puntos `LatLng` que, al unirlos, forman la ruta completa entre el cliente y el recolector, permitiendo mostrarla visualmente en el mapa.

**Figura 65**

*Definición de parámetros para la generación de ruta entre cliente y recolector mediante el uso de la API de Direcciones de Mapbox*

```
// Definicion de parametros HTTP para la api de mapbox para obtener la ruta entre la ubicación actual y el punto B
String url = "https://api.mapbox.com/directions/v5/mapbox/driving/$lng,$lat;${position
    .longitude},${position
    .latitude}?alternatives=false&geometries=geojson&overview=full&steps=false&access_token=pk.eyJ1IjoibmFjaG93ZWIi
var response = await http.get(
    Uri.parse(url)); // Realiza la solicitud HTTP
var json = jsonDecode(response.body); // Decodifica la respuesta JSON
var points = (json['routes'][0]['geometry']['coordinates'] as List)
    .map((e) => LatLng(e[1], e[0]))
    .toList(); // Obtiene los puntos de la ruta
```

Para los datos de información del proceso de recolección el programa sigue la lógica de programación que se ilustra en la Figura 67, aquí se calcula el tiempo estimado de llegada del recolector. Inicialmente, se captura la velocidad del recolector directamente desde Firebase Realtime Database y se la convierte en formato de cadena de texto, para luego almacenarla en la variable `Speed`. Luego, se convierte esta velocidad en un valor numérico mediante `double.parse(speedData).`

Paralelamente, se extrae la distancia en metros calculada por la API de Mapbox desde la respuesta JSON y se la almacena en la variable `distance`. Luego, la cadena de distancia es dividida en partes utilizando el método `split(" ")` para separar el valor numérico de su unidad. El valor numérico resultante es convertido a un tipo double utilizando `double.parse(),` almacenando este valor en la variable `dist.` Luego, el valor de `dist` es

redondeado al entero más cercano utilizando el método `round()`, almacenando este valor en la variable `distanceIntTemp`. Finalmente, la distancia en metros es convertida a kilómetros dividiendo el valor de `dist` por 1000, y el resultado es almacenado en la variable `distanceInKm`.

Con estos parámetros, se calcula el tiempo aproximado de llegada mediante la Ec. 4, donde `d` y `v` corresponden a los datos de distancia y velocidad respectivamente, considerando el caso especial cuando la velocidad es igual a 0, por tanto, no se realiza el cálculo y se muestra `Recolector Detenido`.

$$t = \frac{d}{v} \qquad \textbf{Ec. 4}$$

Todos los valores se guardan en variables que luego alimentan la interfaz, proporcionando al usuario información clara y actualizada sobre el progreso del servicio de recolección.

**Figura 66**

*Programación para preparación y cálculo de valores de distancia, velocidad y tiempo de arribo del recolector en el proceso de recolección*

```
// Capturando la velocidad directamente desde la base de datos
final speedData = data['speed'].toString();
speed = "$speedData Km/h"; // Mostrando la velocidad tal como está en la base de datos
double speedInKmPerHour = double.parse(speedData);
// Obtener la distancia calculada por MapBox
distance = "${json['routes'][0]['distance']} m";
double dist = double.parse(
    distance.split(" ")[0]); // Distancia en metros
int distanceIntTemp = dist.round();
// Convertir la distancia de metros a kilómetros
double distanceInKm = dist / 1000;

if (speedInKmPerHour == 0) {
  // Si la velocidad es cero, el recolector está detenido
  arrivalTime = "Recolector detenido";
} else {
  // Calcular el tiempo en horas y convertirlo a minutos
  double timeInHours = distanceInKm / speedInKmPerHour;
  int arrivalTimeMinutes = (timeInHours * 60).round(); // Convertir a minutos y redondea
```

Para el sistema de notificaciones, la Figura 68, ilustra cómo se programa las notificaciones basadas en el tiempo y la distancia calculados previamente. La lógica de programación se basa en el uso de condicionales para determinar el formato del tiempo de llegada: si el tiempo es menor a un minuto, se muestra `< 1 min.` , de lo contrario, se muestra el tiempo en minutos. Además, para las notificaciones, si el tiempo de llegada `arrivalTimeMinutes` es mayor a 0 minutos, se emite una notificación informando el tiempo de llegada.

Asimismo, el código genera notificaciones en función de la distancia `dist`: si el recolector está a 200 metros o menos. De manera similar, si la distancia es de 10 metros o menos, se emite una notificación final indicando la llegada inminente del recolector al punto de recolección.

**Figura 67**

*Lógica de programación para la implementación de las notificaciones sobre el proceso de recolección para alertar al cliente*

```
if (arrivalTimeMinutes == 0) {
  arrivalTime =
  "< 1 min."; // Si el tiempo es menor a un minuto a mostrar cuando el tiempo de llegada sea 0 minutos
} else {
  arrivalTime =
  "$arrivalTimeMinutes min."; // Tiempo estimado de llegada
}

// Mostrar el mensaje de diálogo de notificacion una vez que se haya calculado el tiempo de llegada
if (arrivalTimeMinutes > 0 && !arrivalTimeAlertShown) {
  showArrivalDialog(arrivalTimeMinutes);
  arrivalTimeAlertShown = true;
}

// Mostrar el mensaje de diálogo de notificacion una vez que el recolector este a <= 200m de distancia del punto
if (dist <= 200 && !nearAlertShown) {
  showNearAlertShow(dist);
  nearAlertShown = true;
}

// Mostrar el mensaje de diálogo de notificacion una vez que el recolector este a <= 10m de distancia del punto
if (dist <= 10 && !arrivalAlertShown) {
  showArrivalAlertShown(dist);
  arrivalAlertShown = true;
```

Como se visualiza en la Figura 68, para activar las notificaciones se llama a las funciones correspondientes de cada notificación. A continuación, se detalla el ejemplo de la programación de la función para la notificación del tiempo de llegada.

En la Figura 69, al llamar a la función `showArrivalDialog(),` considerando el parámetro `arrivalTimeMinutes`, se verifica que esta función realiza varias acciones importantes.

Primero, crea un mensaje que indica el tiempo de llegada del recolector en minutos. Luego, reproduce un sonido de notificación usando `FlutterRingtonePlayer.play().` A continuación, muestra una notificación push con el mensaje creado mediante `mostrarNotificaciones(mensaje).` Finalmente, abre un cuadro de diálogo con el mensaje del tiempo de llegada, este incluye un botón `OK` que permite al usuario cerrarlo. De forma similar se encuentran programadas las funciones de las demás notificaciones, esto se puede revisar más detalladamente en el código fuente de la aplicación móvil en el **"Anexo 5".**

**Figura 68**

*Programación de la función para mostrar la notificación del tiempo aproximado de llegada del recolector al punto de recolección*

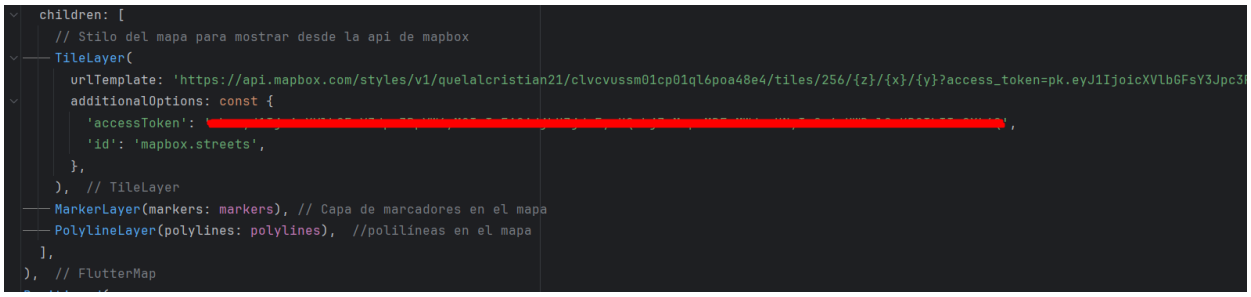
```
void showArrivalDialog(int minutes) {
// Función para mostrar el diálogo del tiempo de llegada
String mensaje = "El recolector llegará en $minutes minutos aproximadamente";
FlutterRingtonePlayer.play(
    fromAsset: "assets/sounds/sound_alert.mp3"); // Reproduce el sonido de notificación
mostrarNotificaciones(mensaje); // Muestra la notificación push
showDialog(
  context: context,
  builder: (BuildContext context) {
    return AlertDialog(
      title: const Text("Tiempo de Llegada"),
      content: Text(
          mensaje),  // Text
      actions: <Widget>[
        TextButton(
          child: const Text("OK"),
          onPressed: () {
            Navigator.of(context).pop();
```

Una vez realizada la programación de las características necesarias para la interfaz de mapas de clientes, se procede a crear la interfaz utilizando las librerías propias de Flutter. Esto incluye el diseño del cuerpo de la interfaz y los respectivos botones. Un apartado importante que hay que considerar es que el mapa principal que se muestra en la interfaz de cliente se implementa con la ayuda de la API de Mapbox, como se muestra en la Figura 70.

Por tanto, se configura el estilo del mapa mediante la clase `TileLayer`, utilizando una plantilla mediante la `URL` proporcionada por Mapbox. Esta configuración incluye el `accessToken` necesario para autenticar las solicitudes a la API de Mapbox y el `id` asignado al mapa. Además del mapa base, se añaden capas adicionales: `MarkerLayer` para mostrar los marcadores correspondientes en el mapa y `PolylineLayer` para dibujar la ruta entre cliente-recolector. Es importante mencionar que el `accessToken,` se encuentra oculto por razones de seguridad, ya que son datos sensibles y únicos del proyecto.

**Figura 69**

*Programación para configurar las capas del Mapa con API de Mapbox*



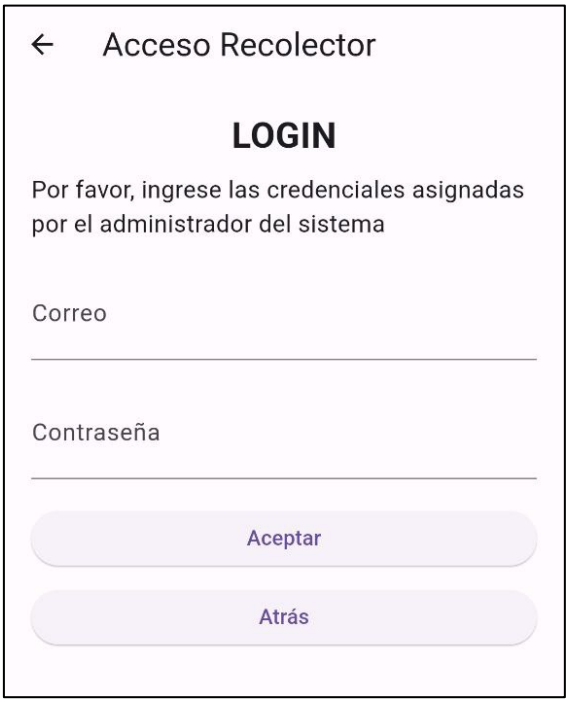

- **Interfaces para Recolectores de la aplicación móvil del Sistema de recolección de RSU**

Tras analizar el desarrollo del apartado del usuario Cliente, se procede a examinar el de Recolectores. Al acceder como Recolector, se muestra la interfaz de Login, que requiere las credenciales (`correo` y `contraseña`) asignadas por el administrador, como se indica en la Figura 71.

Al introducir las credenciales correctas y pulsar `Aceptar,` el usuario es redirigido a la siguiente interfaz. En caso de ingresar datos erróneos, se muestra un cuadro de diálogo informando sobre el error y solicitando el ingreso de las credenciales adecuadas.

**Figura 70**

*Interfaz para inicio de sesión del usuario Recolector mediante las credenciales asignadas por el administrador del Sistema de recolección de RSU*




Tras un login exitoso, el Recolector es redirigido automáticamente a la interfaz de `Puntos de Recolección`, como muestra la Figura 72. En esta pantalla, se visualiza un mapa con las ubicaciones de los puntos de recolección, identificados mediante iconos de tachos de basura verdes.

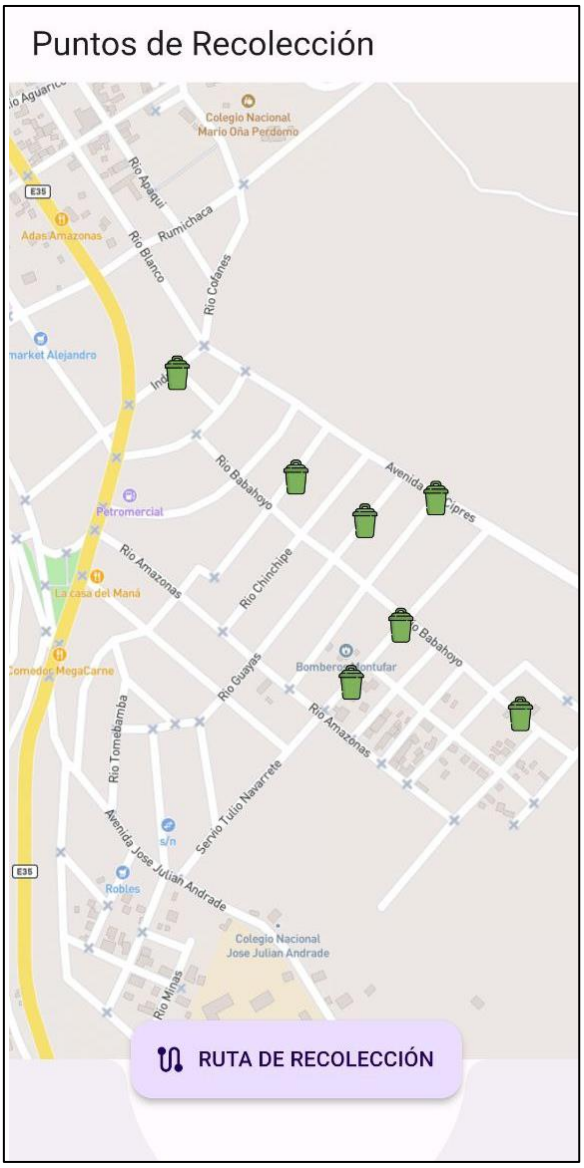
Esta interfaz se implementa de manera similar al mapa del Cliente. Utilizando la API de Mapbox, se genera un mapa que muestra marcadores correspondientes a las ubicaciones de los clientes, cuyos datos se extraen de la base de datos Firestore.

Además, se incluye un botón `RUTA DE RECOLECCIÓN` que conduce a la siguiente interfaz, donde se puede generar la ruta o finalizar el recorrido. Para un análisis más detallado

de la implementación, se puede consultar el código fuente de la aplicación, específicamente las líneas de programación del archivo `mapa_recolectores.dart` en el "**Anexo 5**".

**Figura 71**

*Interfaz de visualización de los puntos de recolección a partir de las solicitudes de cada cliente generada en el Sistema de recolección de RSU*




La Figura 73 muestra la interfaz accesible tras pulsar el botón `Ruta de Recolección` en la pantalla anterior (Figura 72). Esta sencilla pantalla consta de dos botones: `COMENZAR LA NAVEGACIÓN` y `TERMINAR RECORRIDO`.

Al activar `COMENZAR LA NAVEGACIÓN`, se genera la ruta óptima de recolección basada en los puntos de los clientes, extraídos de Firestore, por tanto, se muestra una interfaz con la ruta e indicaciones visuales y de audio, guiando al recolector por cada punto de recolección.

Una vez finalizada la navegación, se retorna a la interfaz de la Figura 73, donde al seleccionar `TERMINAR  RECORRIDO` se concluye el proceso, eliminando las solicitudes de recolección del sistema. De esta forma, el recolector queda listo para repetir el procedimiento al día siguiente.

**Figura 72**

*Interfaz con los botones para Comenzar la Navegación y Terminar Recorrido para la ruta de recolección de RSU*


La generación de la ruta es un componente base de la aplicación móvil en el sistema de recolección de RSU. Para comprender su implementación, se analiza los apartados más importantes del desarrollo de su lógica de programación, cuyo código fuente completo se encuentra en el archivo `ruta.dart` del "**Anexo 5**".

El primer paso en la creación de la ruta óptima de recolección y navegación guiada es la importación de los paquetes necesarios, como se ilustra en la Figura 74. Los comentarios asociados a cada línea detallan la función de estos paquetes en la lógica de programación.

**Figura 73**

*Importación de paquetes necesarios para la generación y visualización de la ruta optima de recolección*

```
//Importacion de paquetes necesarios
import 'package:flutter/material.dart'; // Importa el paquete de Flutter para construir interfaces de usuario
import 'package:flutter_mapbox_navigation/flutter_mapbox_navigation.dart'; // Importa la biblioteca de navegación Mapbox
import 'package:geolocator/geolocator.dart';  // Importa la biblioteca Geolocator para obtener la posición geográfica
import 'package:cloud_firestore/cloud_firestore.dart'; // Importa Firestore para trabajar con la base de datos en la nube
import 'package:flutter_svg/svg.dart';  // Importación para usar SvgPicture
```

Para generar la ruta de recolección se hace uso del método `iniciarNavegación()`, la Figura 75 presenta un segmento clave del método, primero, se obtiene la posición actual del dispositivo con `Geolocator.getCurrentPosition(),` especificando una alta precisión de ubicación `LocationAccuracy.high`. Esta posición se utiliza para crear una lista de puntos de ruta `wayPoints,` empezando con un punto de inicio que contiene las coordenadas de latitud y longitud obtenidas.

Luego, el código obtiene datos de clientes desde Firestore a través de una consulta a la colección clientes usando `_firestore.collection('clientes').get().` Los documentos obtenidos se almacenan en `clientesSnapshot.docs`.

A continuación, se crea una lista vacía de puntos de destino `destinationPoints` tipo array. Para cada documento en `clientesSnapshot.docs`, se extraen los datos de latitud y longitud de los clientes, se convierten a tipo `double`, y se verifica si no son nulos. Si ambos valores son válidos, se añade un nuevo punto de destino a la lista `destinationPoints` con el nombre `Destino` y las coordenadas correspondientes.

Este proceso es fundamental, ya que prepara los datos geográficos necesarios: el punto de partida del recolector y todos los destinos a visitar. Estos datos sirven posteriormente para generar la ruta óptima de recolección.

**Figura 74**

*Lógica de programación para obtener los waypoints de Inicio y Destino para generar la ruta de recolección de RSU*

```
// Obtener la posición actual del dispositivo
Position startPosition = await Geolocator.getCurrentPosition(
    desiredAccuracy: LocationAccuracy.high);
// Crear lista de puntos de ruta (waypoints) con el punto de inicio
List<WayPoint> wayPoints = [
  WayPoint(
    name: "Inicio",
    latitude: startPosition.latitude,
    longitude: startPosition.longitude,
  ),
];

// Obtener datos de clientes desde Firestore
var clientesSnapshot = await _firestore.collection('clientes').get();
List<WayPoint> destinationPoints = [];

// Crear puntos de destino basados en los datos de clientes
for (var doc in clientesSnapshot.docs) {
  var data = doc.data() as Map<String, dynamic>;
  double? lat = data['Latitud']?.toDouble();
  double? lng = data['Longitud']?.toDouble();
  if (lat != null && lng != null) {
    destinationPoints.add(
      WayPoint(
        name: "Destino",
        latitude: lat,
        longitude: lng,
```

Tras obtener los puntos de recolección, como se detalló en la Figura 75, se procede a un proceso importante: la organización de los waypoints según un algoritmo de optimización de ruta, ilustrado en la Figura 76. Este algoritmo implementa una versión simplificada del Problema del Viajante (TSP, por sus siglas en inglés) para determinar la secuencia más eficiente de visitas a los puntos de recolección.

El método `_ordenarPuntosPorRutaOptimizada` implementa el algoritmo de optimización de ruta basado en una versión simplificada del Problema del Viajante (TSP). Este

método toma como entrada una `Position startPosition` y `una List<WayPoint> destinationPoints`, retornando una `List<WayPoint>` optimizada. La función inicia creando un `WayPoint` inicial utilizando las coordenadas de `startPosition`, estableciendo así el punto de partida para la optimización.

En la parte principal del algoritmo se conforma por un bucle `while` que continúa hasta que `destinationPoints` esté vacía. En cada iteración, se utiliza el método `sort()` de la lista para ordenar los puntos restantes. La función de comparación calcula la distancia entre el punto actual y cada punto de destino utilizando `Geolocator.distanceBetween().` Esta función de permite determinar la proximidad entre puntos geográficos.

Para manejar posibles valores nulos y asegurar la precisión de los cálculos, el código implementa verificaciones de nulidad y conversiones a double para las coordenadas de latitud y longitud. Esto se realiza mediante operadores ternarios y el método `toDouble(),` garantizando que se utilicen valores numéricos válidos en los cálculos de distancia.

Tras ordenar los puntos, el algoritmo selecciona el más cercano utilizando `removeAt(0),` luego se añade a `orderedPoints` con `add(),` y lo establece como el nuevo punto actual. Este proceso iterativo construye progresivamente la ruta optimizada, seleccionando en cada paso el punto más cercano al actual.

El uso de estructuras de datos como `List` para `orderedPoints` y `destinationPoints` permite manipulaciones eficientes de los puntos de ruta.

**Figura 75**

*Lógica de programación para ordenar los puntos de destino/recolección a partir del algoritmo del problema del viajante (Travelling Salesman Problem: TSP por sus siglas en inglés)*

```
// Método para ordenar los puntos de destino usando un algoritmo TSP simplificado
List<WayPoint> _ordenarPuntosPorRutaOptimizada(Position startPosition, List<WayPoint> destinationPoints) {
  List<WayPoint> orderedPoints = [];
  WayPoint currentPoint = WayPoint(
    name: "Inicio",
    latitude: startPosition.latitude,
    longitude: startPosition.longitude,
  );
  while (destinationPoints.isNotEmpty) {
    destinationPoints.sort((a, b) {
      // Verificación de que las coordenadas sean no nulas y transformación a double
      double latA = a.latitude != null ? a.latitude!.toDouble() : 0.0;
      double lonA = a.longitude != null ? a.longitude!.toDouble() : 0.0;
      double latB = b.latitude != null ? b.latitude!.toDouble() : 0.0;
      double lonB = b.longitude != null ? b.longitude!.toDouble() : 0.0;

      // Verificación de que currentPoint.latitude y currentPoint.longitude no sean nulos y transformación a double
      double currentLat = currentPoint.latitude != null ? currentPoint.latitude!.toDouble() : 0.0;
      double currentLon = currentPoint.longitude != null ? currentPoint.longitude!.toDouble() : 0.0;

      double distA = Geolocator.distanceBetween(currentLat, currentLon, latA, lonA);
      double distB = Geolocator.distanceBetween(currentLat, currentLon, latB, lonB);

      return distA.compareTo(distB);
    });
    WayPoint nextPoint = destinationPoints.removeAt(0);
    orderedPoints.add(nextPoint);
    currentPoint = nextPoint;
  }
  return orderedPoints;
```

Tras la preparación de la ruta con wayPoints, que incluye el punto de inicio y los destinos ordenados por mediate el algoritmo TSP, se procede a iniciar la navegación utilizando el SDK(Kit de Desarrollo de Software) de Mapbox, como se ilustra en la Figura 77, en donde se muestra la configuración de `MapBoxOptions`, esto permite la personalización de la experiencia de navegación.

Se establecen parámetros iniciales como la posición de partida `StartPosition`, nivel de zoom, y orientación del mapa. Características importantes `como enableRefresh, alternatives, e isOptimized` se activan para garantizar rutas actualizadas, opciones de rutas alternativas y generar trayectos optimizados entre los puntos de recolección

respectivamente. Además, para la navegación se activa `voiceInstructionsEnabled` y `bannerInstructionsEnabled` para ofrecer guía por voz y visual. De igual forma, se especifica el modo de navegación como:

`MapBoxNavigationMode.drivingWithTraffic`, esto permite que para las rutas generadas se considere las condiciones de tráfico del sector.

Por último, la función `_navigation.startNavigation()` inicia el proceso utilizando las opciones configuradas y la lista `wayPoints`. Aquí se aprovecha de la API de Mapbox Directions para calcular la ruta óptima, producir instrucciones detalladas y adaptarse al perfil de desplazamiento especificado, en este caso, navegación con tráfico.

**Figura 76**

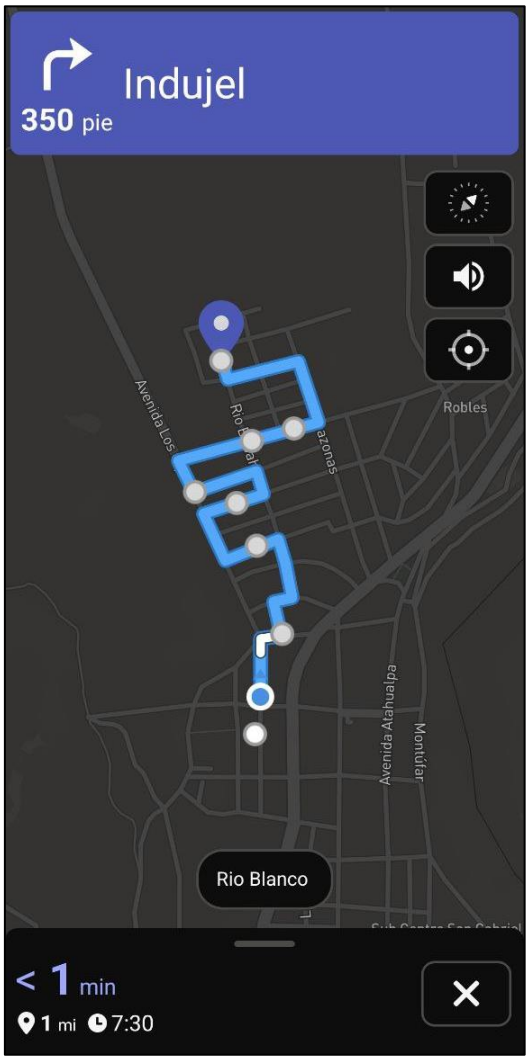
*Configuración y activación de la navegación con el SDK de API de Mapbox*

```
  // Configuración de opciones para Intefaz de MapBoxNavigation
  var options = MapBoxOptions(
    initialLatitude: startPosition.latitude,  // Latitud inicial para centrar el mapa al inicio de la navegación
    initialLongitude: startPosition.longitude,   // Longitud inicial para centrar el mapa al inicio de la navegación
    zoom: 15.0,   // Nivel de zoom inicial del mapa
    tilt: 0.0, // Ángulo de inclinación inicial del mapa
    bearing: 0.0, // Orientación inicial del mapa en grados
    enableRefresh: true,  // Habilita la actualización automática de la ruta
    alternatives: true,  // Permite rutas alternativas
    isOptimized: true,  // Optimiza la ruta para la navegación
    voiceInstructionsEnabled: true, // Habilita las instrucciones de voz durante la navegación
    bannerInstructionsEnabled: true, // Habilita las instrucciones en banner durante la navegación
    allowsUTurnAtWayPoints: true,   // Permite hacer giros en U en los puntos de ruta
    mode: MapBoxNavigationMode.drivingWithTraffic, // Modo de navegación (con tráfico, sin tráfico, etc.)
    units: VoiceUnits.metric,   // Unidades de voz para las instrucciones de navegación
    simulateRoute: true,  // Simula la ruta sin conexión a internet
    animateBuildRoute: true, // Animación al construir la ruta en el mapa
    longPressDestinationEnabled: false, //Agrega un punto mas al mapa si se mantiene presionado la pantalla
    language: "es", // Idioma de las instrucciones de navegación
  );
  // Iniciar navegación con las opciones y waypoints configurados
  await _navigation.startNavigation(
    options: options,
    wayPoints: wayPoints,
  );
}
```

Finalmente, la Figura 78 muestra el resultado de activar `COMENZAR LA NAVEGACIÓN`, que invoca el método `_iniciarNavegacion()`. La interfaz provee información dinámica como el tiempo estimado de llegada al siguiente destino, la distancia restante y la hora prevista de arribo.

**Figura 77**

*Interfaz de navegación y ruta optima generada por API de Navegación de Mapbox*




Al completar el recorrido, el usuario `recolector` puede cerrar la navegación con el botón `X`, retornando a la interfaz de la Figura 73. Allí, se puede seleccionar el botón `Terminar Recorrido` y eliminar las solicitudes procesadas, preparando el sistema para reiniciar el ciclo al día siguiente. Este flujo asegura una gestión eficiente de la recolección de residuos, optimizando rutas y manteniendo actualizado el estado de las solicitudes.

# CAPÍTULO IV: PRUEBAS DE FUNCIONAMIENTO

Siguiendo la metodología en cascada, este capítulo aborda las pruebas de funcionamiento del sistema de recolección de RSU. Tras completar las etapas de requerimientos, diseño y codificación de las características del sistema, se procede a realizar la integración y pruebas del sistema. Este apartado complementa a la codificación, debido a que, asegura que cada componente del sistema pueda operar en conjunto, permitiendo identificar y corregir posibles fallos de diseño. Además, se presenta un análisis de costo-beneficio del sistema diseñado y se elabora un plan de implementación. Este plan propone cómo se debe poner en operación y el mantenimiento del sistema diseñado, cumpliendo así con las etapas finales de la metodología en cascada.

## 4.1 Pruebas del Sistema de recolección de RSU

Para realizar un proceso ordenado de verificación de las características de funcionamiento del sistema de recolección de RSU, se establece un cronograma de pruebas. La Tabla 26 presenta la planificación de las pruebas con los respectivos resultados esperados y el tiempo de duración de cada prueba, permitiendo validar la viabilidad técnica del sistema.

**Tabla 26**

*Cronograma de Pruebas del Sistema de recolección de RSU*

| **Cronograma de Pruebas** | | | |
|---|---|---|---|
| **Tipo de Prueba** | **Ubicación** | **Resultado Esperado** | **Duración** |
| **Prueba 1** Funcionamiento del bloque de obtención de datos | Entorno Controlado | Se constata la inicialización adecuada del microcontrolador y su conexión exitosa a la red Claro. Se verifica la obtención, actualización y transmisión continua de datos GPS al bloque de almacenamiento y procesamiento, así como el consumo de datos durante la transferencia. Se comprueba la duración de la batería para estimar su rendimiento operativo. Finalmente, se | 7 días |

| | | | |
|---|---|---|---|
| | | determina el tiempo de encendido y envío de datos del microcontrolador, estableciendo así la duración del proceso de inicialización. | |
| **Prueba 2** Funcionamiento del bloque de almacenamiento y procesamiento | Entorno Controlado | Se comprueba la integración y funcionamiento de los servicios de Firebase y Mapbox. Primero, se verifica mediante la interfaz de Realtime Database la recopilación constante de los datos del GPS enviados por el nodo recolector, al igual que, se constata el almacenamiento de los datos de las solicitudes de clientes en Cloud Firestore. Luego, se comprueba la eficacia del servicio de autenticación de Firebase para el login del usuario recolector. Finalmente, se valida el procesamiento adecuado de las solicitudes de rutas y la generación de mapas utilizando la API de Mapbox. | 7 días |
| **Prueba 3** Funcionamiento del bloque de visualización de datos | Entorno Controlado | Se verifica la recepción y presentación de los datos en las diferentes interfaces de la aplicación móvil desarrollada, asegurando que toda la información sea legible, coherente y se actualice correctamente en tiempo real. | 8 días |
| **Prueba 4** Funcionamiento del sistema de recolección de RSU con solicitudes de recolección simuladas | Entorno Controlado | Se comprueba la integración efectiva de todos los componentes del sistema de recolección de Residuos Sólidos Urbanos (RSU). Se verifica la respuesta adecuada del sistema ante diferentes cantidades de solicitudes de recolección simuladas, asegurando la correcta comunicación entre el nodo recolector, las bases de datos y la aplicación móvil. Se comprueba la generación precisa de rutas y actualización en tiempo real de la información de recolección. | 16 días |

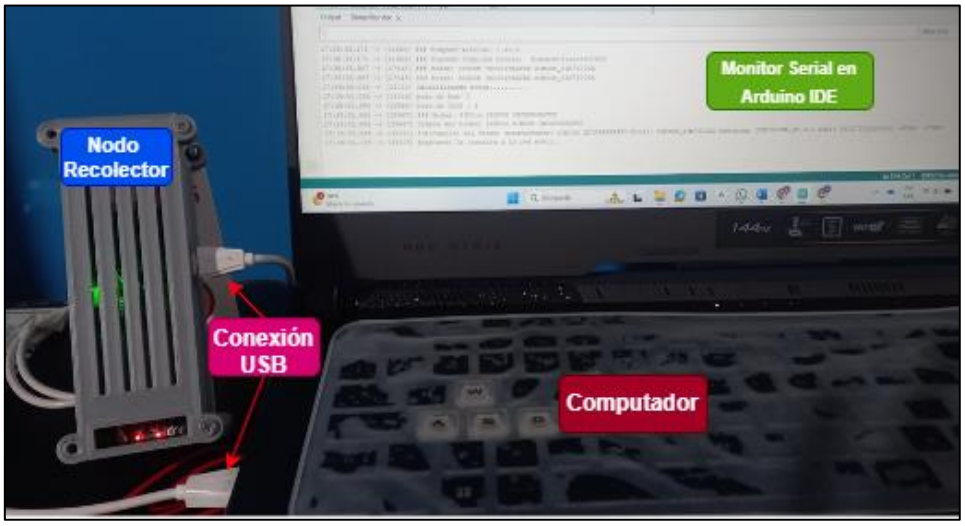
### *4.1.1 Pruebas de funcionamiento del bloque de obtención de datos*

Tras la integración física del sistema y la codificación de las características requeridas, se inicia la fase de verificación del bloque de obtención de datos. La Figura 79 muestra el nodo recolector, el cual es el componente central de este bloque.

Para verificar el funcionamiento , se conecta a un computador y se utiliza el "Monitor Serial" del IDE de Arduino. Este depurador permite comprobar mediante mensajes, la correcta inicialización del sistema, la conexión a la red, la obtención de datos GPS y el envío exitoso de la información a Firebase. Este proceso asegura que todos los elementos del nodo recolector operen de manera adecuada y coordinada.

**Figura 78**

*Conexión nodo recolector y computadora para verificar su funcionamiento correcto mediante los mensajes de depuración en el monitor serial del Arduino IDE*




La Figura 80 muestra los mensajes de inicialización del Arduino ESP32 T-SIM7600NA, estos mensajes proporcionan información sobre la versión del software, la identificación del módem, modelo, versión y confirma su inicialización exitosa. Además, se verifica que los modos de Red y GNSS se han establecido correctamente, indicando que las configuraciones programadas se han aplicado con éxito. **Figura 79**

*Verificación de mensajes de inicialización correcta del microcontrolador ESP32 T-SIM7600NA*

```
[11054] ### TinyGSM Version: 0.12.0
[11054] ### TinyGSM Compiled Module:  TinyGsmClientSIM7600
[11112] ### Modem: SIMCOM INCORPORATED SIMCOM_SIM7600NA
[11112] ### Modem: SIMCOM INCORPORATED SIMCOM_SIM7600NA
[11141] Modem inicializado exitosamente
[11152] Modo de Red: 1
[11164] Modo de GNSS : 1
[13181] ### Modem: SIMCom SIMCOM INCORPORATED
[13181] Nombre del Modem: SIMCom SIMCOM INCORPORATED
[13210] Informacion del Modem: Manufacturer: SIMCOM INCORPORATED Model: SIMCOM_SIM7600NA Revision: SIM7600NA_V2.0.2
```

Luego de inicializar el dispositivo, se establece la conexión a la red móvil, la Figura 81 ilustra el proceso de conexión del dispositivo a la red móvil. Como se observa, el sistema espera inicialmente la conexión a la red, luego confirma que se ha establecido exitosamente. Inmediatamente, se conecta al APN (Access Point Name) `internet.claro.com.ec` y verifica el estado GPRS(General Packet Radio Service). También se visualiza la obtención correcta de información importante como el CCID, IMEI, IMSI, el operador al cual se ha establecido conexión (CLARO EC Claro), la dirección IP local asignada, y la potencia de la señal de la red móvil, la cuál es `-67 dBm`, indicando una buena intensidad de señal 4G LTE para la operación del sistema de recolección de RSU.

**Figura 80**

*Verificación del proceso de conexión del dispositivo a la red móvil*

```
[13220] Esperando la conexión a la red movil..
[13244] Red movil conectada
[13244] Conectando al APN:  internet.claro.com.ec
[13397] GPRS estado: Conectado
[13397] Conexion establecida con:  internet.claro.com.ec
[13412] CCID: 8959301001008364823
[13422] IMEI: 860371052664582
[13433] IMSI: 740010200836482
[13450] Operador: CLARO EC Claro
[13464] Direccion IP Local asignada: 100.92.53.150
[13476] Potencia de la señal de la red móvil: -67  dBm
```

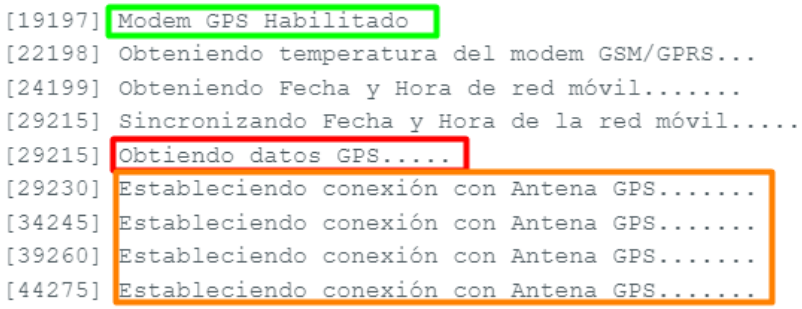
En caso de que la red móvil, el APN o el GPRS no se conecten, el sistema muestra mensajes informando sobre la falla en la conexión. Luego, reintenta estas conexiones, esto

asegura que el dispositivo establezca la conectividad necesaria para su funcionamiento adecuado.

Una vez establecida la conexión a la red móvil LTE, el sistema procede a obtener los datos de ubicación y otros parámetros importantes para verificar la operación del dispositivo. La Figura 82 ilustra esta secuencia: primero se habilita el módem GPS y se obtiene la temperatura del módem GSM/GPRS. Luego, el dispositivo sincroniza su fecha y hora con la red móvil para garantizar la precisión de los registros. Finalmente, inicia el proceso de obtención de datos GPS, realizando múltiples intentos de conexión con la antena GPS hasta lograr una conexión estable.

**Figura 81**

*Verificación de secuencia para la obtención de datos de ubicación, temperatura y fecha/hora*




Continuando con el proceso de obtención de datos de ubicación, la Figura 83 muestra el resultado exitoso de la obtención de los datos GPS, tras varios intentos de establecer conexión con la antena GPS. Por tanto, se presentan las coordenadas geográficas (latitud y longitud), así como información adicional importante como la velocidad, altitud, precisión de la medición, fecha, hora y zona horaria. Estos datos son fundamentales para permitir la localización exacta del vehículo recolector en tiempo real.

**Figura 82**

*Verificación de la obtención exitosa de los datos GPS*

```
[39260] Estableciendo conexión con Antena GPS.......
[44275] Estableciendo conexión con Antena GPS.......
[49296] Coordenadas Obtenidas: Latitude: 0.59020883 °   Longitude: -77.82667542  °
[49297] Velocidad: 0.00  km/h   Altitud: 2829.90  m
[49307] Velocidad: 0.00  m/s    Precicion: 1.50  m
[49308] Fecha: 27/06/2024
[49308] Hora: 18:21:04
[49308] Timezone: -5.00
```

Una vez obtenidos los datos de ubicación, se inicia la secuencia de conexión con el servidor de base de datos en la nube para enviar la información recopilada. La Figura 84 muestra este proceso: el dispositivo se conecta al servidor específico del proyecto a través del puerto 443, confirmando una conexión exitosa mediante el código de estado 200. Luego, el sistema transmite los datos recopilados a Firebase. Al final, se registra el tiempo total de ejecución del proceso de inicialización, que es de 24553 ms (aproximadamente 25 segundos). Es importante notar que este tiempo se reducirá significativamente en operaciones posteriores, ya que el sistema estará inicializado y solo realizará la obtención y sincronización de datos.

**Figura 83**

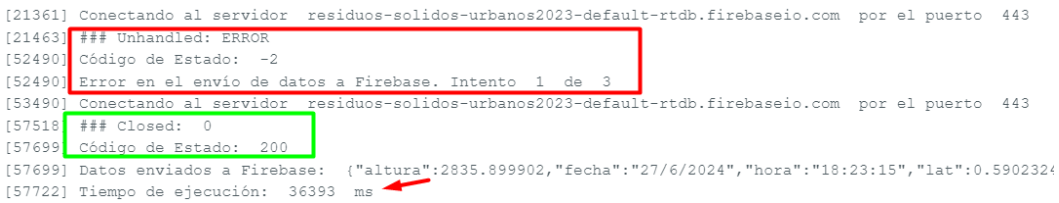
*Verificación de la secuencia exitosa de establecimiento de conexión con el servidor de base de datos y envio de los datos GPS recolectados*

```
[49308] Conectando al servidor residuos-solidos-urbanos2023-default-rtdb.firebaseio.com  por el puerto  443
[53564] ### Closed:  0
[53744] Código de Estado:  200
[53745] Datos enviados a Firebase:  {"altura":2829.899902,"fecha":"27/6/2024","hora":"18:21:4","lat":0.590208828,"lng":
[53768] Tiempo de ejecución:  24553  ms
```

En caso de que no se establezca la conexión con el servidor, el sistema muestra un mensaje de `ERROR` y un código de estado diferente a 200, consecuentemente, el dispositivo intenta reconectarse hasta tres veces. Si no se logra la conexión después de estos intentos, el módem se reinicia para reconfigurarse, de lo contrario los datos se envían como se muestra en la Figura 85. Además, es importante notar que estos reintentos de conexión aumentan el tiempo total de ejecución del proceso.

**Figura 84**

*Verificación de la secuencia seguida ante una conexión fallida con el servidor de base de datos*




El proceso descrito previamente abarca la inicialización completa del nodo recolector, desde el encendido del dispositivo hasta la confirmación de recepción del primer dato en el servidor Firebase. Este ciclo incluye la conexión a la red móvil, la obtención de datos GPS, el envío de información al bloque de almacenamiento y procesamiento, y la recepción del código de estado.

Por tanto, el tiempo de inicialización es esencial para el funcionamiento óptimo del sistema. Las pruebas detalladas en la Tabla 27 muestran que el nodo recolector tarda un promedio de 41.91 segundos en inicializarse. El tiempo fluctúa debido a factores como la conexión a la red móvil y la adquisición de datos GPS, que dependen de los procesos internos de establecimiento de conexión y la disponibilidad de satélites, respectivamente. Basándose en estas pruebas, se verifica la necesidad de esperar aproximadamente 1 minuto después de encender el dispositivo del nodo recolector para garantizar una inicialización completa y una transmisión exitosa de los datos de ubicación del camión recolector.

**Tabla 27**

*Pruebas de tiempo de inicialización del nodo recolector*

| Número de Prueba | Tiempo promedio (s) |
|---|---|
| #1 | 33.26 |
| #2 | 33.20 |
| #3 | 54.10 |

| | |
|---|---|
| #4 | 43.80 |
| #5 | 35.66 |
| #6 | 44.25 |
| #7 | 33.81 |
| #8 | 53.55 |
| #9 | 43.63 |
| #10 | 43.83 |
| **Tiempo promedio** | 41.91 |

Finalmente, se verifica que durante la operación normal del nodo recolector, los datos se envían cada 5 segundos, como se muestra en la Figura 86. Este intervalo corto permite una actualización prácticamente en tiempo real de la información. Sin embargo, es importante destacar que, en situaciones específicas como la inicialización del sistema o cuando se producen errores, el tiempo de ejecución puede extenderse, en estos casos, los mensajes se envían rápidamente una vez que se resuelve la situación.

**Figura 85**

*Verificación de la operación normal del nodo recolector del Sistema de recolección de RSU*

```
19:48:45.307 -> [215567] ....                              ...
19:48:45.307 -> [215567] ....PROCESANDO NUEVOS DATOS.      ...
19:48:45.307 -> [215567] ....                              ...
19:48:45.307 -> [215577]
19:48:45.307 -> [215595] Temperatura Modem GSM/GPRS: 25.00 °C
19:48:45.343 -> [215608] Sincronizando Fecha y Hora de la red móvil.....
19:48:45.343 -> [215608] Obteniendo datos GPS.....
19:48:45.343 -> [215629] Coordenadas Obtenidas: Latitude: 0.59023672 °  Longitude: -77.82665253  °
19:48:45.381 -> [215629] Velocidad: 0.00  km/h  Altitud: 2837.70  m
19:48:45.381 -> [215640] Velocidad: 0.00  m/s   Precicion: 1.10  m
19:48:45.381 -> [215641] Fecha: 27/06/2024
19:48:45.381 -> [215641] Hora: 19:48:45
19:48:45.381 -> [215641] Timezone: -5.00
19:48:45.381 -> [215652] Conectando al servidor  residuos-solidos-urbanos2023-default-rtdb.firebasei
19:48:49.399 -> [219650] ### Closed:  0
19:48:49.580 -> [219846] Código de Estado:  200
19:48:49.580 -> [219846] Datos enviados a Firebase:  {"altura":2837.699951,"fecha":"27/6/2024","hora
19:48:49.580 -> [219872] Tiempo de ejecución:  4264  ms
19:48:50.348 -> [220608]
19:48:50.348 -> [220608] ....                              ....
19:48:50.348 -> [220608] ....PROCESANDO NUEVOS DATOS.....
19:48:50.348 -> [220608] ....                              ....
19:48:50.348 -> [220618]
19:48:50.348 -> [220636] Temperatura Modem GSM/GPRS: 25.00 °C
19:48:50.383 -> [220651] Sincronizando Fecha y Hora de la red móvil.....
19:48:50.383 -> [220651] Obteniendo datos GPS.....
19:48:50.383 -> [220675] Coordenadas Obtenidas: Latitude: 0.59023672 °  Longitude: -77.82665253  °
19:48:50.416 -> [220675] Velocidad: 0.00  km/h  Altitud: 2837.70  m
19:48:50.416 -> [220686] Velocidad: 0.00  m/s   Precicion: 1.10  m
19:48:50.416 -> [220686] Fecha: 27/06/2024
19:48:50.416 -> [220686] Hora: 19:48:50
19:48:50.416 -> [220686] Timezone: -5.00
```

#### *4.1.1.1 Pruebas de consumo de datos móviles en el envío de datos del GPS del nodo recolector a la base de datos*

Una vez verificado el funcionamiento del nodo recolector del Sistema de recolección de RSU, es importante conocer el consumo de datos móviles que se genera durante la transmisión de información GPS a la base de datos, con el fin de optimizar el consumo en caso de ser necesario. La Tabla 28 muestra los resultados de seis pruebas, representando los días de operación del sistema (de martes a domingo), cada una con una duración de 2 horas para simular el tiempo máximo diario del proceso de recolección actual.

El análisis revela un consumo promedio de 9.99 MB por cada período de 2 horas, resultando en un consumo semanal estimado de 59.94 MB (6 días de operación). Considerando un escenario máximo de 27 días de operación mensual de acuerdo con el calendario, se proyecta un consumo mensual estimado de 269.73 MB.

**Tabla 28**

*Pruebas de consumo de datos móviles en la transmisión de datos del GPS del nodo recolector a la base de datos*

| **Número de Prueba** | **Tiempo de Prueba (horas)** | **Consumo promedio (MB)** |
|---|---|---|
| #1 | 2 | 10.32 |
| #2 | 2 | 9.27 |
| #3 | 2 | 10.19 |
| #4 | 2 | 10.25 |
| #5 | 2 | 10.45 |
| #6 | 2 | 9.49 |
| **Consumo Promedio** | 2 | 9.99 |

Es importante notar la variabilidad diaria en el consumo visualizado en la Tabla 28, que oscila entre 9.27 MB y 10.45 MB, es atribuible a factores como procesos internos de conexión

a la red móvil, reintentos de envío de paquetes, y variaciones en los procesos de solicitudes a los servidores en internet. Además, el bajo consumo de datos móviles registrado demuestra que la transmisión de datos se está realizando de manera óptima.

Teniendo en cuenta esta variabilidad y el máximo de días operativos, se estima que un plan de datos móviles con una capacidad mensual de al menos 500 MB asegura la operación continua del sistema en todos los escenarios posibles.

#### *4.1.1.2 Pruebas de duración de batería del nodo recolector*

Tras analizar el consumo de datos, es crucial examinar la duración de la batería del nodo recolector. Este hace uso de una batería de 2600 mAh, con un consumo teórico para el microcontrolador de 200 mAh en operación normal, según lo establecido en el capítulo previo. Inicialmente, basándose en estos datos teóricos, se estimó que la batería cargada completamente tiene una duración de 13 horas de operación sin necesidad de recarga, equivalente a 6.5 días de funcionamiento de 2 horas diarias.

Sin embargo, las pruebas prácticas revelan una considerable discrepancia con respecto a la estimación teórica. La Tabla 29 presenta los resultados la prueba inicial, realizadas con la batería completamente cargada y verificando el estado de operación cada dos horas.

Los resultados indican una duración total de 12 horas y 10 minutos de operación, lo que se traduce en 6 días completos de operación de dos horas cada uno, más 10 minutos adicionales en un séptimo día. Esta duración es 50 minutos menor que la estimación teórica inicial.

**Tabla 29**

*Pruebas de duración de la batería del nodo recolector*

| Número de Verificación | Tiempo de duración de batería | Estado de Operación |
|---|---|---|
| #1 | 2 horas | Operativo |
| #2 | 2 horas | Operativo |

| | | |
|---|---|---|
| #3 | 2 horas | Operativo |
| #4 | 2 horas | Operativo |
| #5 | 2 horas | Operativo |
| #6 | 2 horas | Operativo |
| #7 | 10 min | Batería agotada |
| **Total** | 12:10:00 horas | |

Para obtener resultados más concluyentes, se obtienen datos en dos pruebas adicionales. La Tabla 30 presenta un resumen de las tres pruebas completas de duración de batería. El tiempo promedio de duración de la batería es de 12 horas y 2 minutos, lo que confirma la discrepancia con la estimación teórica inicial. Esta diferencia resalta la importancia de considerar factores adicionales que afectan el consumo energético en condiciones reales, como la temperatura de operación.

Es importante notar que la prueba #2 muestra una duración de 11:55:00 horas, esto demuestra que no alcanza a cumplir las 12 horas de operación necesarias para alcanzar los 6 días de funcionamiento de 2 horas diarias. Sin embargo, cabe destacar que estas 2 horas representan el tiempo máximo actual de operación. Con la implementación del sistema de rutas óptimas, se espera que el tiempo de operación diario sea menor a 2 horas, lo que permitiría cumplir con el requisito de 6 días de funcionamiento incluso con la duración de batería más baja registrada.

**Tabla 30**

*Resumen de pruebas de duración total de batería del nodo recolector*

| **Número de Verificación** | **Tiempo de duración de batería** |
|---|---|
| #1 | 12:10:00 horas |
| #2 | 11:55:00 horas |

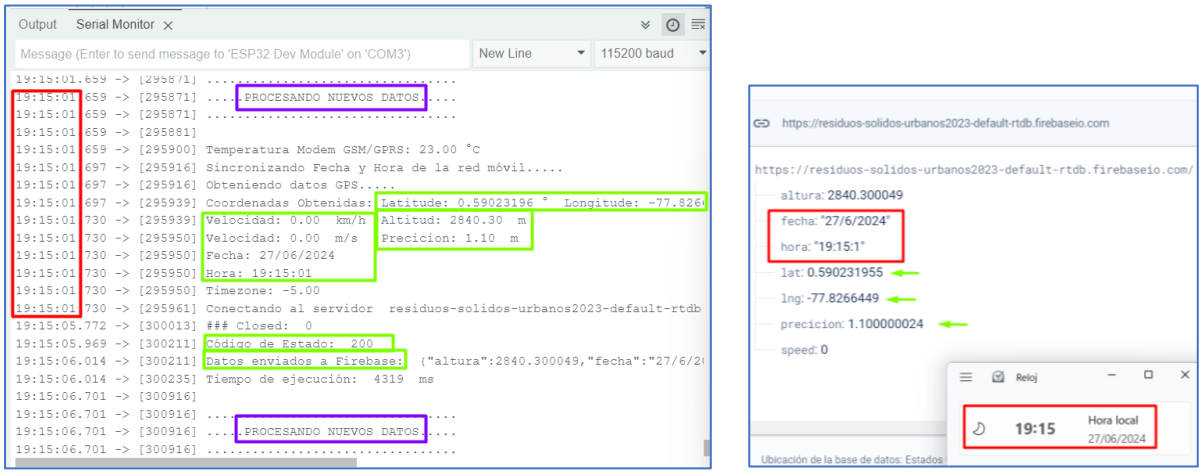
| | |
|---|---|
| #3 | 12:01:00 horas |
| **Tiempo de duración de batería promedio** | 12:02:00 horas |

### *4.1.2 Pruebas de funcionamiento del bloque de almacenamiento y procesamiento*

Tras la correcta recolección de datos en el bloque de obtención, se procede a verificar su correcta transmisión y almacenamiento hacia la base de datos en tiempo real de Firebase. La Figura 87 ilustra este proceso: los datos capturados por el nodo recolector se envían al servidor de la base de datos (Figura 87a) y, posteriormente, se visualizan en la interfaz de Realtime de Firebase(Figura 87b), por tanto, se verifica la integridad y disponibilidad inmediata de la información recopilada por el nodo recolector del sistema de recolección de RSU.

**Figura 86**

*Verificación de transmisión exitosa de datos en tiempo real del nodo recolector a la base de datos de Firebase*




**(a)** **(b)**

A igual que la prueba previa, la Figura 88 ilustra la operación eficiente entre el nodo recolector y la base de datos. En la Figura 88a, se muestra la correcta conexión al servidor y el envío de datos GPS, y la Figura 88b refleja cómo estos datos se actualizan en tiempo real en

Firebase, verificando la exactitud de la fecha y hora con el reloj del computador. Además, se constata que el tiempo de ejecución se mantiene consistentemente entre 4-5 segundos.

**Figura 87**

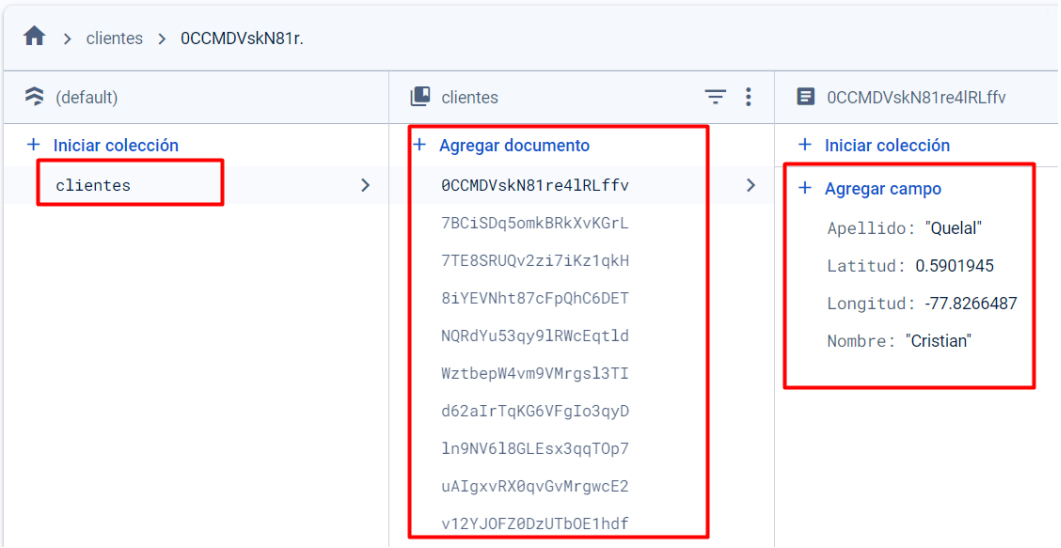
*Verificación de transmisión de datos del nodo recolector a la base de datos de Firebase*


**(a)** **(b)**

Asimismo, en el proyecto se utiliza la base de datos Cloud Firestore de Firebase para almacenar los datos de las solicitudes de los clientes. Como se muestra en la Figura 89, los datos se organizan dentro de colecciones, en este caso, se llama `clientes`. A cada cliente se le asigna un documento que contiene los campos que identifican su solicitud, incluyendo `nombre/apellido` y los datos de `longitud/latitud` de la ubicación.

**Figura 88**

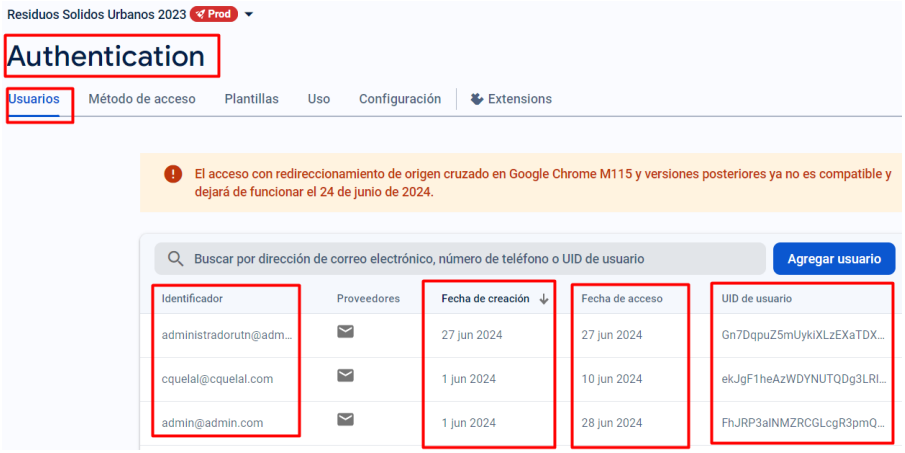
*Verificación de almacenamiento de los datos de solicitud de recolección de RSU del cliente*



Además, se utiliza el servicio de Autenticación de Firebase para gestionar el inicio de sesión de los usuarios `Recolectores`. En la Figura 90 se muestran las credenciales creadas por el administrador, donde el identificador para cada usuario es el correo electrónico, también se visualizan la fecha de creación y la fecha de último acceso de cada usuario, lo cual permite monitorear los accesos realizados. Además, se verifica que cada usuario tiene un UID único que facilita su identificación en el sistema.

**Figura 89**

*Verificación de los usuarios creados en el servicio de Autenticación de Firebase*




En la Figura 91 se muestran los mensajes de depuración que se emiten cuando se realiza un inicio de sesión exitoso mediante el servicio de Autenticación de Firebase. En este ejemplo, el usuario `admin@admin.com` inicia sesión, y el proceso de verificación se realiza con `FirebaseAuth`, utilizando el token asociado al `UID` del usuario. Una vez verificado las credenciales correctas, se muestra el mensaje de inicio de sesión exitoso para el usuario.

**Figura 90**

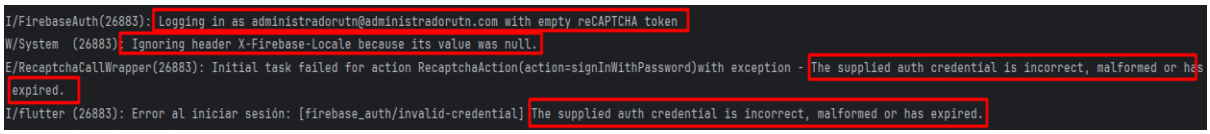
*Mensajes de depuración para el Login exitoso mediante el uso del servicio de Autenticación de Firebase*

```
I/FirebaseAuth(26883): Logging in as admin@admin.com with empty reCAPTCHA token
W/System  (26883): Ignoring header X-Firebase-Locale because its value was null.
W/System  (26883): Ignoring header X-Firebase-Locale because its value was null.
D/FirebaseAuth(26883): Notifying id token listeners about user ( FhJRP3alNMZRCGLcgR3pmQ7t9CC2 ).
I/flutter (26883): Inicio de sesión exitoso: admin@admin.com
```

Si las credenciales de inicio de sesión son incorrectas, el acceso es denegado y se muestran mensajes de depuración indicando el problema, como se ilustra en la Figura 92. El mensaje final muestra un error de inicio de sesión identificado como `firebase_auth/invalid-credential,` señalando que las credenciales pueden ser incorrectas, mal estructuradas o estar expiradas.

**Figura 91**

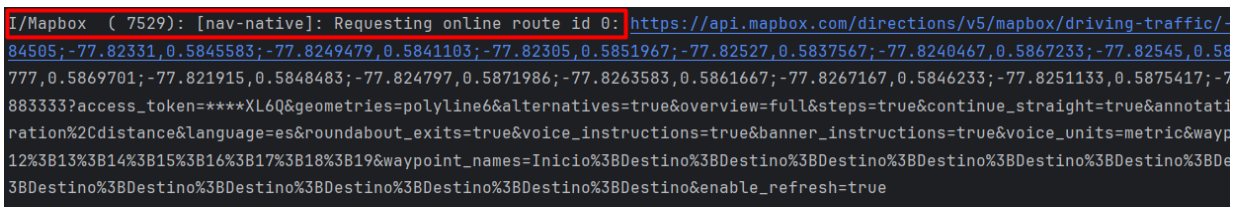
*Mensajes de depuración para el Login invalido mediante el uso del servicio de Autenticación de Firebase*




Finalmente, se verifica el correcto funcionamiento de la API de Mapbox. Esto se realiza a través de los mensajes de depuración de cuando se genera la ruta en la aplicación móvil, como se muestra en la Figura 93. Al realizar una solicitud de ruta a la API de Mapbox, se genera un `request online route id`. La solicitud se estructura según la documentación de la API, utilizando la API de direcciones y especificando que se requiere una ruta para `driving-traffic`, lo cual considera el tráfico para conducir. Además, se incluyen las ubicaciones de los puntos de la ruta y las configuraciones especificadas en la codificación de este apartado, como se indicó en el capítulo previo.

**Figura 92**

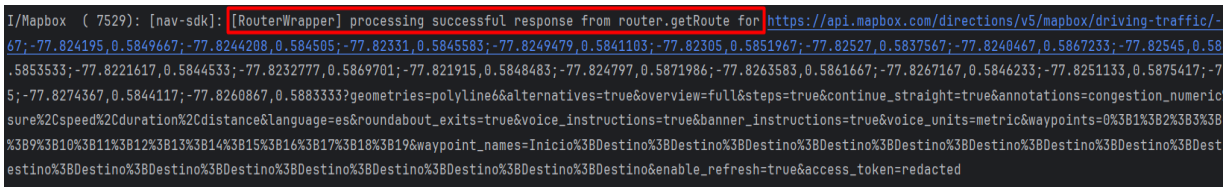
*Verificación de la depuración de mensajes de la solicitud de ruta de recolección de RSU a la API de Mapbox*



En la Figura 94, se verifica el mensaje de depuración "response", indicando el éxito en el procesamiento de la solicitud con el mensaje `processing successful response from router-getRoute.` Esto confirma que la API de Mapbox funciona correctamente para cada solicitud de ruta. De la misma forma, se realizan solicitudes a la API de Mapbox cuando se necesitan mapas estáticos u otros datos importantes para el sistema de recolección de RSU.

**Figura 93**

*Verificación de la depuración de mensajes sobre el procesamiento exitoso y obtención de la ruta de recolección de RSU desde la API de Mapbox*



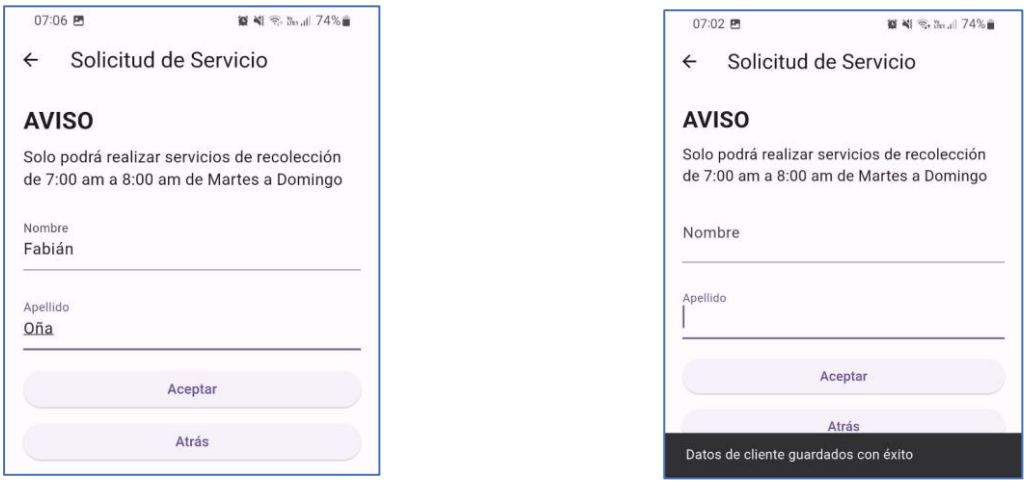

### *4.1.3 Pruebas de funcionamiento del bloque de visualización de datos*

Para asegurar el correcto funcionamiento del bloque de visualización, se realizan pruebas de todas las funciones disponibles en la aplicación móvil desarrollada.

La primera prueba implica verificar el envío de solicitudes de recolección por parte de los clientes. Como se muestra en la Figura 95, los datos del cliente se ingresan y, si coincide con el horario indicado en el aviso, el botón `Aceptar` estará habilitado (Figura 95a). Al presionar el botón, la ubicación del cliente se captura automáticamente del dispositivo que este usando y, si el proceso es exitoso, se muestra una barra inferior indicando que los datos se han guardado correctamente (Figura 95b).

**Figura 94**

*Verificación de envio de solicitud de recolección de cliente*

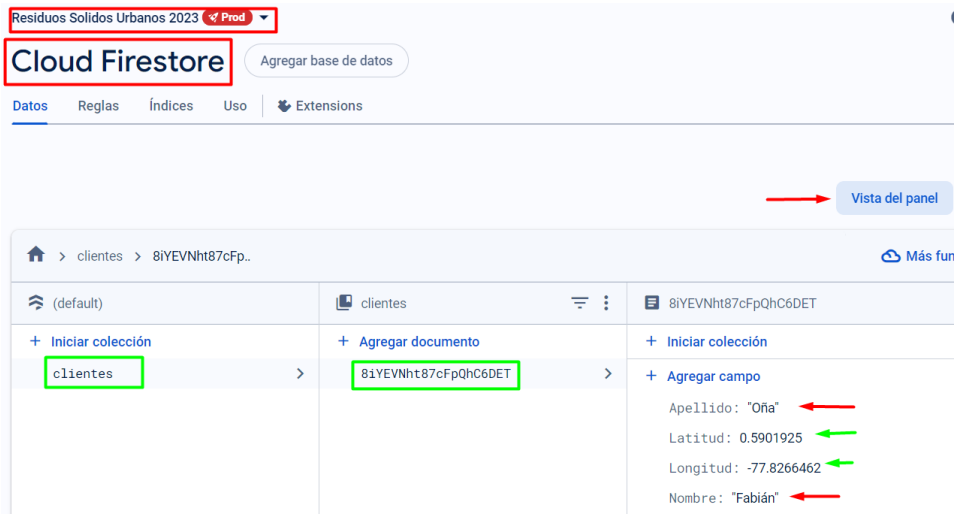



**(a)** **(b)**

En la Figura 96, se confirma que la solicitud generada previamente se ha almacenado en la base de datos para su posterior procesamiento. Los datos de `nombre/apellido` y `longitud/latitud` están disponibles y listos para ser utilizados en la aplicación móvil para crear un punto de recolección.

**Figura 95**

*Verificación de almacenamiento de la solicitud de recolección de RSU en la base de datos*

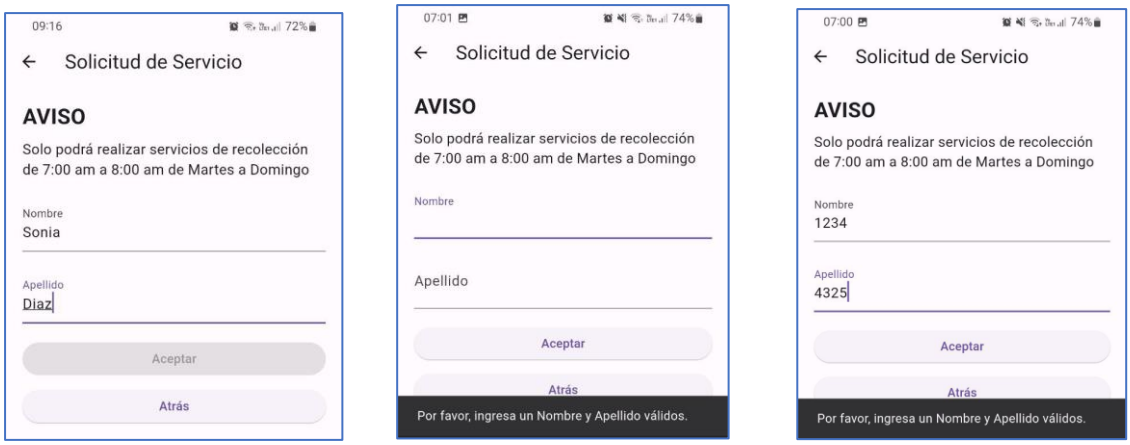



Para asegurar el éxito del proceso anterior, como se muestra en la Figura 97, es crucial considerar varias condiciones. Primero, ingresar en el horario adecuado según lo indicado en

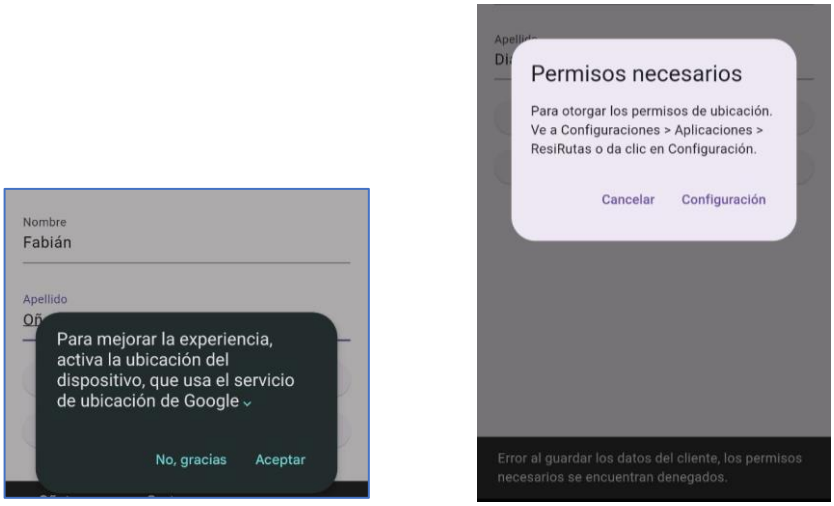
el `Aviso`; de lo contrario, el botón `Aceptar` estará deshabilitado (Figura 97a). Además, los datos ingresados en los campos de Nombre y Apellido no pueden estar vacíos (Figura 97b) ni contener caracteres no válidos (Figura 97c), de lo contrario se muestra una barra inferior que notifica que se deben ingresar parámetros válidos. También es necesario que el servicio de ubicación del dispositivo esté activado; de lo contrario, se mostrará una notificación para activarlo (Figura 97d). En algunos casos, la aplicación podría tener denegados los permisos de ubicación, los cuales deben ser concedidos siguiendo las indicaciones mostradas en un cuadro de dialogo que se emite (Figura 97e).

**Figura 96**

*Verificación de excepciones de error del envio de la solicitud de recolección de RSU*




**(a)** **(b)** **(c)**

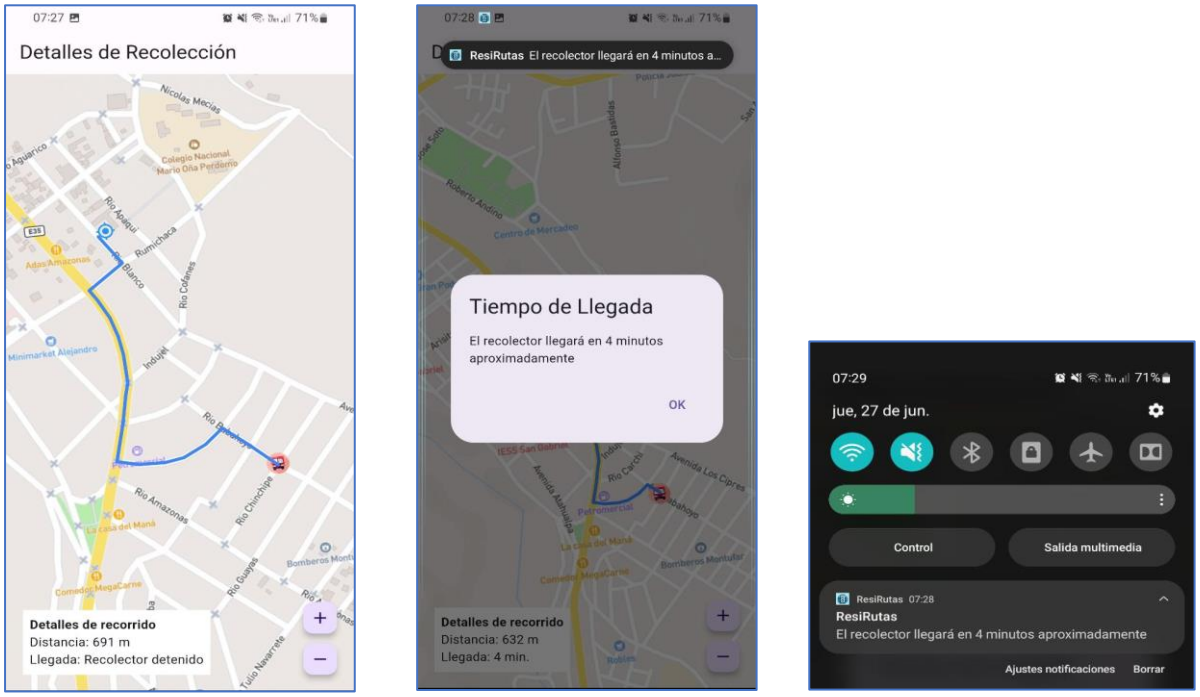



**(d)** **(e)**

Después de que el cliente envía la solicitud de recolección, accede a la interfaz de seguimiento del proceso(Figura 98a), donde puede ver la ubicación actual del recolector y recibir notificaciones tanto en cuadros emergentes(Figura 98b) como push(Figura 98c). Además, se visualiza la distancia en metros entre el cliente y el recolector, así como el tiempo de llegada actualizado en tiempo real. Cuando el recolector detiene su marcha, se muestra `Recolector Detenido`, y si el tiempo de espera es inferior a un minuto, se indica `<1 min`. Esto proporciona información clara al cliente sobre el progreso de su recolección. Es crucial mantener esta página activa en primer o segundo plano para recibir las notificaciones; de lo contrario, no se recibirán notificaciones.

**Figura 97**

*Interfaz de visualización de información y notificaciones sobre el proceso de recolección para cada cliente*



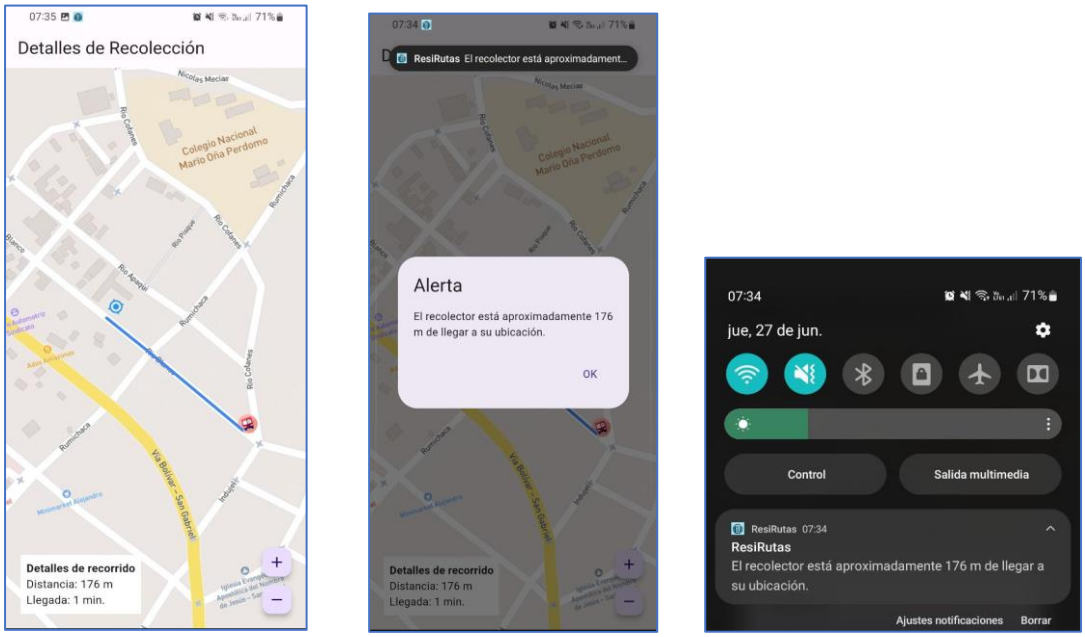

**(a)** **(b)** **(c)**

Durante el proceso de recolección, el cliente puede realizar un seguimiento continuo y será notificado cuando el recolector esté a menos de 200 metros de distancia(Figura 99a).

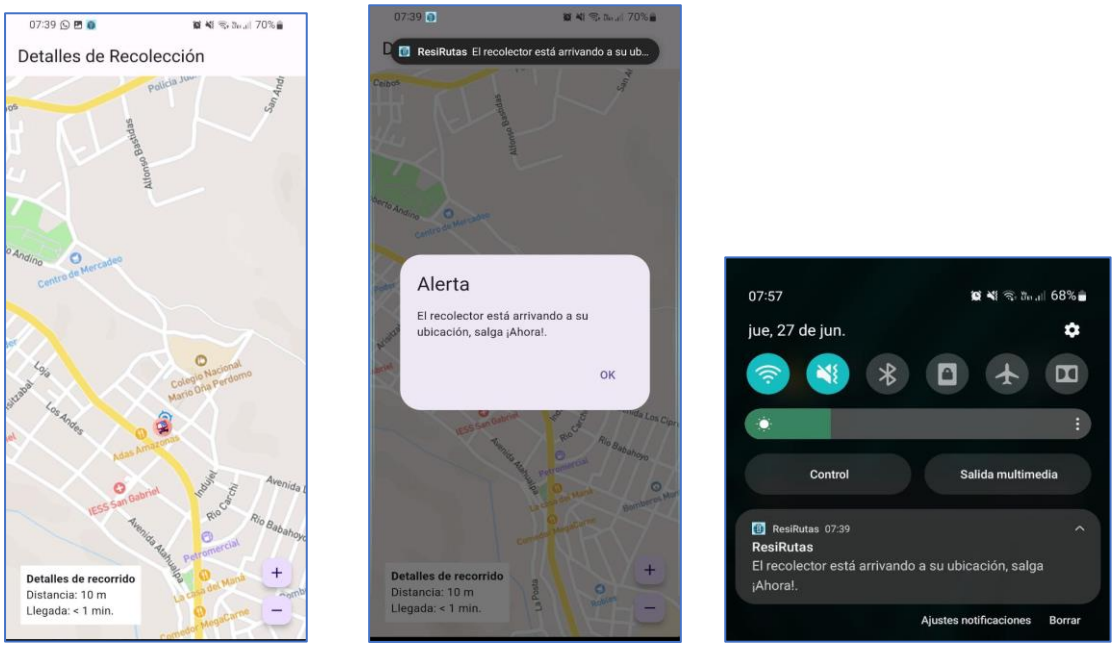
Además, recibirá una notificación cuando el recolector esté llegando al punto de recolección, que se ha establecido (menos de 10 metros de distancia) (Figura 99b).

**Figura 98**

*Interfaz de información y notificaciones sobre el proceso de recolección para los clientes*




**(a)**



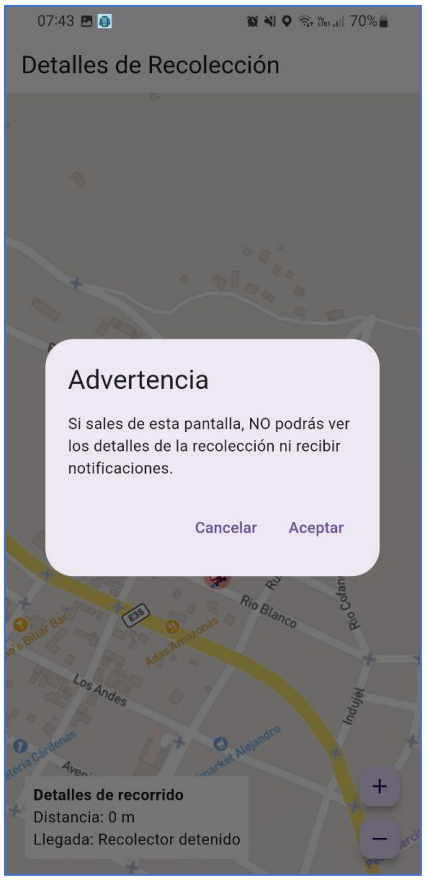

**(b)**

Como se mencionó anteriormente, es importante que los clientes no abandonen el proceso de recolección. Si cierran la aplicación o regresan a la interfaz anterior, no podrán

recibir notificaciones importantes sobre el estado de su solicitud de recolección. La interfaz del proceso de recolección debe mantenerse en primer o segundo plano para garantizar que las notificaciones sean recibidas y permitir la visualización del progreso de la recolección. En la Figura 100 se muestra la generación de un mensaje emergente que advierte al usuario sobre la pérdida de detalles de recolección y la incapacidad de recibir notificaciones si intenta abandonar la interfaz del seguimiento. El mensaje ofrece las opciones de cancelar o aceptar; aceptar cerrará la interfaz de seguimiento, mientras que cancelar mantendrá al usuario en la misma interfaz.

**Figura 99**

*Verificación de generación de ventaja emergente para la advertencia de abandono de interfaz de seguimiento del proceso de recolección*

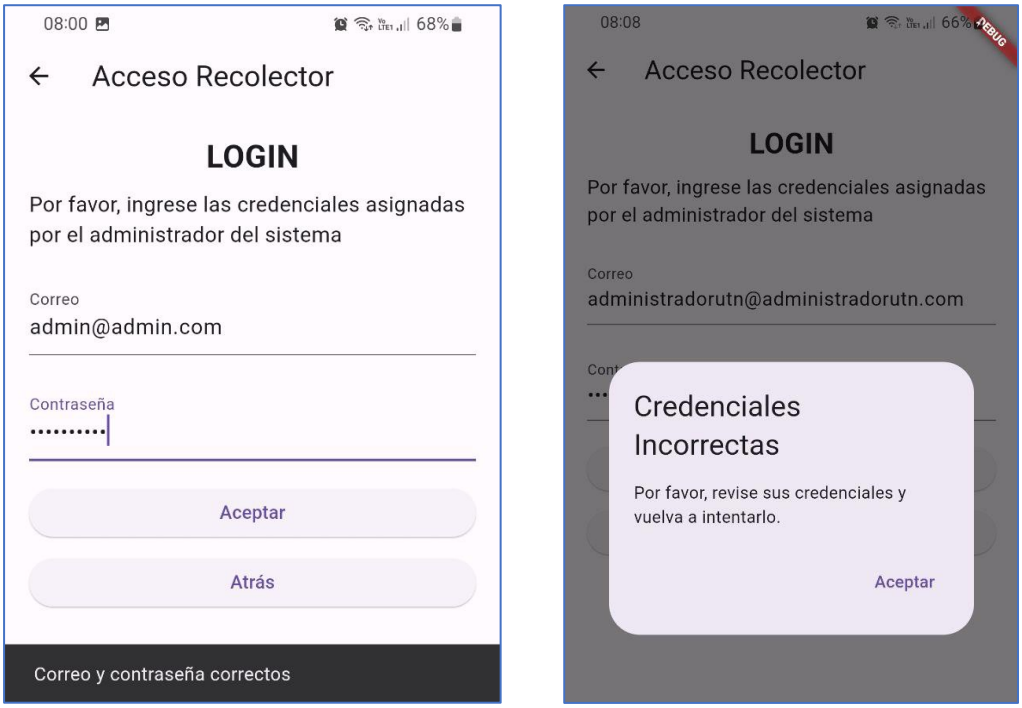



Una vez verificadas el funcionamiento correcto de las características del bloque de visualización para el cliente, se realizan las pruebas para las características del `Recolector`.

La primera prueba consiste en verificar la autenticación del usuario `Recolector`, como se evidencia en la Figura 101, si la autenticación es exitosa, se muestra una notificación en la barra inferior, confirmando las credenciales correctas(Figura 101a); por otro lado, si la autenticación es incorrecta se muestra un mensaje informativo(101b) para que verifique las credenciales. Este paso es importante para garantizar seguridad en el sistema.

**Figura 100**

*Verificación de la Autenticación del usuario "Recolector" en la aplicación móvil*

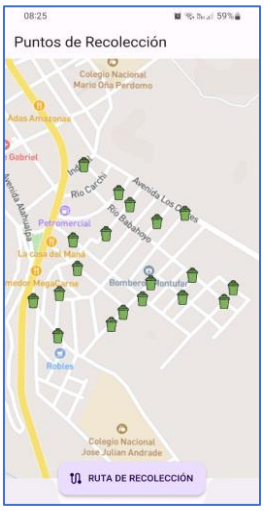



**(a)** **(b)**

Una vez iniciado sesión correctamente se accede a la interfaz que muestra todos los puntos de recolección. Desde aquí, el recolector puede verificar que existan los puntos de recolección suficientes(al menos 1) antes de generar la ruta de recolección, como se muestra en la Figura 102. Para generar la ruta de recolección, debe dirigirse a la siguiente interfaz utilizando el botón `Ruta de Recolección`.

**Figura 101**

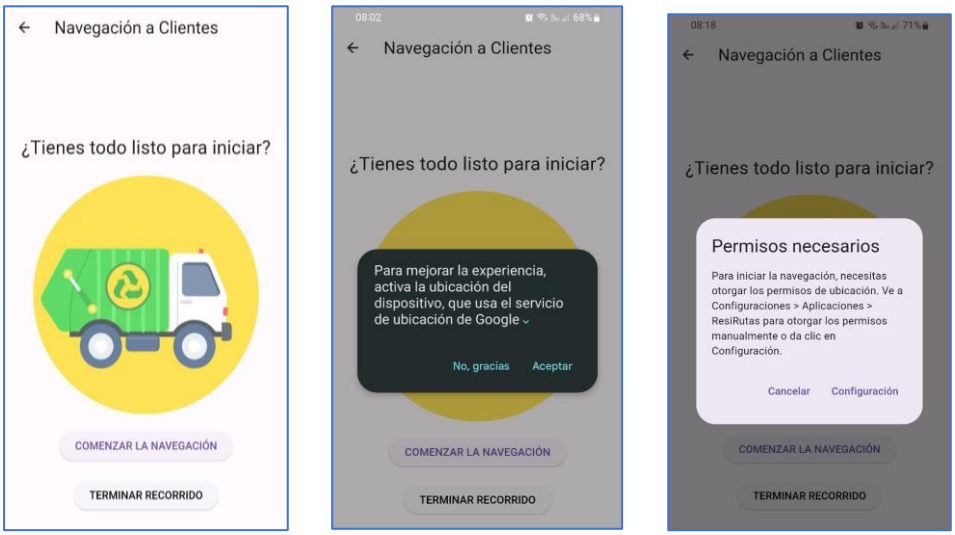
*Interfaz de visualización de puntos de recolección*




En la Figura 103a, se muestra la interfaz donde se puede generar la ruta y comenzar el recorrido. Al seleccionar el botón de inicio de navegación, si el servicio de ubicación del dispositivo móvil del recolector no está activado, se recibirá una notificación indicando que es necesario activarlo (Figura 103b). Del mismo modo, si la aplicación no tiene concedidos los permisos de ubicación, se abrirá una ventana emergente con las instrucciones necesarias para activar estos permisos(Figura 103c).

**Figura 102**

*Interfaz con los botones para generar la ruta de recolección e iniciar la navegación*

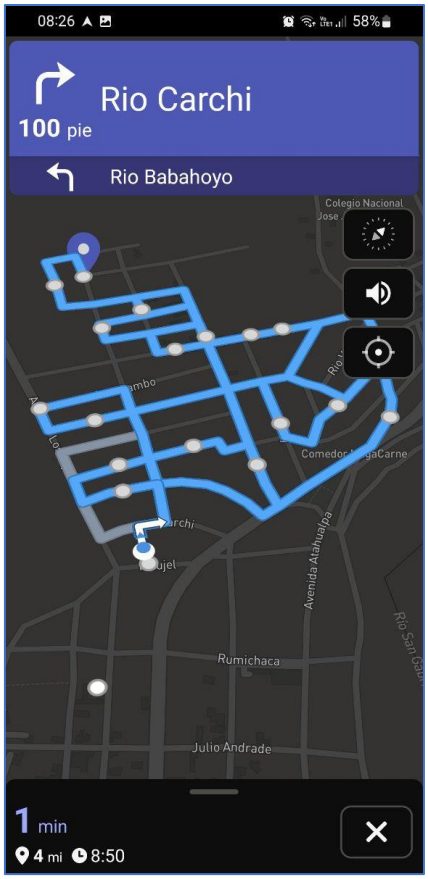



**(a)** **(b)** **(c)**

Una vez que todo esté activado correctamente y se presione el botón `COMENZAR LA NAVEGACIÓN`, se mostrará la ruta de recolección generada por la API de Mapbox, teniendo en cuenta las condiciones de tráfico del sector. En la Figura 104, se observa una ruta generada basada en puntos de recolección simulados, proporcionando indicaciones visuales y por voz para guiar al recolector a lo largo del proceso de recolección.

**Figura 103**

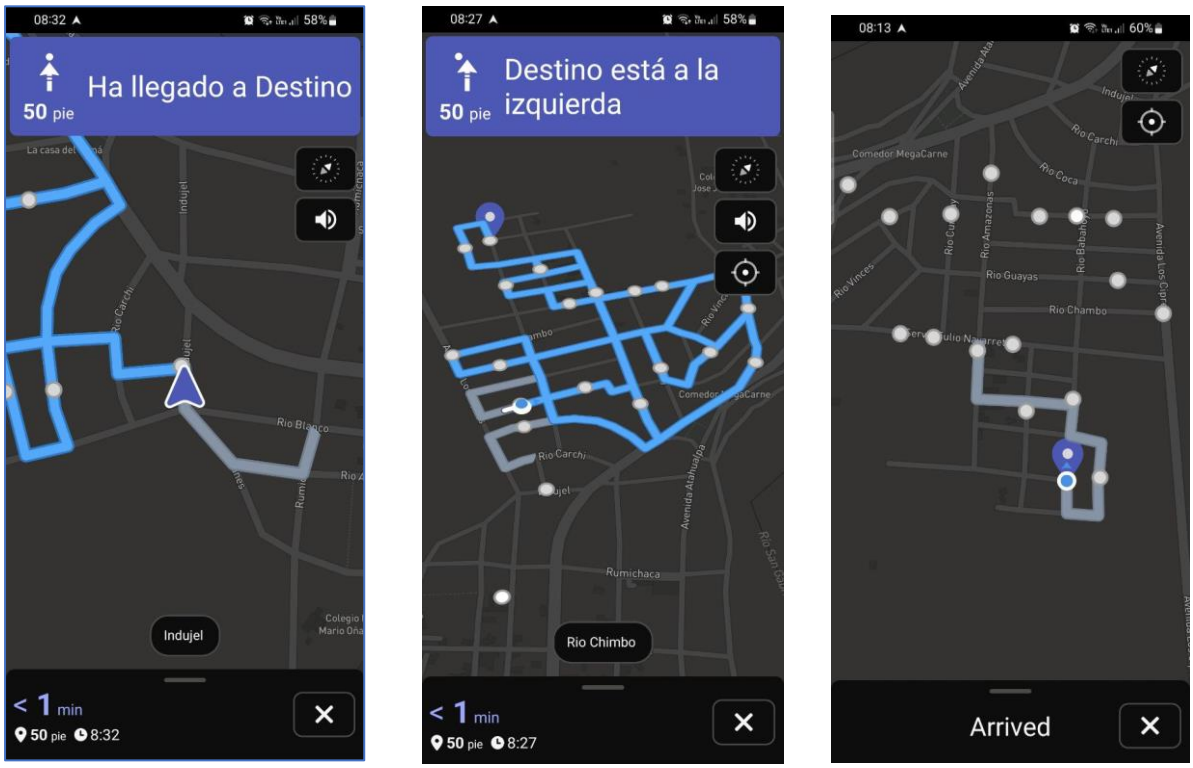
*Ruta de recolección generada con indicaciones visuales y voz para el recolector*




Un aspecto relevante de utilizar la API de Mapbox, como se verifica en la Figura 105, es su SDK de Direcciones, que permite utilizar la función de navegación. Esta función proporciona guía visual y por voz, además de notificaciones al llegar a un punto de recolección designado como destino por la API(Figura 105a). De igual forma, esta genera rutas alternativas en color gris que pueden seguirse(Figura 105b). Finalmente, se notifica en la interfaz cuando se completa la ruta planificada en su totalidad(Figura 105c).

**Figura 104**

*Generación de la ruta de navegación para el proceso de recolección de RSU*




**a)** **b)** **c)**

Una vez completada la ruta, el recolector puede finalizar el recorrido en la interfaz de la Figura 103, utilizando el botón `Terminar recorrido` como se visualiza en la Figura 106. Al presionar el botón para terminar el recorrido, aparecerá una ventana emergente para confirmar el proceso (Figura 106a). Una vez confirmado, este se ejecutará y se verificará su correcta realización mediante una notificación en una barra emergente inferior con el mensaje `Recorrido terminado` (Figura 106b). De esta manera, el sistema estará preparado para el día siguiente, cuando se deberá repetir el proceso descrito tanto para el cliente como para el recolector.

**Figura 105**

*Verificación de proceso de botón Terminar Recorrido*

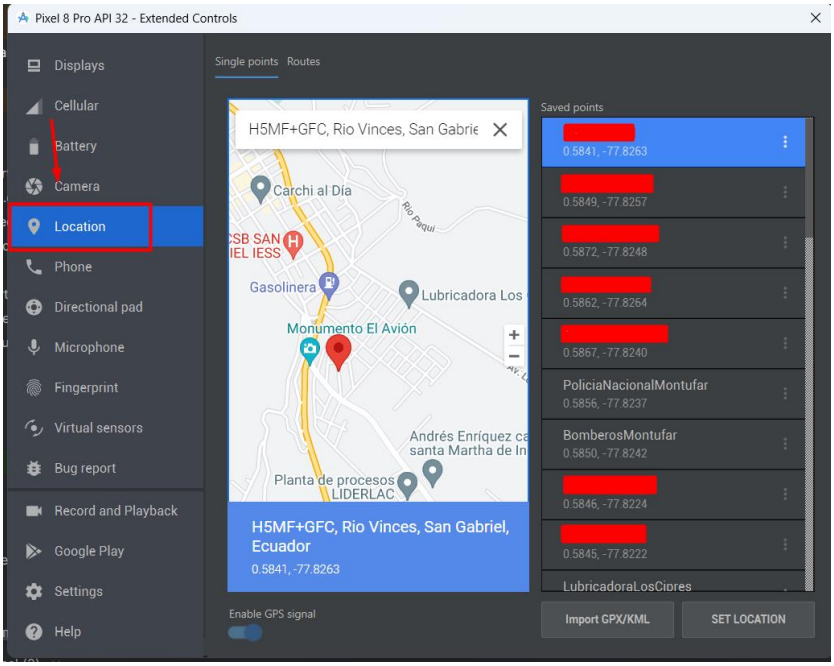

**(a)** **(b)**

### *4.1.4 Pruebas de funcionamiento del sistema de recolección de RSU con solicitudes de recolección simuladas*

Para llevar a cabo esta prueba, es necesario simular solicitudes de recolección. Utilizando Android Studio, se cambia la ubicación del dispositivo móvil emulado, creando así diferentes ubicaciones para distintos usuarios. Como se muestra en la Figura 107, se crean diferentes ubicaciones basándose en la información de algunos habitantes del sector. Por motivos de privacidad, los nombres de los usuarios están ocultos.

**Figura 106**

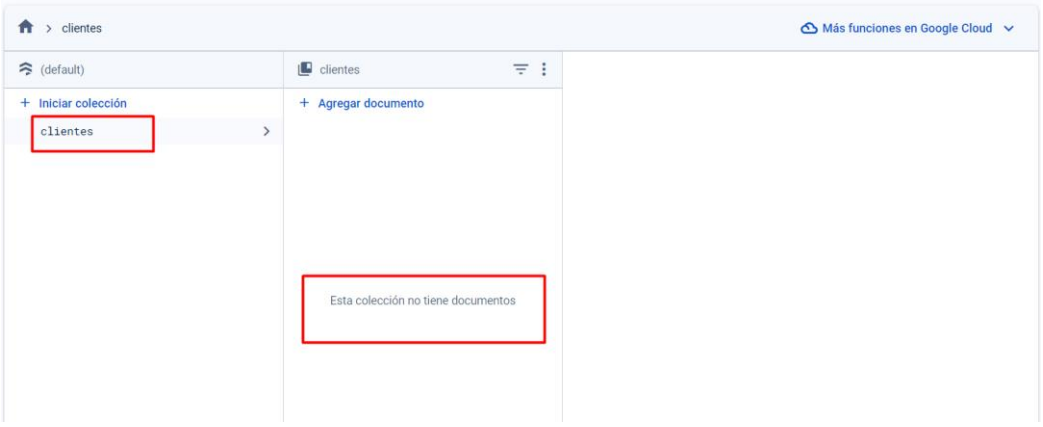
*Creación de ubicaciones de clientes en Android Studio para simular solicitudes de recolección*




Para iniciar la verificación del funcionamiento del Sistema de recolección de RSU con solicitudes de recolección simuladas, primero se confirma que no hay solicitudes de recolección existentes. Como se muestra en la Figura 108, la base de datos de Firestore no contiene información de clientes, ya que aún no se ha realizado ninguna solicitud de recolección.

**Figura 107**

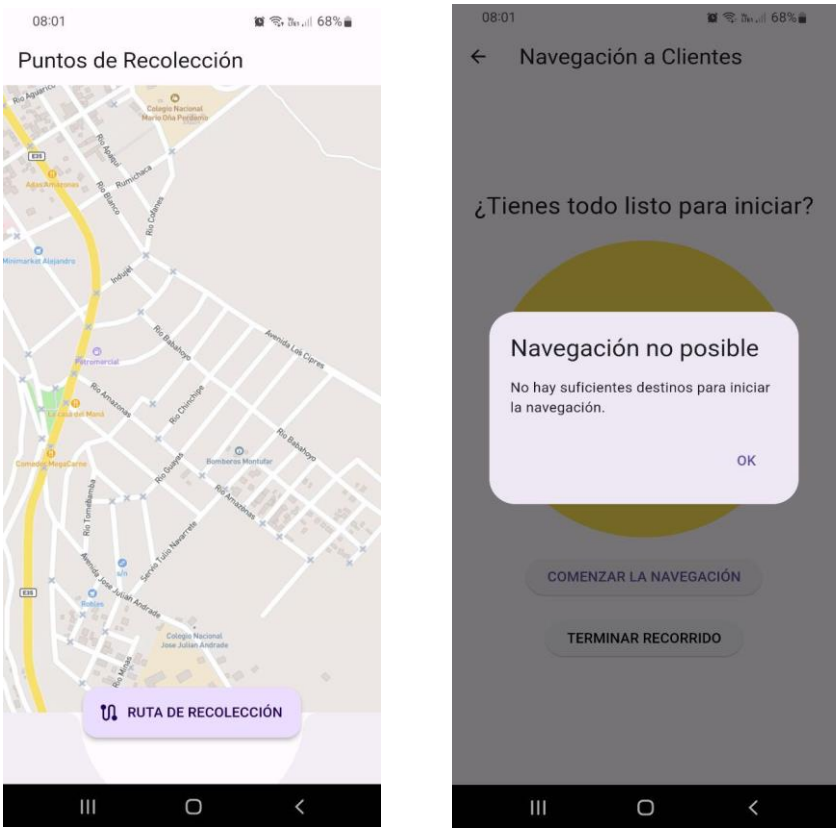
*Verificación de Firestore Database sin información de solicitudes de clientes*



De igual forma en la Figura 109, se verifica que al ingresar al apartado del recolector, no se muestra ningún puntos de recolección (Figura 109a). Si se intenta redirigir a la interfaz para generar la ruta y se presiona el botón de comenzar navegación, aparecerá un cuadro emergente notificando que la navegación no es posible porque no hay suficientes (al menos uno) destinos para iniciar la navegación (Figura 109b).

**Figura 108**

*Verificación de ausencia de puntos de recolección y notificación de falta de destinos para navegación*

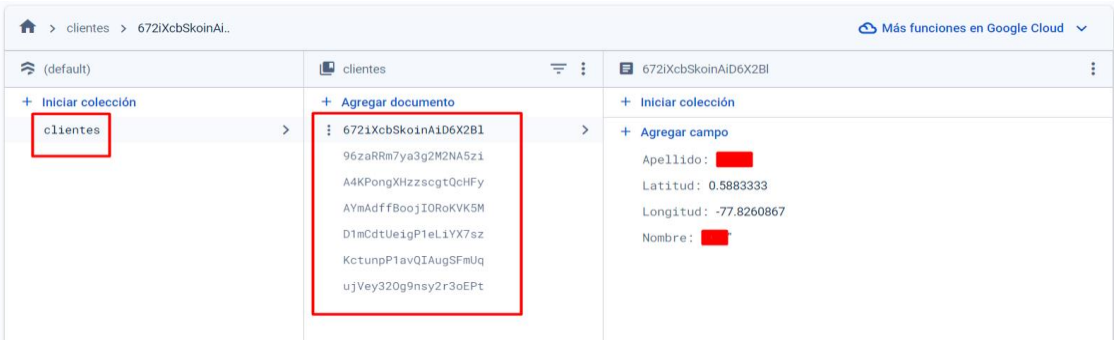



**(a)** **(b)**

Por tanto, para la primera evaluación del Sistema de recolección de RSU, se generan solicitudes de recolección simuladas, en este caso 7 solicitudes de recolección, de modo que, se crean 7 documentos en la colección `clientes` de la Firestore Database, como se muestra en la Figura 110. De esta forma, se confirma la existencia de 7 solicitudes de recolección, permitiendo así la generación de una ruta de recolección.

**Figura 109**

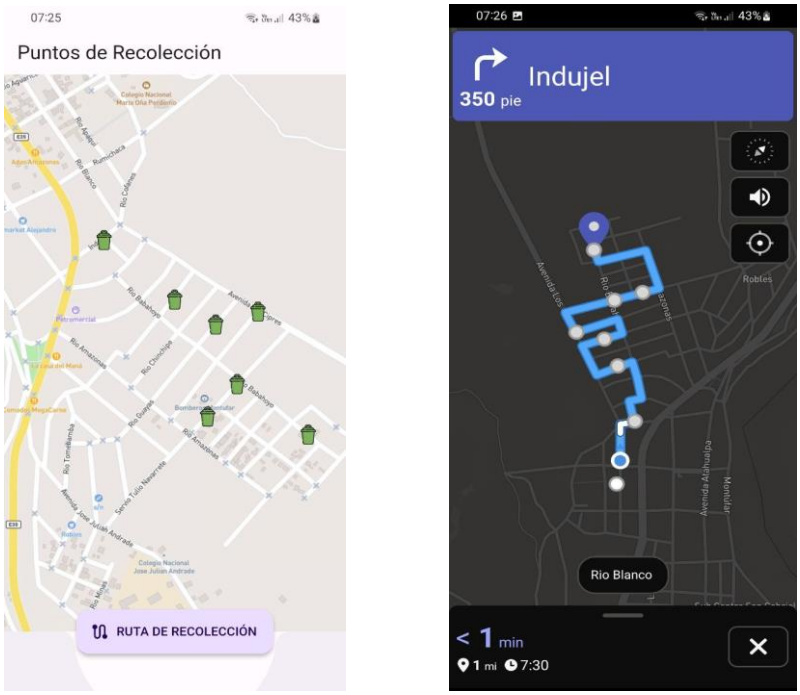
*Verificación de la creación de documentos en la colección clientes para solicitudes de recolección de la primera evaluación del Sistema de recolección de RSU.*




En la Figura 111 se verifica que, al ingresar al apartado del recolector, se muestran los 7 puntos de recolección (Figura 111a). Luego al generar la ruta de recolección mediante el botón `Comenzar la Navegación`, se visualiza la ruta con las indicaciones visuales y por voz (Figura 111b).

**Figura 110**

*Verificación de primera prueba de visualización de puntos de recolección simulados y generación de ruta de recolección con indicaciones para conductor*

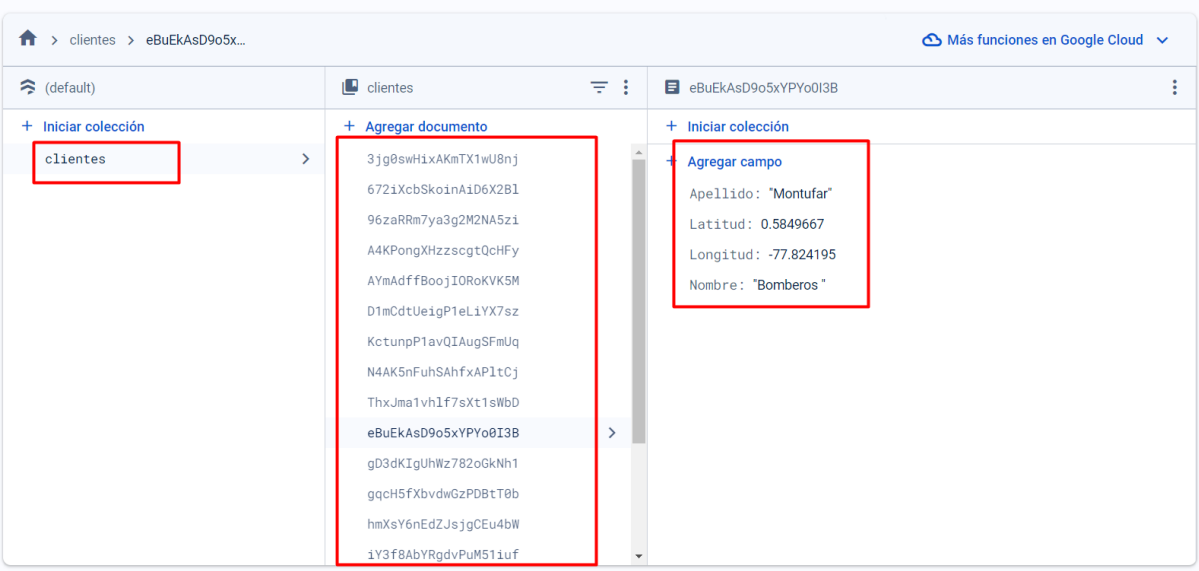



**(a)** **(b)**

Luego, para la segunda evaluación del Sistema de recolección de RSU, se generan solicitudes de recolección simuladas, en este caso 19 solicitudes de recolección, por tanto, se crean 19 documentos en la colección `clientes` de la Firestore Database, como se muestra en la Figura 112. De esta forma, se confirma la existencia de 19 solicitudes de recolección, permitiendo así la generación de una ruta de recolección.

**Figura 111**

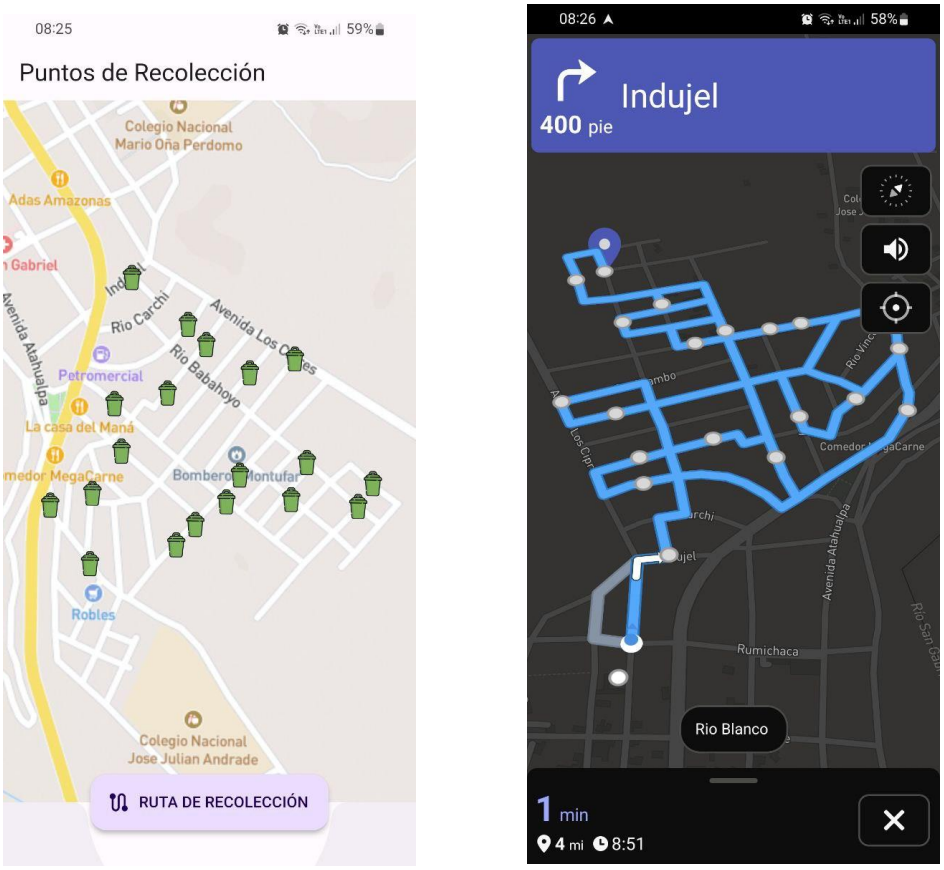
*Verificación de la creación de documentos en la colección "clientes" para solicitudes de recolección de la segunda evaluación del Sistema de recolección de RSU.*




En la Figura 113 se verifica que, al ingresar al apartado del recolector, se muestran los 19 puntos de recolección (Figura 113a) extraídos desde la base de datos de Firebase. Luego al generar la ruta de recolección mediante el botón `COMENZAR LA NAVEGACIÓN`, se visualiza que se genera una ruta más larga en comparación con la anterior, y que igualmente contiene las indicaciones visuales y por voz (Figura 113b).

**Figura 112**

*Verificación de segunda prueba de visualización de puntos de recolección simulados y generación de ruta de recolección con indicaciones para conductor*

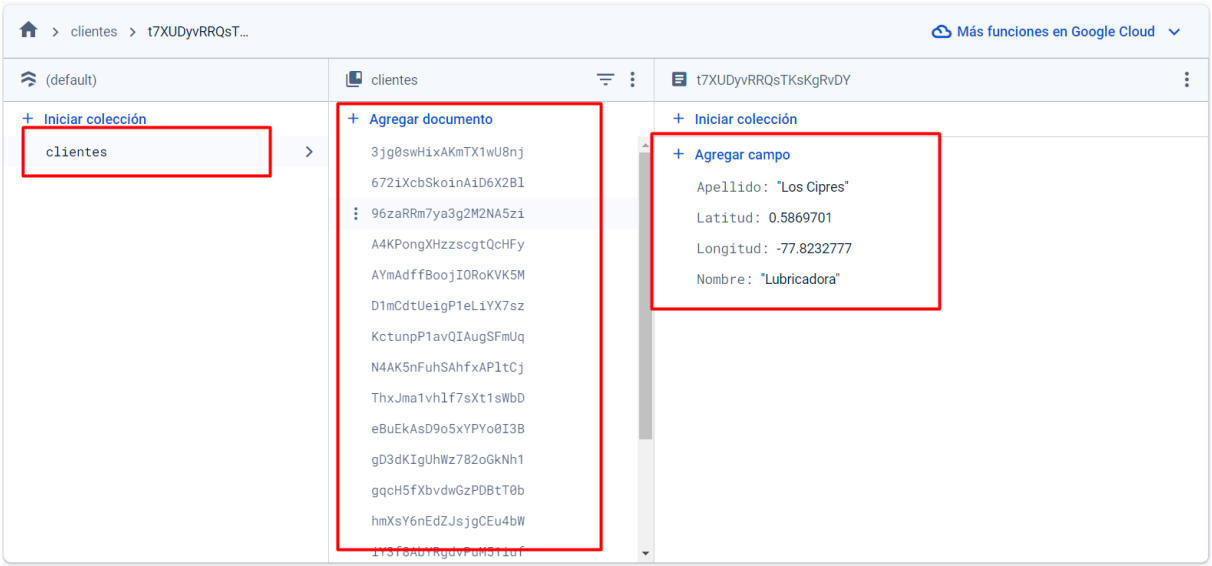



**(a)** **(b)**

Finalmente, se realiza la tercera y última evaluación en la que se crean 22 solicitudes de recolección. Siguiendo el procedimiento, se generan 22 documentos en la colección `clientes` en Firestore como se muestra en la Figura 114. Luego a partir de la información de ubicación de la solicitud de cada cliente se crean los puntos de recolección.

**Figura 113**

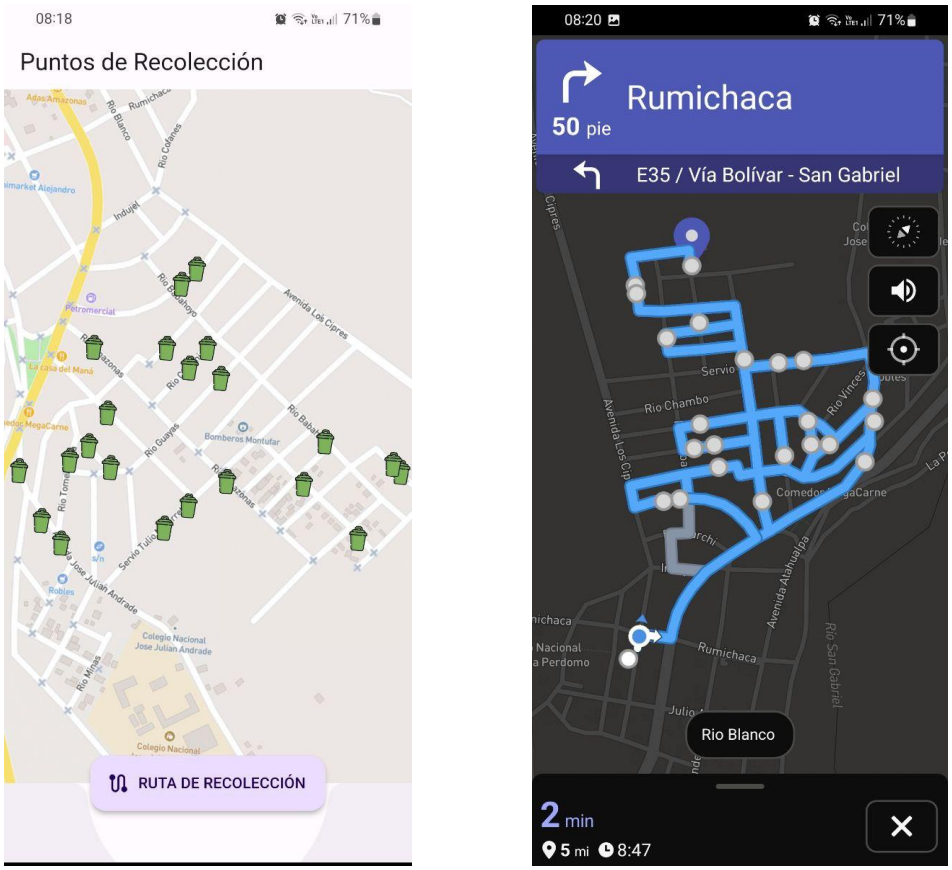
*Verificación de la creación de documentos en la colección clientes para solicitudes de recolección de la tercera evaluación del Sistema de recolección de RSU.*




Por lo tanto, siguiendo el mismo proceso de las dos pruebas anteriores, se verifica en la Figura 115 que al acceder al apartado del recolector, se muestran los 22 puntos de recolección extraídos de la base de datos de Firebase (Figura 115a). Posterior, al generar la ruta de recolección y comenzar la navegación, se observa que la ruta generada es más extensa que la anterior, incluye las indicaciones visuales y por voz, además de posibles rutas alternas en color plomo(Figura 115b).

**Figura 114**

*Verificación de tercera prueba de visualización de puntos de recolección simulados y generación de ruta de recolección con indicaciones para conductor*




**(a)** **(b)**

## 4.2 Resultados de Pruebas

La culminación exitosa de las pruebas confirma el funcionamiento óptimo del sistema de recolección de Residuos Sólidos Urbanos (RSU) en el barrio Centenario de San Gabriel.

Se constata que, la batería alcanza una duración de aproximadamente 12 horas, adecuada para la operación prevista con rutas optimizadas. El sistema de posicionamiento global (GPS) del nodo recolector muestra una precisión aceptable, estimada entre 1 y 2 metros en condiciones ideales, aunque según el fabricante, factores ambientales(lluvia, neblina, etc) pueden afectar esta precisión. Por otra parte, la API de Mapbox, en su versión gratuita, se verifica que permite generar rutas con un máximo de 25 puntos de recolección. Para una

implementación a mayor escala, se recomienda considerar un plan empresarial de Mapbox, que ofrecería mayor capacidad y beneficios adicionales, aunque tendría un costo dependiendo de las capacidades solicitadas.

La prueba de generación de solicitudes de recolección reveló que estas deben crearse en el punto exacto de recogida de residuos, preferiblemente en la calle o frente a la casa. Si se genera dentro del domicilio, el sistema de navegación no reconocerá el destino como alcanzado, lo que puede resultar en múltiples visitas innecesarias al mismo punto. Este problema se identificó al permitir que la ubicación se capture directamente desde el dispositivo móvil del cliente.

Para solucionar esto, se propone implementar una función que permita al cliente seleccionar manualmente la ubicación de recolección en un mapa interactivo. De esta manera, incluso si el cliente está dentro de su casa, podrá buscar y marcar con precisión el punto de recolección deseado, preferiblemente en la calle frente a su domicilio. Esta mejora permitiría una mayor precisión en la ubicación de recolección y optimizaría el funcionamiento del sistema de navegación en tiempo real.

En la Tabla 31, se presenta un resumen de los resultados de las pruebas realizadas. Esta tabla proporciona una visión general de los cuatro tipos de pruebas ejecutadas, detallando la duración de cada una y las observaciones más relevantes. Estos resultados son fundamentales para comprender las limitaciones del sistema e identificar áreas potenciales de mejora.

**Tabla 31**

*Resultados de las pruebas del Sistema de recolección de RSU*

| Tipo de Prueba | Duración | Observaciones |
|---|---|---|
| **Prueba 1**<br>Funcionamiento del bloque de obtención de datos | 7 días | -La batería demostró una duración adecuada de 12 horas, suficiente para la operación diaria prevista. |

| | | |
|---|---|---|
| | | -La precisión del GPS (1-2 metros) puede variar debido a condiciones climáticas. |
| **Prueba 2**<br>Funcionamiento del bloque de almacenamiento y procesamiento | 7 días | La API de Mapbox en versión gratuita limita a 25 puntos de recolección por ruta. Se sugiere la contratación de un plan empresarial para mayor escala. |
| **Prueba 3**<br>Funcionamiento del bloque de visualización de datos | 8 días | No existen observaciones, las características funcionan adecuadamente. |
| **Prueba 4**<br>Funcionamiento del sistema de recolección de RSU con solicitudes de recolección simuladas | 16 días | Se observa que, El problema de generación de solicitudes dentro del domicilio causa que el sistema de navegación no reconozca el punto como alcanzado. Esto puede resultar en múltiples visitas innecesarias al mismo lugar, afectando la eficiencia de la ruta de recolección. Además, se visualiza que entre más |

| | |
|---|---|
| | puntos de recolección exista, se generan más rutas alternativas. |

Para garantizar una implementación y utilización óptima, se desarrollan recursos adicionales, descritos en el Anexo 6, el cual contiene el Plan de Implementación, que incluye el cronograma para la instalación del hardware, configuración del software, capacitación del personal, recomendaciones técnicas y manuales detallados para administradores, clientes y recolectores. La adherencia a este plan es fundamental para asegurar una integración fluida del sistema de recolección de RSU en el barrio Centenario.

## 4.3 Análisis Costo/Beneficio del Sistema de recolección de RSU

Una vez verificado el funcionamiento del sistema de recolección de RSU a través de diversas pruebas exhaustivas, es fundamental evaluar su viabilidad técnica/económica y su impacto en la comunidad. El análisis costo/beneficio que se presenta a continuación ofrece una perspectiva sobre la inversión requerida para implementar y mantener el sistema, así como los beneficios que se esperan obtener. La evaluación permite comprender el valor real del sistema y su potencial para mejorar la eficiencia de los servicios de recolección de RSU a largo plazo.

### *4.3.1 Costos de implementación*

El sistema de recolección de RSU propuesto para el barrio Centenario de San Gabriel requiere una inversión inicial en hardware y software, como se detalla en la Tabla 32, en donde se presenta un desglose detallado de los costos asociados con la implementación del sistema de recolección de Residuos Sólidos Urbanos (RSU) en el barrio Centenario de San Gabriel. El costo total de implementación asciende a $1583.00, incluyendo un costo anual de $123.00 por el plan de datos. Este presupuesto se divide en tres categorías principales: costos de hardware, costos de software y costos del plan de datos.

Aunque en la Tabla 32 se indican como gratuitos los servicios de Mapbox y Firebase, es importante tener en cuenta que estos planes tienen límites establecidos. Una vez superados, generan costos adicionales.

En el caso de Mapbox, ofrece un plan gratuito con 25,000 cargas de mapas móviles y 100,000 solicitudes de dirección por mes. Al exceder estos límites, se cobra $4.00 por cada 1,000 cargas de mapa adicionales y $2.00 por cada 1,000 solicitudes de dirección extras. Para usos más intensivos, Mapbox dispone de planes personalizados con un costo anual mínimo de $60,000.

Por su parte, proporciona un plan gratuito con límites específicos para cada servicio. El servicio de autenticación permite 50,000 usuarios mensuales, mientras que Cloud Firestore ofrece 1 GiB de almacenamiento total y 20,000 escrituras diarias. En cuanto a Realtime Database, el plan gratuito incluye 100 conexiones simultáneas, 1 GB de almacenamiento y 10 GB de descargas por mes. Al superar estos límites, se activa el plan Blaze, donde se paga por el consumo real de cada servicio.

En el plan Blaze de Firebase, los costos varían según la región y el tipo de operación. En Ecuador, Cloud Firestore cobra aproximadamente $0.18 por 100,000 escrituras y $0.06 por 100,000 lecturas. Por su parte, Realtime Database cobra alrededor de $0.026 por GiB al mes en almacenamiento y $0.12 por GiB adicional en descargas.

Es fundamental considerar estos costos en la planificación a largo plazo del proyecto, ya que podrían representar gastos significativos a medida que la aplicación crece y aumenta su uso de estos servicios.

**Tabla 32**

*Costos de Hardware y Software del Sistema de recolección de RSU*

| Ítem | Cantidad | Costo Unitario ($) | Costo Total($) |
|---|---|---|---|
| **Costos de Hardware** | | | |
| Módulo Arduino T-SIM7600NA | 1 | 125.00 | 125.00 |
| Batería ICR 18650 2600 mAh | 2 | 7.50 | 15.00 |
| Shield | 1 | 20.00 | 20.00 |
| Dispositivos móviles para recolectores | 1 | 300.00 | 300.00 |
| **Costos de Software** | | | |
| API de Mapas (Mapbox) | 1 | 0(plan gratuito con restricciones) | 0(plan gratuito con restricciones) |
| Servicios de base de datos (Firebase) | 1 | 0(plan gratuito) | 0(plan gratuito) |
| Desarrollo de la aplicación móvil | 1 | 1000 | 1000.00 |
| **Costos Anual de Plan de datos** | | | |
| Plan de Conexión 10 | 1 | 10.25 | 123.00 |
| | | **Costo total de implementación:** | 1583.00 |

De igual forma, la Tabla 33 desglosa los costos de capacitación del sistema de recolección de RSU para el barrio Centenario. La capacitación, que suma $1000.00, es una inversión única.

**Tabla 33**

*Costos de Capacitación del sistema de recolección de RSU*

| Ítem | Costo ($) |
|---|---|
| Capacitación administrador sistema GADM Montufar | 300.00 |
| Capacitación de conductores de los recolectores | 200.00 |
| Capacitación a moradores del barrio Centenario | 500.00 |
| **Total:** | **1000.00** |

En conjunto con la Tabla 32, que detalla los costos de implementación, se calcula una inversión inicial total de $2583.00, esta suma engloba los costos de hardware, software, plan de datos y capacitación. Además, para el mantenimiento del sistema se estima un costo anual

de $500.00, esto considera soporte técnico y actualizaciones necesarias del sistema de recolección de RSU, por tanto, el costo total para el primer año sería de $3083.00.

### *4.3.2 Beneficios esperados de la implementación del Sistema de recolección de RSU*

La implementación del sistema de recolección de RSU en el barrio Centenario de San Gabriel promete una serie de beneficios que abarcan aspectos sociales, económicos, ambientales y tecnológicos.

#### *4.3.2.1 Beneficios Sociales*

El principal beneficio social esperado es la mejora significativa en la calidad de vida de los residentes del barrio Centenario, debido a que, la optimización de la recolección de residuos reducirá la acumulación de basura en las calles, disminuyendo los malos olores y la proliferación de plagas, y contribuyendo así a un entorno más limpio y saludable para la comunidad.

Además, la implementación del sistema fomentará la participación ciudadana en la gestión de residuos, al utilizar la aplicación móvil, los residentes se involucrarán activamente en el proceso de recolección, aumentando la conciencia sobre la importancia del manejo adecuado de los residuos sólidos urbanos (RSU).

De implementarse, esta solución tecnológica posicionará al GADM Montúfar y a la ciudad de San Gabriel como un ejemplo de "smart city", debido a que, no solo mejora la eficiencia del servicio de recolección de residuos, sino que también contribuye al cumplimiento del Eje de Tecnologías emergentes para el desarrollo sostenible, específicamente en los pilares relacionados al fomento de nuevas tecnologías, y ciudades inteligentes de la Agenda de Transformación Digital del Ecuador 2022-2025 (Mintel, 2022).

El éxito de este proyecto en el barrio Centenario podría servir como un modelo replicable para otros barrios de San Gabriel y otras ciudades del Ecuador. La metodología y

tecnología desarrolladas pueden adaptarse y escalarse, ofreciendo una solución innovadora a un problema común en muchas áreas urbanas.

El principio de funcionamiento de este sistema tiene el potencial de aplicarse a diversos servicios municipales, mejorando significativamente su eficiencia, como en la distribución de gas doméstico a domicilio, la optimización de rutas y las notificaciones en tiempo real podrían reducir los tiempos de espera y mejorar la planificación de entregas.

En la recolección de residuos reciclables, se podría implementar un sistema similar para fomentar la separación en origen y mejorar el proceso de reciclaje, con rutas específicas para diferentes tipos de materiales.

Para el mantenimiento del alumbrado público y servicios de agua potable y alcantarillado, el sistema permitiría a los ciudadanos reportar fallas fácilmente y optimizaría las rutas de los equipos de reparación. Esta versatilidad no solo mejoraría la calidad de vida de los residentes, sino que también posicionaría a San Gabriel como un referente en la aplicación de tecnologías inteligentes para la gestión urbana. Esto representa un paso significativo hacia la transformación digital.

#### *4.3.2.2 Beneficios Ambientales*

El sistema proyecta tener un impacto ambiental significativo. La optimización de rutas, de acuerdo con los resultados de las pruebas realizadas, puede reducir entre un 30% y un 40% el recorrido diario de los camiones recolectores, lo que contribuye a disminuir las emisiones de $CO_2$ de estos vehículos. Esta reducción ayuda directamente a mejorar la calidad del aire en la zona, alineándose con el Objetivo 13 de los Objetivos de Desarrollo Sostenible de la Agenda 2030 para el Desarrollo Sostenible de la Organización de las Naciones Unidas (ONU), el cual se centra en la "Acción por el Clima"(Organización de las Naciones Unidas - ONU, 2015). Menos emisiones de $CO_2$ no solo reducen la contaminación del aire, sino que también ayudan a mitigar el cambio climático.

Además, al mejorar la eficiencia de la recolección, se reduce el riesgo de que los residuos terminen en lugares inadecuados, protegiendo así los ecosistemas locales, evitando que materiales peligrosos se filtren y afecten la vida silvestre y los recursos hídricos de la zona.

#### *4.3.2.3 Beneficios Económicos Indirectos*

La implementación del sistema de recolección de RSU promete beneficios económicos indirectos significativos. De acuerdo con el conductor del camión recolector de residuos, actualmente el recorrido diario en el barrio Centenario consume hasta 2 galones de diésel, con un costo de $1.80 por galón, resultando en un gasto diario de $3.60. Con la optimización de rutas, se espera reducir este consumo a 1 galón diario, generando un ahorro de $1.80 por día o aproximadamente $657 anualmente en combustible.

El mantenimiento del vehículo también se beneficiará notablemente, debido a que, según el conductor del camión recolector, el cambio de aceite al motor se realizan cada 6,000 km (aproximadamente cada mes), este proceso podría extenderse a intervalos de mes y medio a dos meses debido a la reducción el recorrido diario según los resultados de las pruebas realizadas. Esto disminuiría la frecuencia de cambios de aceite de 12 a entre 8 y 6 cambios anualmente, ahorrando entre $400 y $600 al año, asumiendo un costo de $100 por cambio. De igual forma el conductor menciona que, actualmente el cambio de neumáticos se realiza anualmente con un costo aproximado de $2000 por año. Con el nuevo sistema implementado, la vida útil de los neumáticos podría extenderse a 18 meses (año y medio), lo que significa que, en un período de 3 años, solo se necesitarán dos cambios en lugar de tres. Esto reduce el costo efectivo anual a aproximadamente $1333, resultando en un ahorro anual de $667. Esta comparación demuestra claramente cómo el nuevo sistema disminuye los costos de mantenimiento de neumáticos en un 33.35% anualmente.

En total, se proyecta un ahorro mínimo anual de aproximadamente $1,724. Estos ahorros pueden reinvertirse en iniciativas comunitarias o mejoras de servicios municipales.

En conclusión, los beneficios identificados demuestran la viabilidad y el valor a largo plazo de este proyecto para el barrio Centenario de la ciudad de San Gabriel.

Es importante destacar que, siendo la recolección de RSU un servicio básico proporcionado por el municipio, el retorno de la inversión no debe medirse únicamente en términos económicos directos. El verdadero valor del sistema radica en la satisfacción social y la contribución a la mejora del medio ambiente, representando un retorno significativo para la comunidad y el municipio en términos de calidad de vida y sostenibilidad ambiental.

En cuanto a la viabilidad económica, si bien la inversión inicial puede parecer significativa, los beneficios económicos indirectos y los ahorros proyectados justifican la implementación. Aunque inicialmente no se contempla cobrar una tasa por el uso del sistema, existe la posibilidad de que, si en el futuro se implementara un cargo mínimo, se podría generar un retorno económico más inmediato.

La viabilidad técnica del sistema queda demostrada por el funcionamiento óptimo de los componentes de hardware y software durante las pruebas realizadas. Las pruebas de consumo de datos, duración de batería, tiempos de inicialización y envío de datos en tiempo real confirman la eficiencia del sistema.

Sin embargo, se identifican áreas de mejora, particularmente en lo que respecta a la usabilidad del sistema por parte de los residentes del barrio. Se recomienda implementar una función que permita a los usuarios seleccionar manualmente el punto de recolección en un mapa interactivo, en lugar de la captura automática actual. Esto evitaría errores de navegación y mejoraría la precisión del sistema.

# CONCLUSIONES

- El sistema de recolección de RSU diseñado para el barrio Centenario de San Gabriel ha demostrado su eficacia en la optimización del proceso de recolección, logrando una interacción en tiempo real entre usuarios y recolectores.
- La implementación del sistema basado en tecnología GPS y una aplicación móvil ha probado ser una solución viable y efectiva, las pruebas de funcionamiento han validado su capacidad para disminuir tiempos y recorridos en el proceso de recolección.
- El sistema demuestra alta eficiencia operativa: la batería del nodo recolector, con 12 horas de duración promedio, cubre adecuadamente los ciclos de recolección diarios, mientras que el bajo consumo de datos móviles (9.99 MB por cada 2 horas) asegura su bajo costo de operación a largo plazo, incluso con variaciones en las rutas.
- El análisis costo/beneficio realizado ha confirmado la viabilidad técnica y económica del sistema, los resultados indican que, a pesar de requerir una inversión inicial los beneficios sociales y ambientales a largo plazo puede ser notables.
- El sistema diseñado mejora la eficiencia operativa y fomenta una mayor participación ciudadana en la gestión de residuos, representando un paso significativo hacia una comunidad más consciente y activa en temas ambientales.
- La implementación exitosa de este sistema tiene el potencial de posicionar a San Gabriel como un modelo de gestión innovadora de RSU para otras ciudades de tamaño similar en Ecuador, demostrando cómo la tecnología puede aplicarse

efectivamente para mejorar los servicios municipales y la calidad de vida de los ciudadanos.

# RECOMENDACIONES

- Se recomienda desarrollar una función que permita a los usuarios seleccionar manualmente el punto de recolección en un mapa interactivo, mejorando así la precisión de las ubicaciones de recolección y optimizando el funcionamiento del sistema de navegación en tiempo real.
- Se recomienda implementar un sistema de monitoreo continuo para evaluar el rendimiento del sistema en tiempo real. Esto permitirá realizar ajustes y mejoras según sea necesario, asegurando una operación óptima y adaptable a las necesidades de la comunidad.
- Es importante mantener actualizados los componentes de software y hardware del sistema. Se recomienda establecer un programa de mantenimiento y actualización regular para garantizar la seguridad, eficiencia y disponibilidad del sistema.
- Se sugiere implementar el sistema inicialmente como un proyecto piloto en el barrio Centenario, con miras a una expansión progresiva a otras áreas de San Gabriel. Esto permitirá mejorar el sistema basándose en los resultados de un plan piloto antes de una implementación más amplia.
- Es vital considerar la posibilidad de actualizar a un plan empresarial de Mapbox en el futuro, especialmente si se planea expandir el sistema a más áreas, para superar la limitación actual de 25 puntos de recolección por ruta en la versión gratuita.
- Se recomienda considerar la implementación futura de contenedores públicos inteligentes con sensores de llenado integrados. Esta adición podría mejorar aún más la eficiencia de la recolección, permitiendo una planificación más precisa de las rutas y frecuencias de recolección.

## REFERENCIAS


3GPP. (2021). *Especificaciones técnicas y los requisitos para el funcionamiento de la capa física de LT: ESpecification #36.101*. https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2411

Ambientum. (2022, November 28). *Composición de los residuos sólidos urbanos - Ambientum*. https://www.ambientum.com/enciclopedia_medioambiental/suelos/composicion_de_los_rsu.asp

Android Developers. (2023). *Desarrollo para-Android | Android Developers*. https://developer.android.com/develop?hl=es-419

Arduino. (2018). *What is Arduino? | Arduino*. https://www.arduino.cc/en/Guide/Introduction

Arduino. (2022). *Getting Started with Arduino IDE 2 | Arduino Documentation*. https://docs.arduino.cc/software/ide-v2/tutorials/getting-started-ide-v2/

Asamblea Nacional Constituyente de Ecuador. (2008). *Constitución de la República del Ecuador*.

Asamblea Nacional del Ecuador. (2010). *Código Orgánico de Organización Territorial, Autonomía y Descentralización (COOTAD)*.

Asamblea Nacional del Ecuador. (2017a). *Código Orgánico del Ambiente (COA)*. www.lexis.com.ec

Asamblea Nacional del Ecuador. (2017b). *TEXTO UNIFICADO DE LEGISLACION SECUNDARIA DE MEDIO AMBIENTE (TULSMA)*. www.lexis.com.ec

Asamblea Nacional del Ecuador. (2019). *REGLAMENTO AL CODIGO ORGANICO DEL AMBIENTE (RCODA)*. www.lexis.com.ec

Barradas, A. (2009). *Gestión Integral de Residuos Sólidos Municipales - Estado del Arte*.

Behdad, S., Esmaeilian, B., Wang, B., Lewis, K., Duarte, F., & Ratti, C. (2018). *The Future of Waste Management in Smart and Sustainable 2 Cities: A Review and Concept Paper*.

Bizzotto, A. (2018). *Intro to Platform Channels: Building an Image Picker in Flutter*. https://codewithandrea.com/articles/platform-channels-flutter/

Ciset ES. (2021). *Software - Concepto y tipos*. https://www.ciset.es/glosario/480-software-concepto-y-tipos

CNT AppGeoportal. (2024). *GEOPORTAL CNT EP*. https://gis.cnt.gob.ec/appgeoportal/index.php?u=-77.82328,0.58725,16

Departamento de Defensa de Estados Unidos. (2007). *GLOBAL POSITIONING SYSTEM PRECISE POSITIONING SERVICE PERFORMANCE STANDARD*.

Firebase. (2023). *Firebase Realtime Database | Almacena y sincroniza datos en tiempo real*. https://firebase.google.com/products/realtime-

database?utm_source=google&utm_medium=cpc&utm_campaign=latam-LATAM-all-es-dr-SKWS-all-all-trial-b-dr-1707800-LUAC0020239&utm_content=text-ad-none-any-DEV_c-CRE_654758081663-ADGP_Hybrid%20%7C%20SKWS%20-%20BRO%20%7C%20Txt_Compute-Firebase-KWID_43700076085058368-kwd-308670941208&utm_term=KW_firebase-ST_Firebase&gad_source=1&gclid=CjwKCAjwuJ2xBhA3EiwAMVjkVOFsjNUAaGxptiHFvtfMPZKwoke6GXZyVLG-q3L3IEY8Qq3bkgUIyxoCK7EQAvD_BwE&gclsrc=aw.ds&hl=es-419

Flutter. (2023). *Docs | Flutter*. https://docs.flutter.dev/

FTP Cloud. (2021). *¿Qué es Firebase? Pros y contras y servicios de Firebase*. https://fptcloud.com/firebase-la-gi/

GADM Montúfar. (2020). *Plan de Desarrollo y Ordenamiento Territorial - PDYOT (Actualización)*.

GADM Montufar. (2021a). *ORDENANZA PARA LA GESTIÓN INTEGRAL DE RESIDUOS SÓLIDOS Y DESECHOS LIQUIDOS (ACEITES DE COCINA USADO) EN EL CANTÓN MONTÚFAR.* http://gadmontufar.gob.ec

GADM Montufar. (2021b). *ORDENANZA: REGULAR LA TARIFA POR PRESTACIÓN DEL SERVICIO GESTIÓN INTEGRAL DE RESIDUOS SOLIDOS NO PELIGROSO EN EL CANTON MONTÚFAR*.

Galindo, M., Revista, S., Derecho, J., & Galindo Soza, M. (2018). La pirámide de Kelsen o jerarquía normativa en la nueva CPE y el nuevo derecho autonómico. *Revista Jurídica Derecho*, *7*(9), 126–148. http://www.scielo.org.bo/scielo.php?script=sci_arttext&pid=S2413-28102018000200008&lng=es&nrm=iso&tlng=es

Geodata. (2024). *Mapas de Cobertura Red Móvil Claro Ec*. https://www.geodata.com.ec/

GisGeography. (2020). *5 Best Web Mapping Platforms - The Battle of Web GIS - GIS Geography*. https://gisgeography.com/web-mapping/

Gob. Argentina. (2019). *Gestión de residuos sólidos urbanos | Argentina.gob.ar*. https://www.argentina.gob.ar/ambiente/control/rsu

Google Cloud Platform Blog. (2015). *Google Cloud Platform Blog: Add backend logic to real-time data with Firebase and Google App Engine*. https://cloudplatform.googleblog.com/2015/10/add-backend-logic-to-real-time-data-with-Firebase-and-Google-App-Engine.html

IBM. (n.d.). *¿Qué es la tecnología móvil?* Retrieved December 2, 2023, from https://www.ibm.com/mx-es/topics/mobile-technology

Instituto Europeo de Postgrado - IEP. (2022, July 7). *Metodología Waterfall: la gestión de proyectos en cascada - IEP*. https://iep.edu.es/metodologia-waterfall/

Instituto Nacional de Estadística y Censo (INEC), Asociación de Municipalidades Ecuatorianas (AME), & Banco de Desarrollo del Ecuador (BDE). (2021). *Boletín*

*técnico No 04-2020-GAD Municipales: Estadística de Información Ambiental Económica en Gobiernos Autónomos Descentralizados Municipales - Gestión de Residuos Sólidos*.

Iqbal, M. S., Rahim, Z. B. A., Hussain, S. A., Ahmad, N., Kaidi, H. M., Ahmad, R., & Dziyauddin, R. A. (2021). Mobile communication (2G, 3G and 4G) and future interest of 5G in Pakistan: A review. *Indonesian Journal of Electrical Engineering and Computer Science*, *22*(2), 1061–1068. https://doi.org/10.11591/IJEECS.V22.I2.PP1061-1068

Kaza, S., Yao, L. C., Bhada-Tata, P., & Van Woerden, F. (2018). What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050. *What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050*. https://doi.org/10.1596/978-1-4648-1329-0

Leal, R. R. (2013). *RECEPTOR GPS BASADO EN SOFTWARE*. Instituto Politécnico Nacional.

Mancera, L. (2004). *Introduccion a la Metrología de Tiempo y Frecuencia*.

Mapbox. (2023). *Mapbox Docs*. https://docs.mapbox.com/

Ministerio del Ambiente. (2022). *Proyecto de Gestión de residuos sólidos y economía circular inclusiva (GRECI): Diagnóstico sectorial de la Gestión integral de residuos y desechos sólidos no peligrosos en municipios del Ecuador.*

Mintel. (2022). *AGENDA DE TRANSFORMACÓN DIGITAL DEL ECUADOR 2022-2025*.

Mompó, S. G. (2019). *Medición y Análisis de las Redes de Comunicaciones Móviles 4G LTE en Cullera*.

Murray R. Spiegel y Larry J. Stephens. (2005). Estadística. 4ta edición. Mc Graw-Hill. México, D.F.

naylampmechatronics. (2016). *Tutorial Módulo GPS con Arduino*. https://naylampmechatronics.com/blog/18_tutorial-modulo-gps-con-arduino.html

Oficina de Coordinación Nacional de Posicionamiento Navegación y Cronometría por Satélite. (n.d.). *El Sistema de Posicionamiento Global (GPS)*. Retrieved December 2, 2023, from https://www.gps.gov/systems/gps/spanish.php

Organización de las Naciones Unidas - ONU. (2015, September 25). *Transforming our world: the 2030 Agenda for Sustainable Development | Department of Economic and Social Affairs*. https://sdgs.un.org/2030agenda

Pozo-Ruz, A., Ribeiro, A., García-Alegre, M. C., García, L., Guinea, D., & Sandoval, F. (n.d.). *SISTEMA DE POSICIONAMIENTO GLOBAL (GPS): DESCRIPCIÓN, ANÁLISIS DE ERRORES, APLICACIONES Y FUTURO*.

Revelo Morales, J. A. (2019). *Propuesta de un plan de manejo integral de residuos sólidos para la población del cantón Piñas, provincia de El Oro*. http://dspace.ups.edu.ec/handle/123456789/17504

Santerre, R., Pan, L., Cai, C., & Zhu, J. (2014). Single Point Positioning Using GPS, GLONASS and BeiDou Satellites. *Positioning*, *05*(04), 107–114. https://doi.org/10.4236/POS.2014.54013

Sasaki, M. (1993). Hardware. *Rinsho Byori. The Japanese Journal of Clinical Pathology*, *41*(4), 443–450. https://doi.org/10.29057/IXTLAHUACO.V6I11.11967

SpeedChecker. (2022). *EXPERIENCIA DE USUARIO EN REDES MÓVILES*. https://www.speedcheckercdn.com/pdfs/rendimiento-de-la-red-movil-ecuador-octubre2022.pdf

Tapias, D. (2014). *Proyectos de Desarrollo Software*.

UNEP. (2018). *Perspectiva de la Gestión de Residuos en América Latina y el Caribe*.

Xinyuan. (2023). *LilyGo SIM7600X*. Https://Github.Com/Xinyuan-LilyGO/T-SIM7600X.

# ANEXOS

## 8.1 ANEXO 1: Formato de encuesta aplicada

**UNIVERSIDAD TÉCNICA DEL NORTE**

**FACULTAD EN CIENCIAS APLICADAS**

**CITEL**

Ingeniería en Telecomunicaciónes

**ENCUESTA DIRIGIDA LOS HABITANTES DEL BARRIO CENTENARIO.**

Levantamiento de información sobre la situación actual y requerimientos de los habitantes del barrio Centenario, ciudad de San Gabriel con relación a la recolección de residuos sólidos urbanos en el sector.

1. **¿Cuántas personas viven en el domicilio?**

a) 0-3
b) 3-5
c) Más de 5

2. **Aproximadamente, ¿Qué cantidad de residuos sólidos urbanos genera en el domicilio a la semana? (*Considérese como medida un costal*)**

a) Menos de ½ costal
b) 1 a 2 costales
c) Más de 2 costales

3. **¿Con qué frecuencia saca los residuos sólidos para recolección?**

a) 1 vez por semana
b) 2 veces por semana
c) 3 veces o más

4. **Respecto a la infraestructura y recursos disponibles** (camiones recolectores, alarma sonora, personal) **del GADM Montufar para la recolección de los residuos sólidos urbanos, los considera**:

a) Muy buena
b) Buena
c) Mala

5. **¿Han existido demoras en los horarios de recolección actual de los residuos sólidos urbanos?**

a) Frecuentemente

b) Rara vez
c) Nunca

**6. ¿Está de acuerdo con los horarios actuales de recolección de los residuos sólidos urbanos?**

a) Si
b) No

**7. ¿Cómo percibe el impacto de las bolsas de basura expuestas en los exteriores de los hogares y la acción de la fauna urbana en romper las fundas de residuos, generando contaminación visual/ambiental?**

a) Muy perjudicial
b) Algo perjudicial
c) Neutral
d) No estoy seguro/a.
e) No lo considero perjudicial

**8. ¿Cómo considera al sistema actual de recolección de residuos sólidos urbanos?**

d) Muy bueno
e) Bueno
f) Malo

**9. ¿Cree que sería útil la implementación de un sistema de recolección de residuos sólidos urbanos, que cuente con una aplicación móvil para solicitar y ser notificado sobre la recolección de los residuos?**

a) Sí
b) No

**10. ¿Dispone de conexión a internet en su hogar?**

a) Sí
b) No

**11. ¿Qué tipo de conexión a internet utilizan en su hogar?** *(Puede elegir más de una opción)*

a) Wifi
b) Datos móviles

**12. Marque con una "X" de acuerdo a su criterio de utilidad respecto a las siguientes funciones de la aplicación del sistema de recolección de residuos sólidos urbanos:**

| **Funciones** | **Muy Útil** | **Útil** | **Poco Útil** |
|---|---|---|---|
| Visualizar en tiempo real la ubicación del camión recolector mediante la aplicación. | | | |

| | | | |
|---|---|---|---|
| Visualizar el tiempo aproximado de llegada del carro recolector al domicilio. | | | |
| Emitir una notificación cuando el camión recolector este próximo a llegar al domicilio. | | | |
| Emitir una notificación de llegada del camión recolector. | | | |

13. **Tiene alguna sugerencia que le gustaría que se considere en el sistema de recolección de residuos sólidos.**

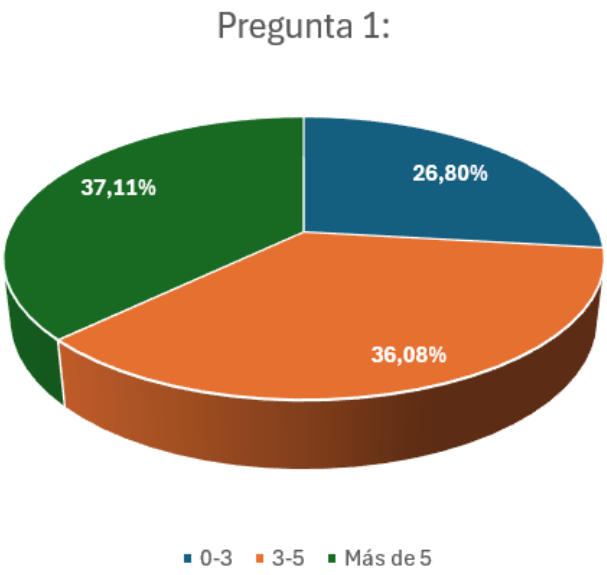
## 8.2 ANEXO 2: Tabulación de resultados de la encuesta

**PREGUNTA 1 -** ¿Cuántas personas viven en su domicilio?

Como se muestra en la Figura 116 la mayoría de los hogares encuestados tienen más de 5 personas (37.11%), seguido por 3-5 personas (36.08%) y 0-3 personas (26.80%). La mayoría de los hogares son de tamaño mediano a grande, con más de 3 personas, lo que implica una mayor generación de residuos y la necesidad de un servicio de recolección eficiente.

**Figura 115**

*Representación en porcentaje de los resultados de la Pregunta 1*

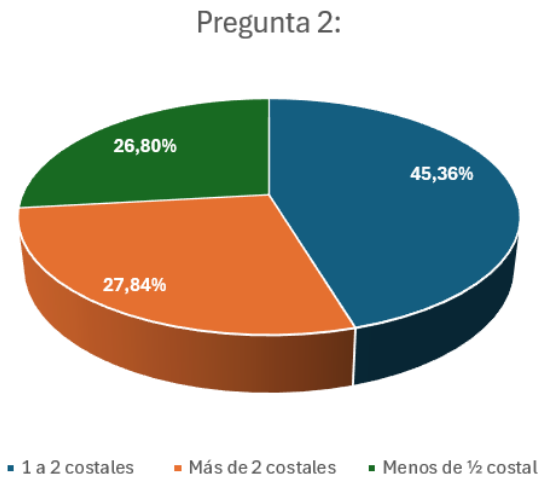



**PREGUNTA 2 -** Aproximadamente, ¿Qué cantidad de residuos sólidos urbanos genera su hogar a la semana? (*Considérese como medida un costal*)

En la Figura 117 se demuestra que el 45.36% genera entre 1 a 2 costales de residuos, el 27.84% genera más de 2 costales, y el 26.80% genera menos de medio costal. Casi la mitad de los hogares generan entre 1 a 2 costales de residuos semanalmente, lo que representa una cantidad considerable de desechos que deben ser manejados de manera adecuada. Esto demuestra la necesidad de un sistema óptimo para manejar esta gran cantidad de desechos generados.

**Figura 116**

*Representación en porcentaje de los resultados de la Pregunta 2*

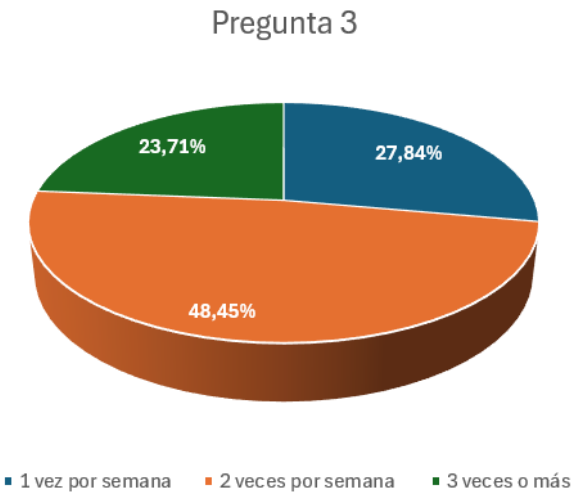



**PREGUNTA 3** - ¿Con qué frecuencia saca sus residuos sólidos para recolección?

En la Figura 118 se visualiza los resultados obtenidos en los cuales se demuestra que la mayoría (48.45%) saca la basura 2 veces por semana, el 27.84% 1 vez por semana, y el 23.71% 3 veces o más por semana. La mayoría saca la basura 2 veces por semana, lo que sugiere que la frecuencia actual es aceptable para la mayoría de los hogares. Sin embargo, un 23.71% la saca 3 veces o más por semana, lo que podría indicar la necesidad de un sistema optimizado para ajustar la frecuencia según las necesidades específicas de cada hogar.

**Figura 117**

*Representación en porcentaje de los resultados de la Pregunta 3*

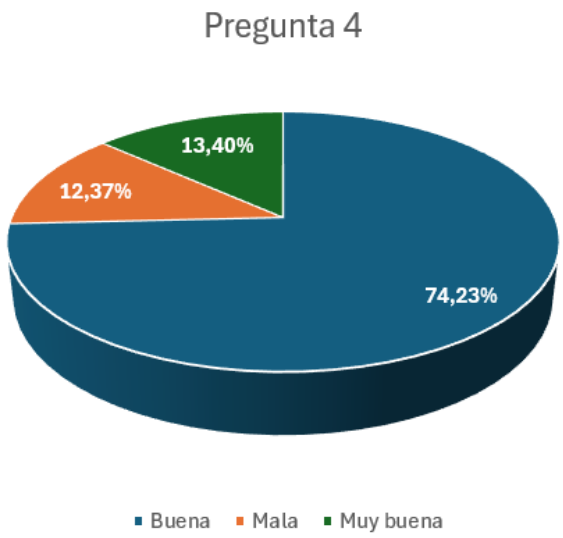


**PREGUNTA 4** - Respecto a la infraestructura y recursos disponibles (*camiones recolectores, alarma sonora, personal*) del GADM Montufar para la recolección de los residuos sólidos urbanos, usted los considera:

Como se muestra en la Figura 119 los resultados obtenidos identifican que la mayoría (74.23%) califica la infraestructura como buena, el 13.40% como muy buena, el 12.37% como mala. La mayoría considera que la infraestructura actual es buena o muy buena, lo que sugiere que no es un factor crítico que requiera una mejora sustancial en este momento. Aunque la percepción general es positiva, no debe descartarse la mejora continua de la infraestructura.

**Figura 118**

*Representación en porcentaje de los resultados de la Pregunta 4*

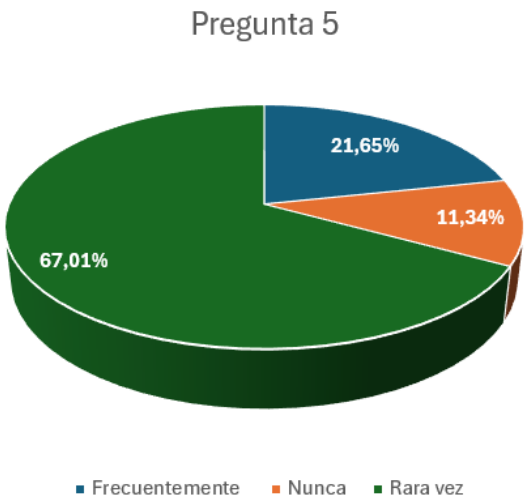



**PREGUNTA 5** - ¿Han existido demoras en los horarios de recolección actual de sus residuos sólidos?

De acuerdo con los resultados que se visualizan en la Figura120 tenemos que el 67.01% indica que rara vez han existido demoras, el 21.65% que frecuentemente, y el 11.34% que nunca. Si bien la mayoría experimenta pocas demoras, la optimización de rutas podría reducir aún más estos inconvenientes.

**Figura 119**

*Representación en porcentaje de los resultados de la Pregunta 5*

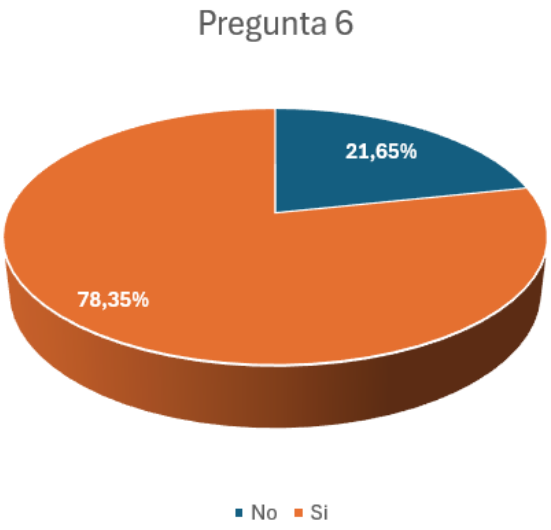



**PREGUNTA 6 -** ¿Está de acuerdo con los horarios actuales de recolección de los residuos sólidos urbanos?

En la Figura 121 se visualizan los resultados que indican un 78.35% para los habitantes que están de acuerdo con los horarios actuales de recolección, mientras que el 22% no lo está. Aunque la mayoría está satisfecha con los horarios actuales, es importante considerar las preferencias minoritarias para mejorar la satisfacción general.

**Figura 120**

*Representación en porcentaje de los resultados de la Pregunta 6*

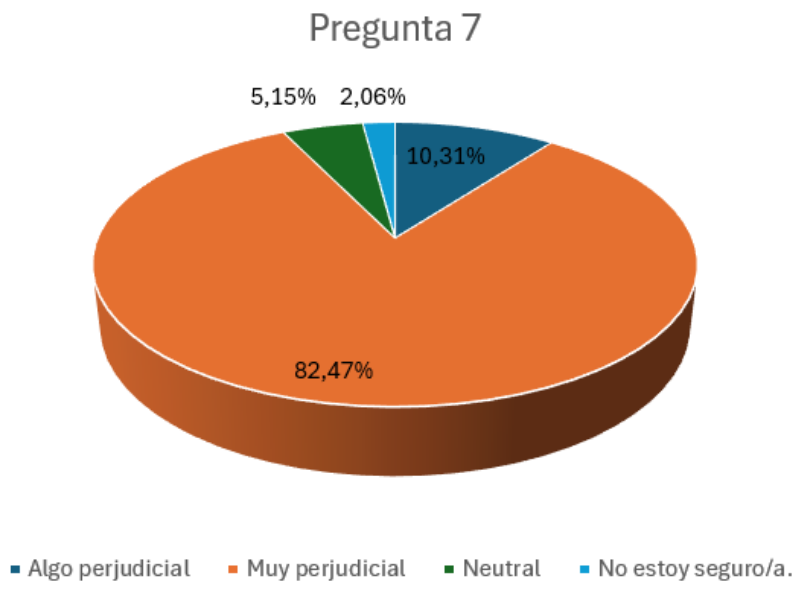


**PREGUNTA 7 -** ¿Cómo percibe el impacto de las bolsas de basura expuestas en los exteriores de los hogares y la acción de la fauna urbana en romper las fundas de residuos, generando contaminación visual/ambiental?

El resultado obtenido de acuerdo con la Figura 122 demuestra que el 82.47% lo considera muy perjudicial, el 10.31% algo perjudicial, el 5.15% neutral y el 2.06% no está seguro, nadie voto por la opción de no lo considera perjudicial. La gran mayoría considera que la acumulación de residuos expuesta en los exteriores de los hogares es muy perjudicial o algo perjudicial, lo que destaca la importancia de un servicio de recolección eficiente para mitigar este problema.

**Figura 121**

*Representación en porcentaje de los resultados de la Pregunta 7*

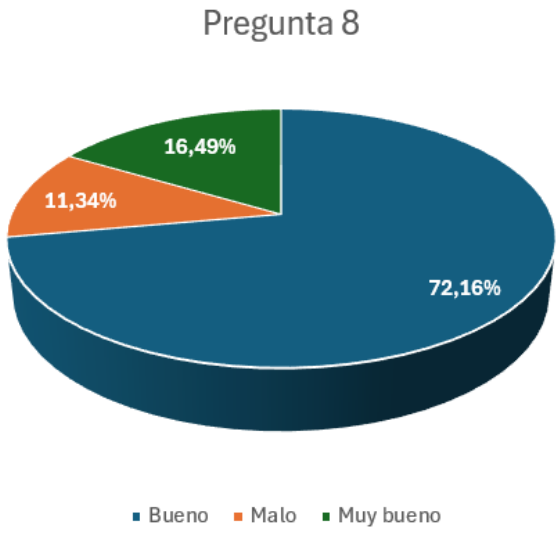



**PREGUNTA 8 -** ¿Cómo considera que es el sistema actual de recolección de residuos sólidos?

Los resultados obtenidos se visualizar en la Figura 123, en donde se tiene que el 72.16% lo considera bueno, el 16.49% muy bueno, el 11.34% malo. Aunque la mayoría está satisfecha, la mejora continua es clave para mantener altos estándares de servicio.

**Figura 122**

*Representación en porcentaje de los resultados de la Pregunta 8*

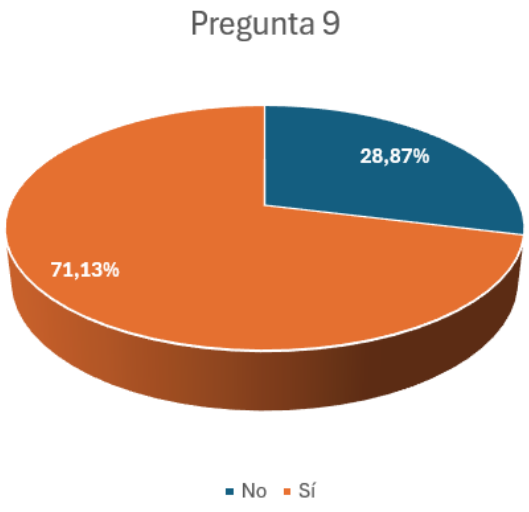



**PREGUNTA 9 -** ¿ Cree que sería útil la implementación de un sistema de recolección de residuos sólidos urbanos, que cuente con una aplicación móvil para solicitar y ser notificado sobre la recolección de los residuos?

De acuerdo con la Figura 124, el 71.13% cree que sería muy útil, el 28.87% no útil. La mayoría cree que un sistema de notificación sería muy útil o útil, lo que demuestra la aceptación por parte de los residentes hacia este tipo de solución tecnológica.

**Figura 123**

*Representación en porcentaje de los resultados de la Pregunta 9*



**PREGUNTA 10 -** ¿Dispone de conexión a internet en su hogar?

Los resultados que se visualizan en la Figura 125 indican que el 95.89% sí dispone de conexión a internet, mientras que el 4.11% no. La gran mayoría dispone de conexión a internet, lo que sugiere que un sistema de notificación en línea sería accesible para la mayoría de los residentes.

**Figura 124**

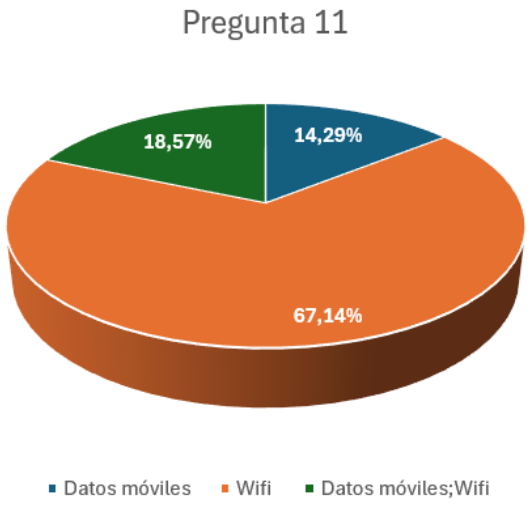
*Representación en porcentaje de los resultados de la Pregunta 10*


**PREGUNTA 11 -** ¿ Qué tipo de conexión a internet utilizan en su hogar? *(Puede elegir más de una opción)*

Los resultados que se muestran en la Figura 126 demuestran que el 67.14% utiliza WiFi, el 18.57% datos móviles y el 14.29% utilizan tienen la posibilidad de usar cualquiera de las dos tecnologías. La gran mayoría dispone de conexión a internet, a través de WiFi o datos móviles, lo que indica que los residentes cuentan con la capacidad de hacer uso de un sistema móvil en línea para la recolección de sus RSU.

**Figura 125**

*Representación en porcentaje de los resultados de la Pregunta 11*



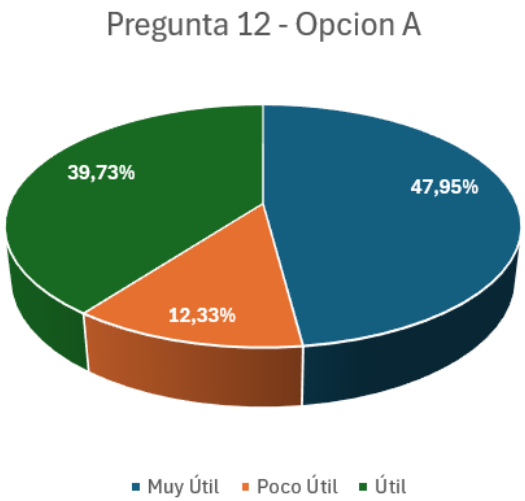

**PREGUNTA 12** – Defina su criterio de utilidad respecto a las siguientes funciones de la aplicación del sistema de recolección de residuos sólidos urbanos

- **Opción A** - Visualizar en tiempo real la ubicación del camión recolector mediante la aplicación.

De acuerdo con la Figura 127 tenemos que el 47.95% lo considera muy útil, el 39.73% útil, y el 12.33% poco útil.

**Figura 126**

*Representación en porcentaje de los resultados de la Pregunta 12 - Opción A*

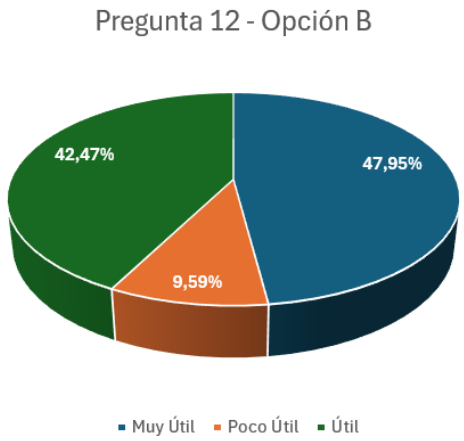


- **Opción B** - Visualizar el tiempo aproximado de llegada del carro recolector al domicilio.

De acuerdo con la Figura 128 tenemos que 47.95% lo considera muy útil, el 42.47% útil, y el 9.59% poco útil.

**Figura 127**

*Representación en porcentaje de los resultados de la Pregunta 12 - Opción B*

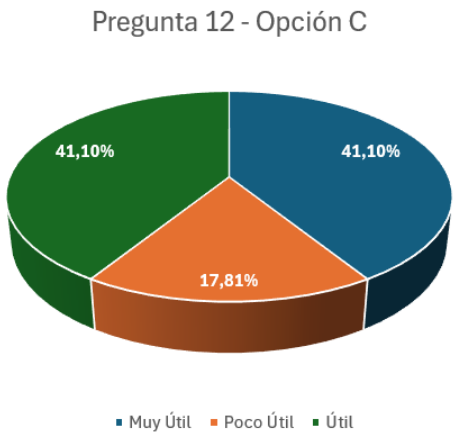



- **Opción C** - Emitir una notificación cuando el camión recolector este próximo a llegar al domicilio.

De acuerdo con la Figura 129 tenemos que el 41.1% lo considera muy útil, el 41.1% útil, y el 17.8% poco útil.

**Figura 128**

*Representación en porcentaje de los resultados de la Pregunta 12 - Opción C*

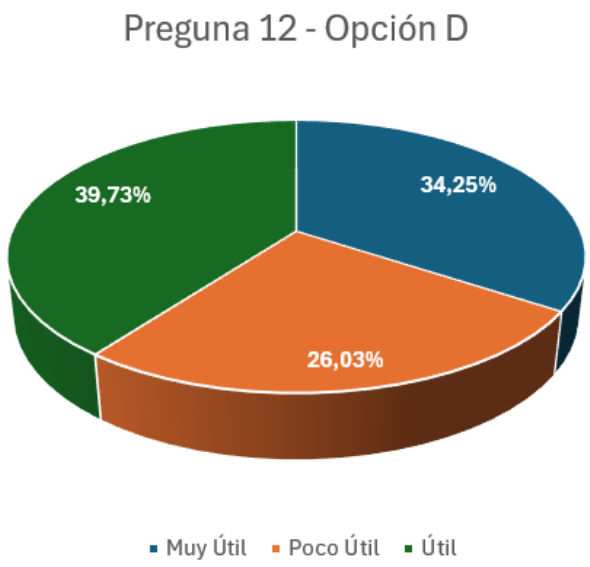


- **Opción D** - Emitir una notificación de llegada del camión relector.

De acuerdo con la Figura 130 tenemos que el 39.73% lo considera muy útil, el 34.25% útil, y el 26.03% poco útil.

**Figura 129**

*Representación en porcentaje de los resultados de la Pregunta 12 - Opción D*




Las funcionalidades más valoradas son visualizar la ubicación en tiempo real y el tiempo aproximado de llegada del camión recolector en conjunto con la emisión de la notificación cuando el este próximo a llegar, lo que indica la importancia de la información en tiempo real para los residentes del barrio Centenario.

**PREGUNTA 13** – Tiene alguna sugerencia que le gustaría que se considere en el sistema de recolección de residuos sólidos.

En este apartado se verifico que más del 50% no tenía sugerencias, mientras que menos del 50% sugirió cosas como implementación de más camiones recolectores, mejores horarios, separación de residuos, contenedores públicos, control de animales, entre otros, lo que sugiere mejoras adicionales.

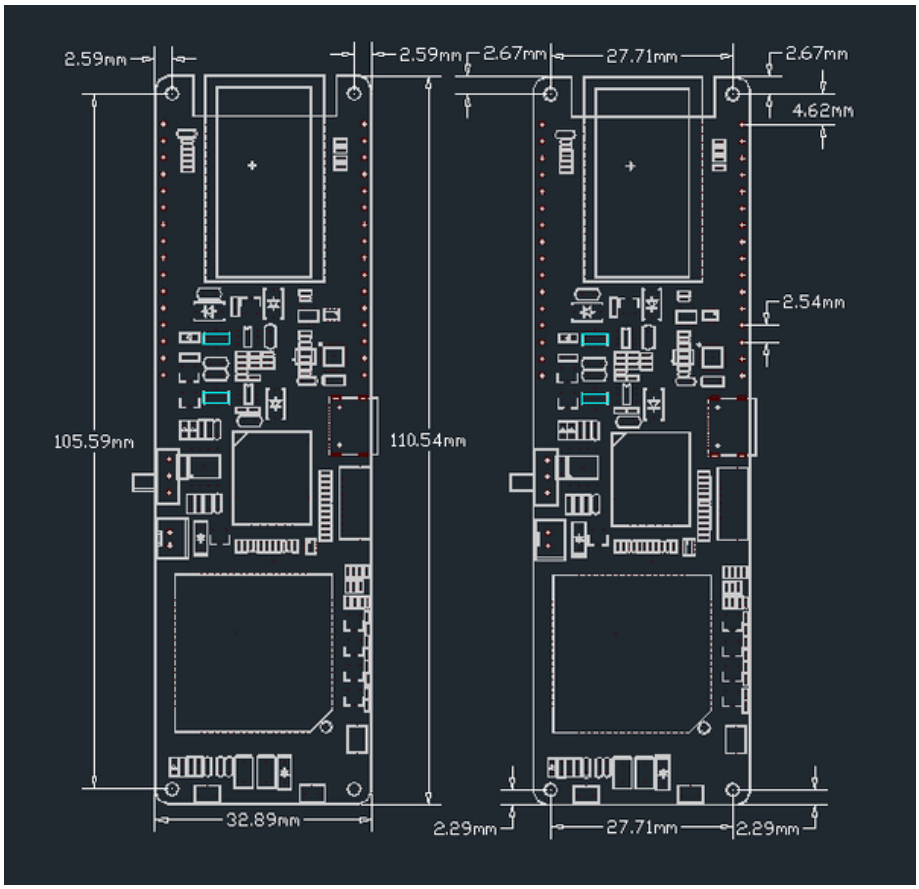
## 8.3 ANEXO 3: Datasheet ESP32 T-SIM7600NA

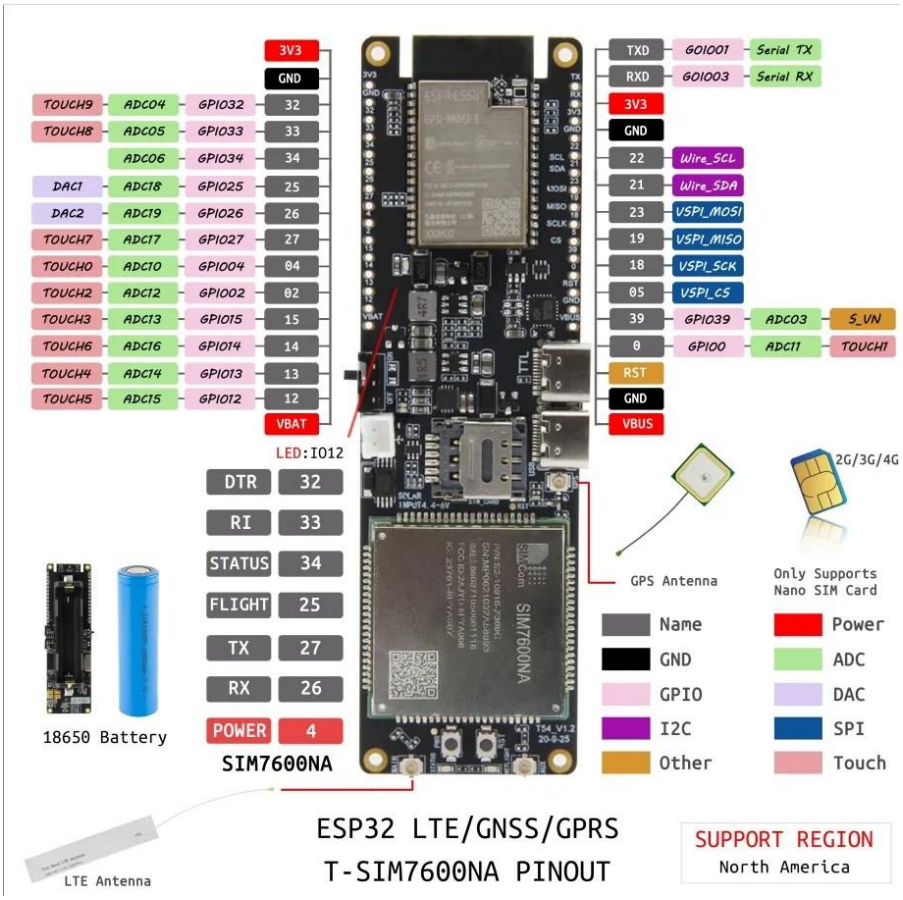
# Specifications

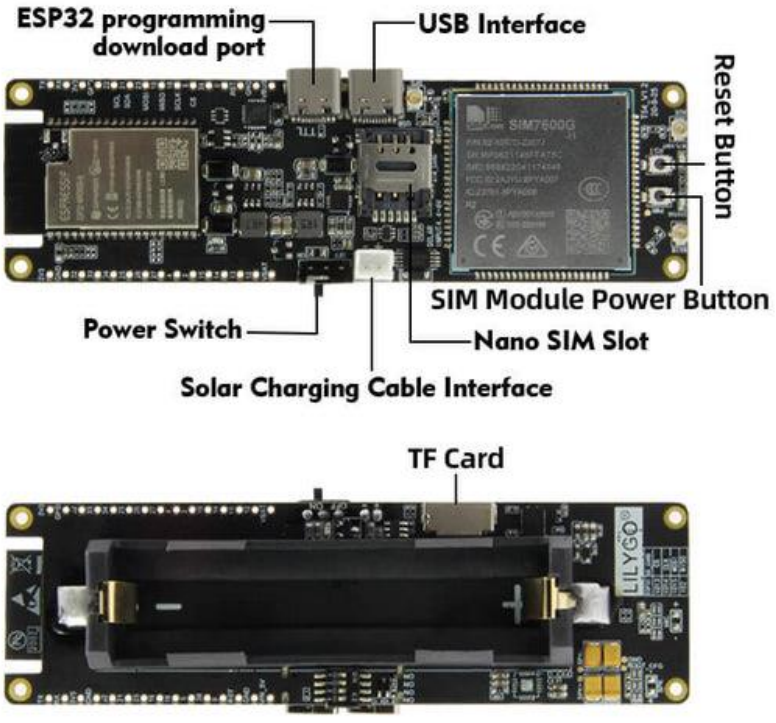
| Product | T-SIM7600G-H R2 | T-SIM7600NA |
|---|---|---|
| MCU | ESP32 | ESP32 |
| LTE&GSM | SIM7600G-H R2 | SIM7600NA |
| SIM Card | ✓ | ✓ |
| TF Card | ✓ | ✓ |
| **ESP32** | | |
| Clock Speed | 240Mhz | 240Mhz |
| Flash | 4MB | 4MB |
| PSRAM | 8MB | 8MB |
| Wireless Connectivity | Wi-Fi: 802.11 b/g/n<br>Bluetooth: v4.2 BR/EDR and BLE | |
| Peripheral Interfaces | I2C、SPI、UART、SDIO、I2S、CAN | |
| **Comparison** | **SIM7600G-H R2** | **SIM7600NA** |
| Form Factor | LCC+LGA, 119PIN | LCC, 87PIN |
| REGION | G:Global | NA:North America<br>America,Canada |
| Dimensions | 30.0*30.0*2.9mm | 30.0*30.0*2.9mm |

| | | | |
|---|---|---|---|
| Frequency Bands | LTE-FDD | B1/B2/B3/B4/B5/B7/ B8/B12/ B13/B18/B19/ B20/B25/B26/B28/B66 | B2/B4/B5/B12/B13 B14/B25/B26/B66/B71 |
| | LTE-TDD | B34/B38/B39/B40/B41 | B41 |
| | WCDMA | B1/B2/B5/B8 | |
| | GSM | 850/900/1800/1900MHz | |
| Power Consumption | LTE(mA) | 2.3 | 2.3 |
| | WCDMA(mA) | 3.3 | 3.3 |
| | GSM(mA) | 2.8 | 2.8 |
| Data Transfer | LTE(Mbps) | 150(DL)/50(UL) | 10(DL)/5(UL) |
| | HSPA(Mbps) | 42(DL)/5.76(UL) | 42(DL)/5.76(UL) |
| | WCDMA (Kbps) | 384(DL)/384(UL) | 384(DL)/384(UL) |
| | GPRS/EDGE (Kbps) | 236.8(DL)/236.8(UL) | 236.8(DL)/236.8(UL) |
| GNSS | | BeiDou、GPS、GLONASS | GPS、GLONASS |
| Protocol | | TCP/IP/IPV4/IPV6/DNS/<br>Multi-PDP/FTP/FTPS/HTTP/HTTPS | |
| Android RIL | | Android 5.0/6.0/7.0/8.0/9.0 | |
| USB Driver | | Microsoft Windows 7/8/10, Linux, Android | |
| Supply Voltage | | 3.4~4.2V | 3.4~4.2V |
| SIM Card | | 1.8V/3.0V | 1.8V/3.0V |
| Certification | | CE-RED/FCC/IC/RCM/CCC/ Telec/Jate/Anatel/ NCC/IMDA*/ ICASA*/PTCRB/GCF/RoHS/REACH | CE-RED/ACMA/FCC/ NCC/RoHS/REACH/Anatel |

| Frequency | | T-SIM7600G-H R2 | T-SIM7600NA |
|---|---|---|---|
| GSM | 850Mhz | ✓ | |
| | 900Mhz | ✓ | |
| | 1800Mhz | ✓ | |
| | 1900Mhz | ✓ | |
| LTE-FDD | B1 | ✓ | |
| | B2 B4 | ✓ | ✓ |
| | B5 B12 | ✓ | ✓ |
| | B13 B14 | | ✓ |
| | B18 B19 | ✓ | |
| | B25 B26 | ✓ | ✓ |
| | B66 | ✓ | ✓ |
| | B71 | | ✓ |
| LTE-TDD | | B34/B38/B39/B40/B41 | B41 |
| GNSS | BeiDou | ✓ | |
| | GPS Glonass | ✓ | ✓ |
| WCDMA | | B1/B2/B4/B5/B6/B8/B19 | |




3V3
GND
TOUCH9 ADC04 GPIO32 32
TOUCH8 ADC05 GPIO33 33
ADC06 GPIO34 34
DAC1 ADC18 GPIO25 25
DAC2 ADC19 GPIO26 26
TOUCH7 ADC17 GPIO27 27
TOUCH0 ADC10 GPIO04 04
TOUCH2 ADC12 GPIO02 02
TOUCH3 ADC13 GPIO15 15
TOUCH6 ADC16 GPIO14 14
TOUCH4 ADC14 GPIO13 13
TOUCH5 ADC15 GPIO12 12
VBAT
LED:IO12
DTR 32
RI 33
STATUS 34
FLIGHT 25
TX 27
RX 26
POWER 4
SIM7600NA
TXD GOIO01 Serial TX
RXD GOIO03 Serial RX
3V3
GND
22 Wire_SCL
21 Wire_SDA
23 VSPI_MOSI
19 VSPI_MISO
18 VSPI_SCK
05 VSPI_CS
39 GPIO39 ADC03 S_VN
0 GPIO00 ADC11 TOUCH1
RST
GND
VBUS
GPS Antenna
2G/3G/4G
Only Supports Nano SIM Card
18650 Battery
LTE Antenna
Name
GND
GPIO
I2C
Other
Power
ADC
DAC
SPI
Touch
ESP32 LTE/GNSS/GPRS
T-SIM7600NA PINOUT
SUPPORT REGION
North America


ESP32 programming download port
USB Interface
Reset Button
SIM Module Power Button
Power Switch
Nano SIM Slot
Solar Charging Cable Interface
TF Card

## 8.4 ANEXO 4: Enlace al repositorio del código de la placa Arduino del Nodo Recolector

**CodigoFuente_NodoRecolector_ESP32-TSIM 7600NA**

## 8.5 ANEXO 5: Enlace al repositorio del código de la App Móvil

**CodigoFuente_AppMovil_ResiRutas**

## 8.6 ANEXO 6: Plan de implementación

# PLAN DE IMPLEMENTACIÓN

**“SISTEMA DE RECOLECCIÓN DE RESIDUOS SÓLIDOS URBANOS (RSU) BASADO EN OPTIMIZACIÓN DE RUTAS Y NOTIFICACIÓN EN TIEMPO REAL PARA EL BARRIO CENTENARIO, CIUDAD DE SAN GABRIEL”**

**AUTOR: CRISTIAN IVÁN QUELAL GUADIR**

**IBARRA-ECUADOR**
**2024**

# ÍNDICE DE CONTENIDOS

# 1. RESUMEN

El presente Plan de Implementación detalla la estrategia para desplegar el Sistema de Recolección de Residuos Sólidos Urbanos (RSU) en el Barrio Centenario de la ciudad de San Gabriel. Este sistema innovador integra tecnologías de geolocalización, optimización de rutas y comunicación en tiempo real para mejorar significativamente la eficiencia del servicio de recolección de residuos.

El plan abarca las fases importantes del proceso de implementación, desde la preparación inicial y el despliegue de infraestructura hasta la capacitación del personal y la evaluación continua del sistema. Se ha diseñado un cronograma detallado que distribuye las actividades en un periodo de 18 semanas, asegurando una transición fluida y minimizando las interrupciones en el servicio actual.

La inversión inicial para este proyecto se estima en $3083, que incluye costos de hardware, software, capacitación y el primer año de mantenimiento. Esta inversión se proyecta para generar beneficios significativos a largo plazo, tanto en términos de eficiencia operativa como en la satisfacción de los residentes del Barrio Centenario.

El éxito de este sistema dependerá en gran medida de la adopción por parte de los usuarios y del personal municipal. Por lo tanto, se ha puesto especial énfasis en las estrategias de comunicación y capacitación, asegurando que todos los involucrados estén bien informados y preparados para utilizar el nuevo sistema de manera efectiva.

# 2. OBJETIVOS

## 2.1. Objetivo General

Implementar el sistema de optimización de recolección de Residuos Sólidos Urbanos (RSU) en el barrio Centenario de San Gabriel, asegurando su correcta integración, funcionamiento y adopción por parte de usuarios y personal municipal.

## 2.2. Objetivos Específicos

- Desplegar la infraestructura tecnológica y capacitar al personal municipal, operativo y comunidad en el uso del sistema de recolección de RSU, garantizando su correcto uso, operación y mantenimiento.
- Ejecutar pruebas piloto del sistema en un sector del Barrio Centenario para evaluar su funcionamiento, realizar ajustes necesarios y asegurar su óptimo desempeño antes del lanzamiento.
- Implementar el sistema en todo el Barrio Centenario, acompañado de una estrategia de comunicación efectiva, para lograr una adopción exitosa por parte de los residentes y maximizar su participación.
- Establecer un proceso de seguimiento, soporte y evaluación continua del sistema para garantizar su funcionamiento eficiente, resolver incidencias oportunamente y proponer mejoras basadas en diferentes análisis y satisfacción de usuarios.

# 3. COMPONENTES DEL SISTEMA DE RECOLECCIÓN DE RSU

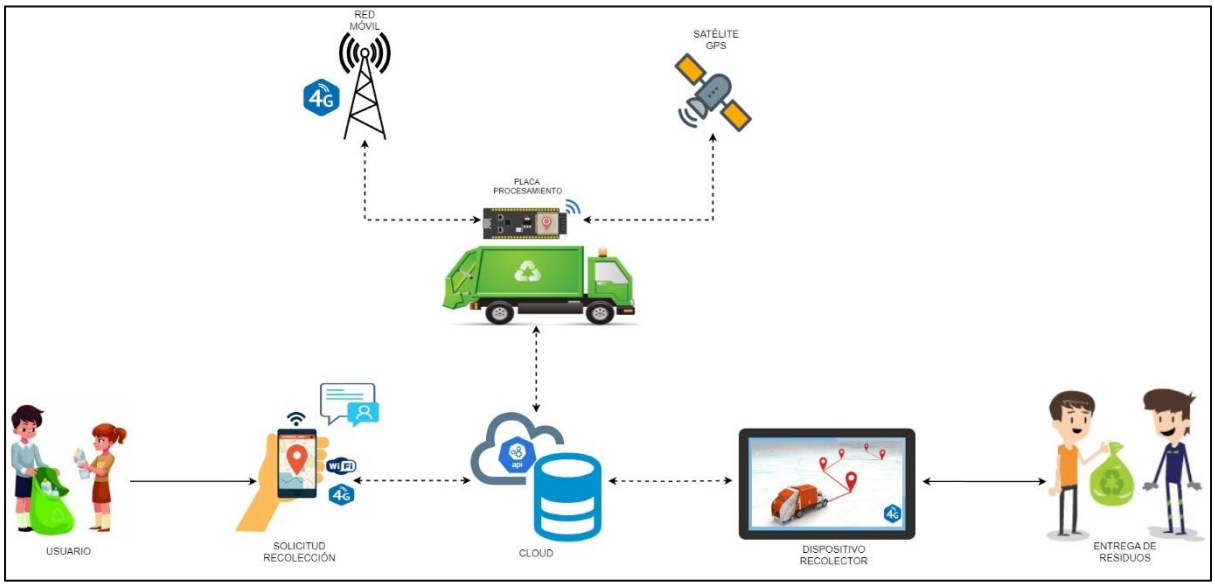
El Sistema de Recolección de RSU integra diversas tecnologías para optimizar la gestión de residuos en el Barrio Centenario. En la Figura 1 se represente gráficamente sus componentes principales que incluyen:

1. **Dispositivo de Geolocalización:** Instalado en el vehículo recolector, consta de una placa de procesamiento con módulos GPRS/GSM y GPS.
2. **Aplicación Móvil:** Diseñada con dos modos de ingreso de usuarios:
    - **Para Conductores:** Muestra puntos de recolección y rutas optimizadas.
    - **Para Clientes:** Permite enviar solicitudes de recolección y rastrear el vehículo en tiempo real, además de recibir notificaciones sobre el proceso de recolección.
3. **Servidor de Base de Datos:** Almacena y procesa la información de solicitudes y rutas.
4. **Servidor de Mapas:** Genera rutas eficientes basadas en las solicitudes recibidas.
5. **Sistema de Notificaciones:** Informa a los usuarios sobre el estado del servicio en tiempo real.

Este sistema permite una interacción dinámica entre usuarios y recolectores, mejorando la eficiencia del servicio y la satisfacción de los residentes del Barrio Centenario.

**Figura 1**

*Representación gráfica del Sistema de Recolección de RSU*



## 4. CRONOGRAMA DE ACTIVIDADES

En la Tabla 1, se especifica el cronograma de actividades del plan de implementación, mismo que se desarrollará en un período de 18 semanas, dividido en 8 fases

**Tabla 1**

*Cronograma de Implementación del Sistema de Recolección de RSU*

| Fase | Actividades Principales | Duración Actividad | Duración Fase |
|---|---|---|---|
| **FASE 1:** Preparación | - Reunión inicial con autoridades municipales | 2 días | 2 semanas |
| | - Definición del equipo de implementación | 3 días | |
| | - Revisión y ajuste final del cronograma | 5 días | |
| **FASE 2:** Infraestructura y Equipamiento | - Adquisición e instalación de hardware necesario | 8 días | 3 semanas |
| | - Configuración de nodo recolector GPS | 4 días | |
| | - Instalación de nodo recolector GPS en vehículo recolector | 3 días | |
| **FASE 3:** Despliegue de Software | - Instalación y configuración de servidores del sistema | 4 días | 2 semanas |
| | - Instalación y configuración de app móvil "ResiRutas" | 3 días | |
| | - Pruebas de conectividad y funcionamiento | 3 días | |
| **FASE 4:** Capacitación | - Sesiones para personal administrativo del GADM Montufar | 2 días | 2 semanas |
| | - Sesiones para conductores de los recolectores | 2 días | |

| | | | |
|---|---|---|---|
| | - Sesiones para habitantes del barrio Centenario | 3 días | |
| | - Elaboración y distribución de material informativo | 3 días | |
| **FASE 5:** Prueba Piloto | - Implementación en sector limitado del Barrio Centenario | 1 semana | 3 semanas |
| | - Monitoreo y ajustes del sistema | 1 semana | |
| | - Prueba de recolección para retroalimentación de funcionalidad | 1 semana | |
| **FASE 6:** Lanzamiento Completo | - Activación del sistema en todo el Barrio Centenario | 3 días | 1 semana |
| | - Campaña de comunicación a la ciudadanía | 4 días | |
| **FASE 7:** Seguimiento y Soporte | - Monitoreo continuo del rendimiento del sistema | 2 semanas | 4 semanas |
| | - Soporte técnico y resolución de incidencias | 1 semana | |
| | - Recopilación de métricas de consumo de los servidores | 1 semana | |
| **FASE 8:** Evaluación Final | - Análisis de datos y métricas recopiladas | 2 días | 1 semana |
| | - Reunión de cierre con autoridades municipales | 1 día | |
| | - Elaboración de informe final y recomendaciones | 2 días | |

## 5. COSTOS DEL PROYECTO

La implementación del Sistema de Recolección de RSU requiere una inversión inicial que se detalla en la Tabla 2.

**Tabla 2**

*Costos de Hardware y Software del Sistema de recolección de RSU*

| Ítem | Cantidad | Costo Unitario ($) | Costo Total ($) |
|---|---|---|---|
| **Costos de Hardware** | | | |
| Módulo Arduino T-SIM7600NA | 1 | 125.00 | 125.00 |
| Batería ICR 18650 2600 mAh | 2 | 7.50 | 15.00 |
| Shield | 1 | 20.00 | 20.00 |
| Dispositivos móviles para recolectores | 1 | 300.00 | 300.00 |
| **Costos de Software** | | | |
| API de Mapas (Mapbox) | 1 | 0(plan gratuito con restricciones) | 0(plan gratuito con restricciones) |
| Servicios de base de datos (Firebase) | 1 | 0(plan gratuito) | 0(plan gratuito) |
| Desarrollo de la aplicación móvil | 1 | 1000 | 1000 |
| **Costos Anual de Plan de datos** | | | |
| Plan de Conexión 10 | 1 | 10.25 | 123.00 |
| | | **Costo total de implementación:** | 1583.00 |

Aunque inicialmente los servicios de Mapbox y Firebase no tienen costo, ambos imponen límites de uso que, al superarse, generan gastos adicionales. A medida que la aplicación crezca, es probable que se active el plan de pago por uso de Firebase (Blaze) y se requieran planes de pago en Mapbox. Se recomienda monitorear constantemente el uso de estos servicios y ajustar el presupuesto según sea necesario para garantizar la sostenibilidad del proyecto a largo plazo.

De igual forma, la Tabla 3 desglosa los costos de capacitación y mantenimiento del sistema de recolección de RSU para el barrio Centenario.

**Tabla 3**

*Costos de Capacitación y Mantenimiento del sistema de recolección de RSU*

| Ítem | Costo ($) |
|---|---|
| Capacitación administrador sistema GADM Montufar | 300.00 |
| Capacitación de conductores de los recolectores | 200.00 |
| Capacitación a moradores del barrio Centenario | 500.00 |
| **Total:** | **$1000.00** |

En conjunto los costos de implementación totales se especifican en la Tabla 4, donde se observa una inversión inicial total de $3083. Esta inversión inicial cubre todos los aspectos necesarios para la puesta en marcha del sistema, incluyendo equipamiento, desarrollo de software, capacitación del personal y usuarios, y el primer año de mantenimiento.

**Tabla 4**

Costos totales de Implementación del Sistema de Recolección de RSU

| Categoría | Costo Total ($) |
|---|---|
| Hardware | 460.00 |
| Software | 1000.00 |
| Capacitación | 1000.00 |
| Mantenimiento Anual | 500.00 |
| Plan de Datos Anual | 123.00 |
| **Total Inversión Inicial** | **3083.00** |

## 6. PROCESO DE SEGUIMIENTO, SOPORTE Y EVALUACIÓN CONTINUA

Para garantizar el funcionamiento eficiente del sistema y su mejora continua, en la Tabla 5, se establece el proceso de Seguimiento, Soporte y Evaluación del Sistema.

**Tabla 5**

*Proceso de Seguimiento, Soporte y Evaluación del Sistema*

| Aspecto | Actividades | Frecuencia |
|---|---|---|
| Monitoreo Técnico | - Revisión de consumo del sistema | Diaria |
| | - Verificación de conectividad de dispositivos | Diaria |

| | | |
|---|---|---|
| | - Control de funcionamiento de servidores | Semanal |
| Soporte a Usuarios | - Atención de consultas y problemas reportados | Continua |
| | - Actualización de material de ayuda | Mensual |
| Evaluación de resultados | - Satisfacción de usuarios | Semestral |
| Mantenimiento | - Actualización de software | Trimestral |
| | - Verificación de precisión de dispositivos GPS | Mensual |
| Mejora Continua | - Análisis de datos y patrones de uso | Mensual |
| | - Reuniones de retroalimentación con involucrados | Trimestral |
| | - Implementación de mejoras basadas en análisis y retroalimentación | Según necesidad |

## 7. RECOMENDACIONES TÉCNICAS

Para garantizar una implementación y mantenimiento efectivos del sistema de recolección de RSU, es fundamental establecer un proceso de monitoreo continuo que permita alertar sobre posibles fallos antes de que estos impacten el servicio. Este monitoreo debe incluir la supervisión de la conectividad de los dispositivos GPS, el rendimiento de los servidores y la integridad de la base de datos. Además, se debe instaurar un calendario de mantenimiento preventivo que abarque todos los componentes del sistema, incluyendo tanto actualizaciones de software como revisiones de hardware, programando estas actividades en horarios de bajo impacto para el servicio.

Es importante asegurar que, la seguridad del sistema debe ser robusta, con medidas como la encriptación de datos en tránsito, autenticación para accesos administrativos y auditorías regulares de seguridad.

Respecto las capacitaciones, es importante desarrollar un programa de capacitación continua para el personal técnico y operativo, asegurando que estén actualizados con las últimas novedades del sistema y las mejores prácticas de operación.

Por último, debe desarrollarse un plan detallado de continuidad del servicio que contemple procedimientos de operación manual en caso de fallas del sistema, asegurando que la recolección de residuos no se vea interrumpida.

Se recomienda establecer un comité de mejora continua que se reúna regularmente permitirá analizar el rendimiento del sistema, evaluar la retroalimentación de los usuarios y proponer e implementar mejoras incrementales.

## 8. MANUALES DE USO DEL SISTEMA DE RECOLECCIÓN DE RSU

### 8.1. Manual de Administrador

# MANUAL DE ADMINISTRADOR

**“SISTEMA DE RECOLECCIÓN DE RESIDUOS SÓLIDOS URBANOS (RSU) BASADO EN OPTIMIZACIÓN DE RUTAS Y NOTIFICACIÓN EN TIEMPO REAL PARA EL BARRIO CENTENARIO, CIUDAD DE SAN GABRIEL”**

**AUTOR: CRISTIAN IVÁN QUELAL GUADIR**

**IBARRA-ECUADOR**
**2024**

# ÍNDICE DE CONTENIDOS

Este manual está diseñado para guiar a los administradores en el uso y gestión del Sistema de Recolección de Residuos Sólidos Urbanos (RSU) diseñado para el Barrio Centenario en la ciudad de San Gabriel. La gestión se puede llevar a cabo mediante las plataformas de Firebase y Mapbox, de esta forma el administrador puede visualizar las estadísticas de uso que se generan por los servicios de cada plataforma corroborando así que el sistema esté funcionando correctamente.

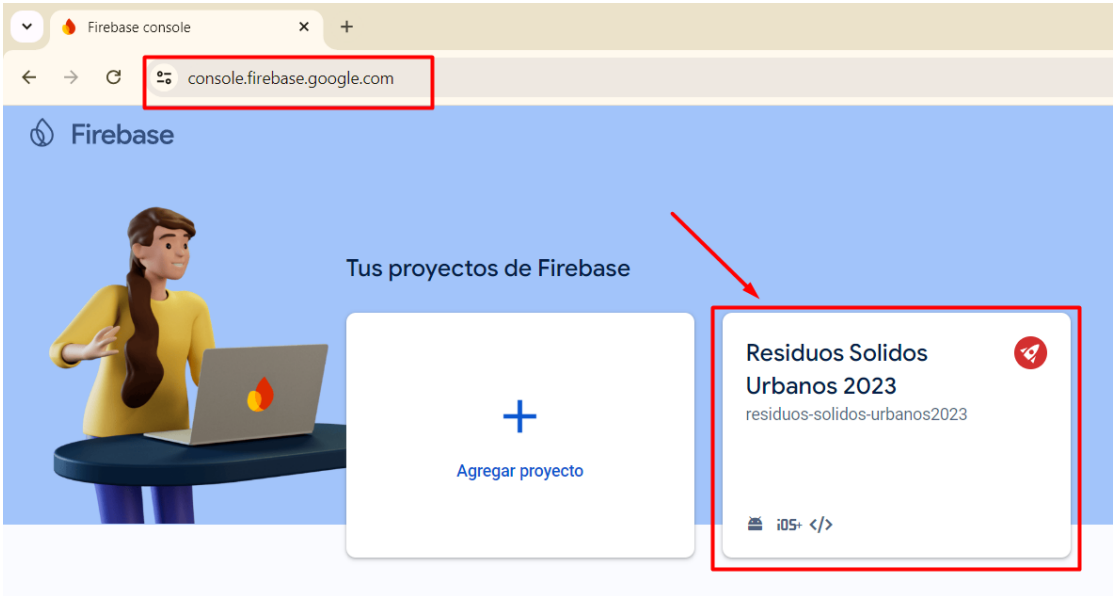
## 1. Acceso a la plataforma de Firebase

Para acceder al panel de administración del sistema, siga estos pasos:

1. Abra su navegador web y diríjase a la siguiente URL: https://console.firebase.google.com
2. Inicie sesión con las credenciales de administrador proporcionadas.
3. Una vez en la consola de Firebase, localice y seleccione el proyecto "Residuos Solidos Urbanos 2023" para ser redirigido al Panel de Administración como se muestra en la Figura 1.

**Figura 1**

*Interfaz inicial de la plataforma de Firebase*



### 1.1. Panel de Gestión

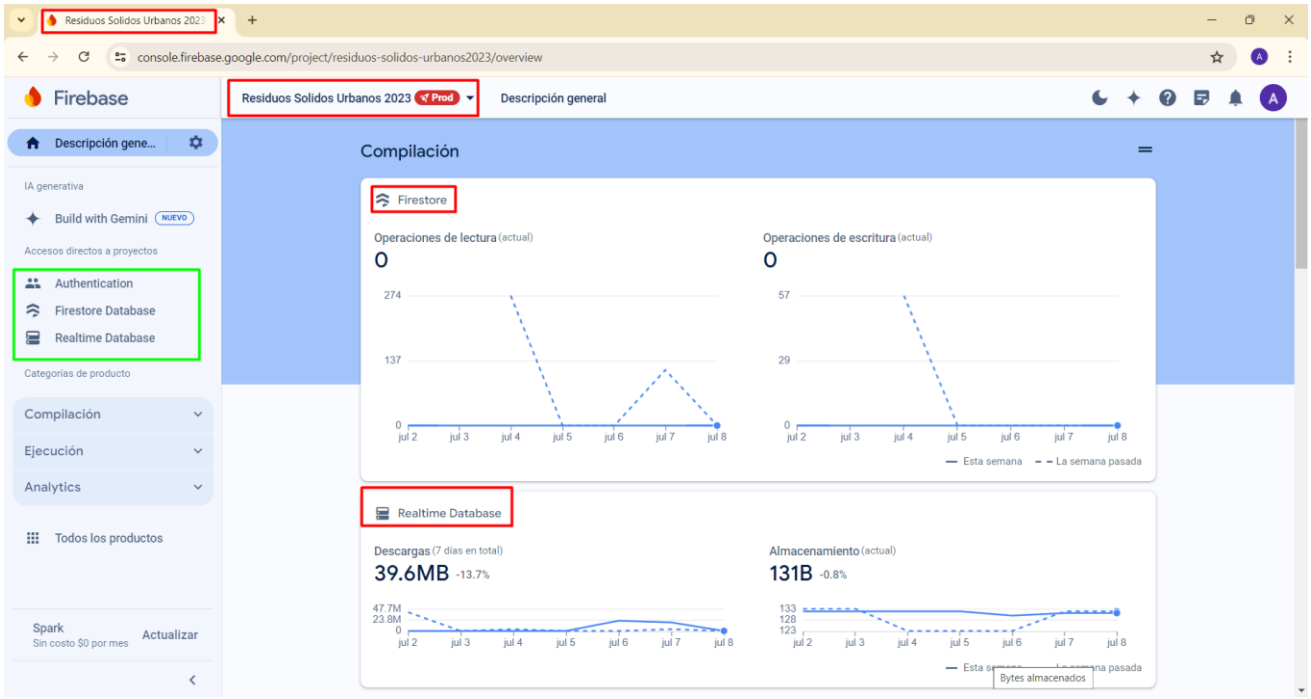
El panel de administración de Firebase proporciona diferentes secciones de servicios para gestionar el sistema como se muestra en la Figura 2. Estos se encuentran anclados y son:

- **Autenticación:** Gestión de usuario "Recolector"
- **Firestore Database:** Almacenamiento de los datos de solicitudes de recolección de residuos de clientes, además, visualización de estadísticas sobre su uso.
- **Realtime Database:** Almacenamiento y sincronización de datos en tiempo real, además, visualización de estadísticas sobre su uso.

Además, se puede visualizar gráficos y estadísticas importantes sobre el uso general de los servicios de base de datos. Estos datos le ayudarán a supervisar el consumo del almacenamiento de datos del sistema en las bases de datos y detectar posibles problemas.

**Figura 2**

*Interfaz inicial de la plataforma de Firebase*



## 2. Servicio de Autenticación: Gestión de Usuarios

En la Figura 3, se visualiza la sección de Autenticación que permite administrar los usuarios "Recolectores" del sistema:

- **Visualizar usuarios registrados y su información de Acceso**

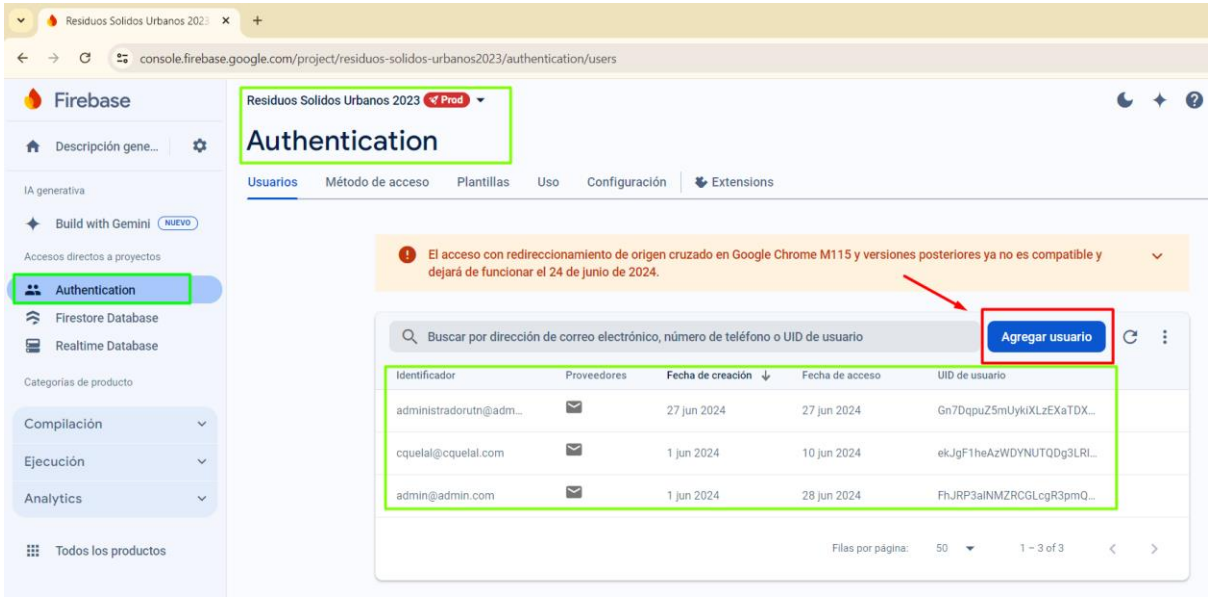
La lista de usuarios muestra información importante:

- ✓ **Identificador:** Correo electrónico del usuario.
- ✓ **Proveedores:** Método de autenticación utilizado.
- ✓ **Fecha de creación:** Cuándo se creó la cuenta.
- ✓ **Fecha de acceso:** Última vez que el usuario inició sesión.
- ✓ **UID de usuario:** Identificador único asignado por Firebase.

**Nota:** Recuerde siempre mantener la seguridad y privacidad de los usuarios al gestionar sus cuentas. Solo realice cambios cuando sea necesario.

**Figura 3**

*Interfaz del servicio de Autenticación*




### 2.1. Agregar nuevos usuarios

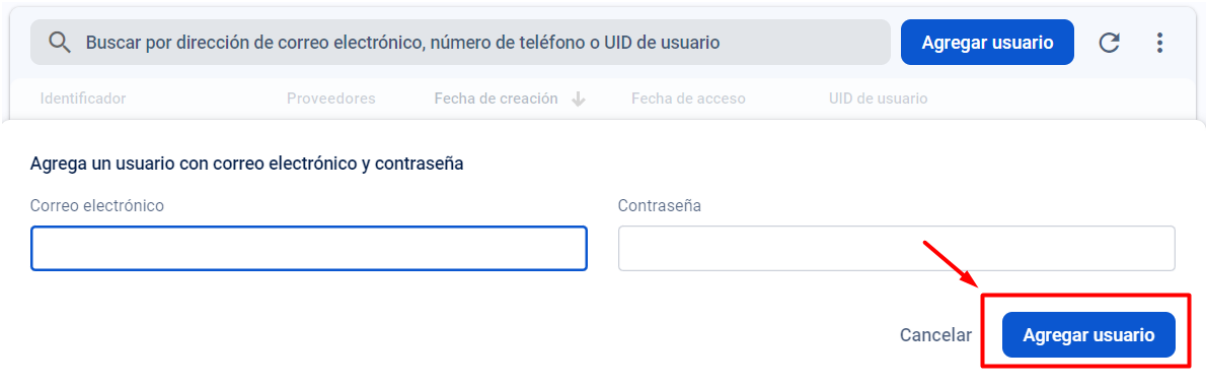
Para agregar un nuevo usuario al sistema:

1. En la parte superior de la pantalla, haga clic en el botón "Agregar usuario".

2. Se abrirá una ventana donde deberá ingresar:
   - ✓ Correo electrónico del nuevo usuario
   - ✓ Contraseña
3. Una vez ingresados los datos, haga clic en "Agregar usuario" para crear la cuenta como se visualiza en la Figura 4.

**Figura 4**

*Interfaz para agregar usuarios "Recolectores"*




### 2.2. Gestionar Usuarios Existentes

Para cada usuario existente, tiene las siguientes opciones de gestión:

- ✓ **Restablecer contraseña:** Permite generar una nueva contraseña para el usuario.
- ✓ **Inhabilitar cuenta:** Suspende temporalmente el acceso del usuario al sistema.
- ✓ **Borrar cuenta:** Elimina permanentemente la cuenta del usuario del sistema.

**Para acceder a estas opciones:**

1. Localice el usuario deseado en la lista.

**Para encontrar un usuario específico:**

- ✓ Utilice la barra de búsqueda en la parte superior de la lista de usuarios.
- ✓ Puede buscar por dirección de correo electrónico, número de teléfono o UID de usuario.

2. Haga clic en los tres puntos (...) al final de la fila del usuario.

3. Seleccione la acción que desea realizar del menú desplegable como se visualiza en la Figura 5.

**Figura 5**

*Opciones para gestionar usuarios "Recolectores"*


## 3. Gestión de la Base de Datos (Cloud Firestore)

Esta base de datos permite almacenar los datos de las solicitudes de recolección de los clientes del barrio Centenario.

Para acceder y gestionar los datos en Cloud Firestore como se muestra en la Figura 6:

1. En el menú lateral, seleccione "Firestore Database".

2. Use la barra de opciones para navegar entre **Datos** (ver los datos almacenados), **Reglas** (configuración de reglas de escritura y lectura), **Índices** (agregar índices para métodos de búsqueda de datos), **Uso** (supervisar el consumo del almacenamiento) y **Extensiones**(configuración de características para ampliación de funcionalidades de Firestore).

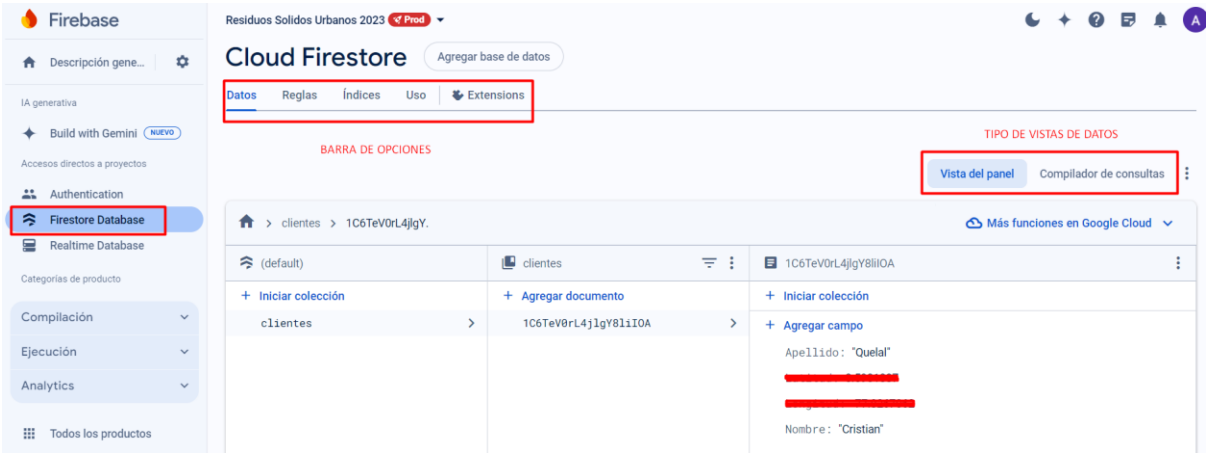
> **Nota:** Las pestañas de las opciones Reglas, Índices y Extensiones son exclusivas para configuración, por tanto, no deben ser modificadas ya que se encuentran configuradas de acuerdo con las necesidades actuales del proyecto, cualquier duda o necesidad de cambio se debe comunicar con el desarrollador del Sistema de recolección de RSU.

3. Puede alternar entre los Tipos de Vista de Datos: Vista de Panel o Compilador de Respuestas para visualizar los datos en Panel o Tablas respectivamente, según sea necesario.

**Nota:** No es necesario añadir manualmente una nueva colección, un documento o agregar campos, debido a que el sistema crea automáticamente cada campo a partir de los datos de las solicitudes de recolección de cada cliente.

**Figura 6**

*Interfaz de Cloud Firestore para el almacenamiento de datos de solicitudes de clientes*




## 4. Gestión de la Base de Datos (Realtime Database)

Esta base de datos permite almacenar los datos de ubicación del camión recolector en tiempo real.

Para acceder y gestionar los datos en **Realtime Database** como se muestra en la Figura 7:

1. En el menú lateral, seleccione " **Realtime Database** ".

2. Use la barra de opciones para navegar entre **Datos** (ver los datos almacenados en el panel de visualización), **Reglas** (configuración de reglas de escritura y lectura), **Copia de Seguridad** (crear una copia de seguridad de los datos si se actualiza un plan Premium),

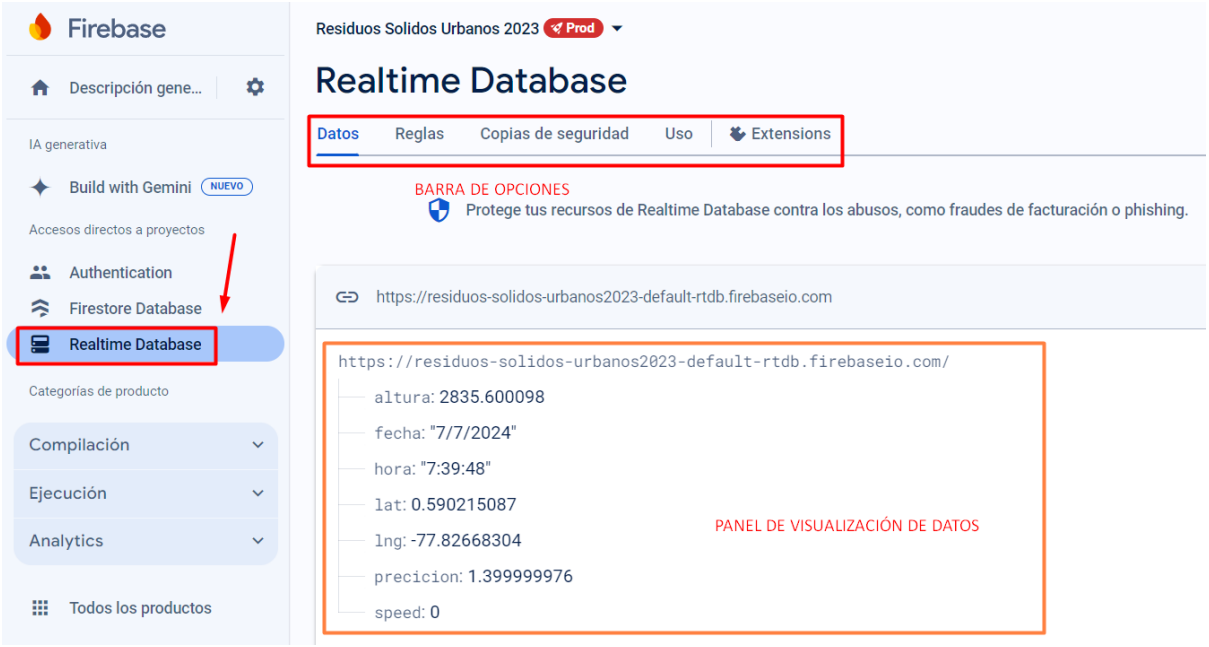
**Uso** (supervisar el consumo del almacenamiento) y **Extensiones**(configuración de características para ampliación de funcionalidades de Realtime Database).

**Nota:** Las pestañas de las opciones Reglas y Extensiones son exclusivas para configuración, por tanto, no deben ser modificadas ya que se encuentran configuradas de acuerdo con las necesidades actuales del proyecto, cualquier duda o necesidad de cambio se debe comunicar con el desarrollador del Sistema de recolección de RSU.

**Nota:** No es necesario añadir manualmente agregar datos en los diferentes campos, debido a que el sistema genera automáticamente cada campo con su valor a partir de los datos recibidos en tiempo real desde la placa de procesamiento instalada en el camión recolector.

**Figura 7**

*Interfaz de Realtime Database para el almacenamiento de datos de Geocalización*




## 5. Gestión de la plataforma del servicio de Mapas (Mapbox)

Este servicio permite procesar los datos de las bases de datos para visualizar los puntos de recolección en el mapa y generar rutas para la recolección.

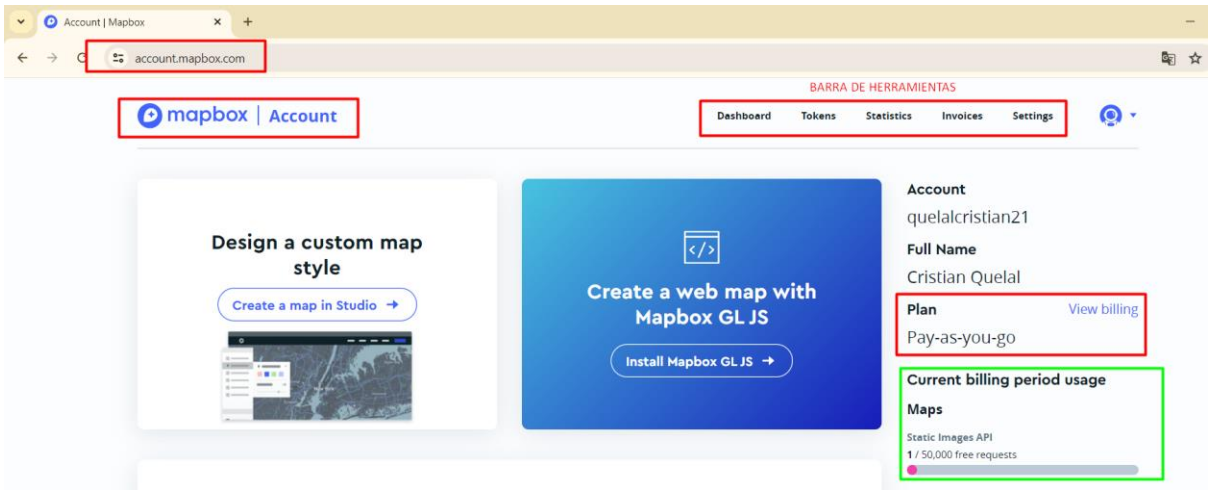
Para acceder al panel de administración del servicio de Mapas, siga estos pasos:

1. Abra su navegador web y diríjase a la siguiente URL: https://account.mapbox.com/
2. Inicie sesión con las credenciales de administrador proporcionadas.
3. Una vez en la Interfaz de Mapbox, localice y seleccione la herramienta que requiera en la "Barra de Herramientas" para ser redirigido al Panel correspondiente.

**Nota:** Los apartados de Tokens y Settings son específicos para el desarrollador del proyecto, por lo que no deben ser modificados. En la interfaz de Dashboard (Figura 8), se puede navegar por la barra de herramientas, visualizar estadísticas de consumo del servicio(Statistics) y acceder a las facturas generadas(Invoices). También muestra datos importantes como el nombre de la cuenta, el tipo de plan y un resumen del consumo de los servicios.

**Figura 8**

*Dashboard del servicio de mapas "Mapbox"*



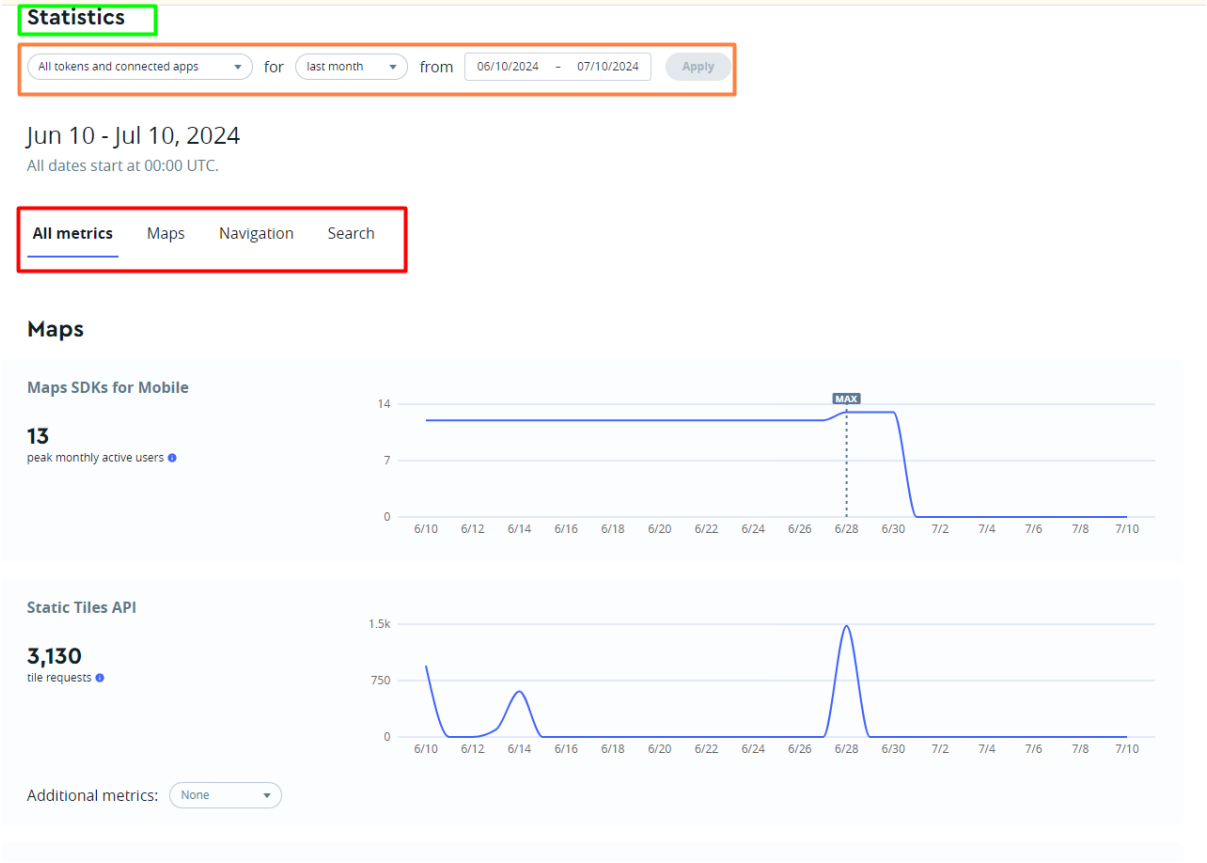

Para monitorear el consumo de los servicios de Mapas, utilice el panel de estadísticas (Statistics) de Mapbox, que muestra datos importantes (ver Figura 9), permitiendo así supervisar el uso adecuado de las funcionalidades de mapeo del sistema mediante:

- ✓ Rango de fechas seleccionable (Cuadro color tomate)

- ✓ Pestañas para elegir las diferentes métricas por categorías de los servicios de Mapbox en uso (Cuadro color rojo)
- ✓ Gráficos de uso de los servicios de Mapas

**Figura 9**

*Dashboard de monitoreo de consumo de los servicios de Mapbox*



**8.2. Manual de usuario**

# MANUAL DE USUARIO

**“SISTEMA DE RECOLECCIÓN DE RESIDUOS SÓLIDOS URBANOS (RSU) BASADO EN OPTIMIZACIÓN DE RUTAS Y NOTIFICACIÓN EN TIEMPO REAL PARA EL BARRIO CENTENARIO, CIUDAD DE SAN GABRIEL”**

**AUTOR: CRISTIAN IVÁN QUELAL GUADIR**

**IBARRA-ECUADOR**
**2024**

# ÍNDICE DE CONTENIDOS

## 1. Requerimientos de Instalación de la Aplicación Móvil “RESIRUTAS”

Los requerimientos mínimos para que la aplicación móvil “ResiRutas” funcione correctamente, son los siguientes:

a) Sistema operativo Android (se recomienda versión 12 o superior)
b) Almacenamiento disponible mínimo 250 MB
c) Memoria RAM base de 16 GB (recomendable)
d) Conectividad (3G, 4G y/o WiFi)
e) Ubicación activada

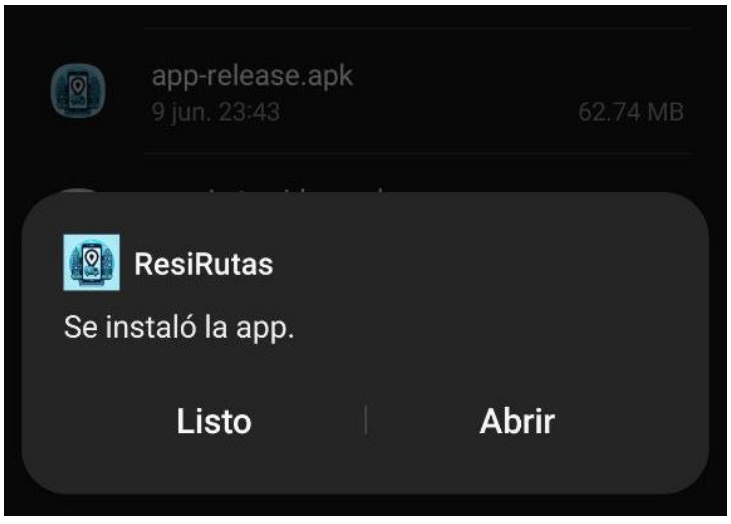
## 2. Proceso de Instalación de Aplicación Móvil “RESIRUTAS”

Para instalar la aplicación móvil del sistema de recolección de RSU, siga estos pasos:

1) Obtenga el archivo .apk de la aplicación (solicitar al administrador del sistema)
2) Localice el archivo .apk descargado en su dispositivo.
3) Toque el archivo para iniciar la instalación y siga las instrucciones en pantalla para completar la instalación
4) Una vez instalada como se visualiza en la Figura 1, abra la aplicación "ResiRutas"

**Figura 1**

*Instalación Exitosa de la aplicación móvil “ResiRutas”*



Al abrir la aplicación por primera vez, deberá conceder los permisos necesarios:

1) La aplicación solicitará permisos de ubicación y notificaciones. Toque permitir o mientras la app está en uso, cuando aparezca el mensaje de permisos como se visualiza en la Figura 2.

**Nota:** En el caso del permiso de ubicación, se le pedirá que otorgue permisos de ubicación. Elija "Precisa" para obtener una buena Geocalización.

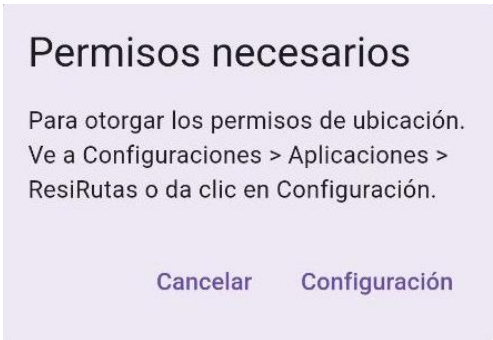
**Figura 2**

*Solicitud de permisos necesarios para la aplicación móvil "ResiRutas"*


Asegúrese de conceder todos los permisos solicitados para garantizar el funcionamiento correcto de la aplicación, de no ser asignado los permisos se visualiza un mensaje emergente como se visualiza en la Figura 3, luego se debe seguir los pasos que se indican para revisar y asignar los permisos requeridos.

**Figura 3**

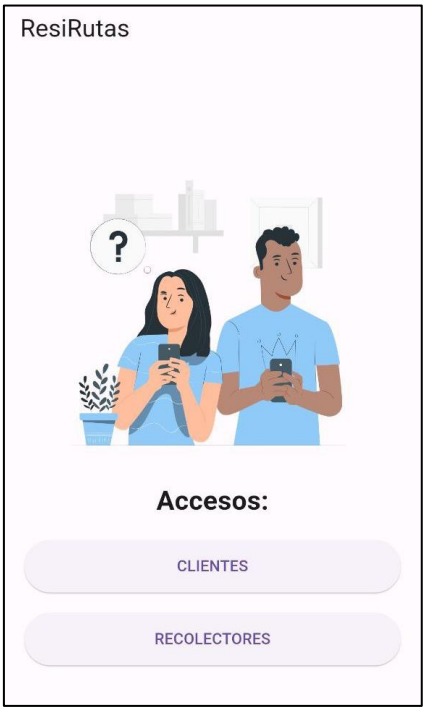
*Ventana emergente de notificación de permisos requeridos para la aplicación "ResiRutas"*




Una vez configurados los permisos, accederá a la pantalla principal donde podrá elegir entre ingresar como recolector o cliente.

**Figura 4**

*Interfaz para el acceso de un usuario determinado en la app "ResiRutas"*



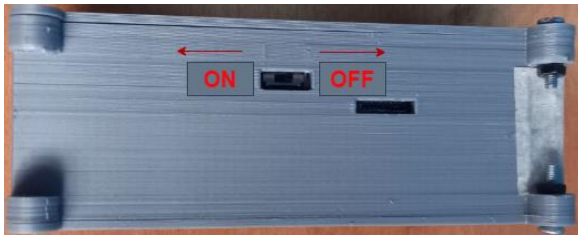

## 3. Guía: Usuario Recolector

**IMPORTANTE:** Encienda el dispositivo del nodo recolector ubicado en el vehículo 5 minutos antes de iniciar la recolección, como se muestra en la Figura 5, mueva el switch a la izquierda para encender y mueva a la derecha para apagar. El dispositivo tardará

aproximadamente 1 minuto en encenderse y enviar el primer dato de geolocalización del camión recolector. Espere 5 minutos para estabilizar el sistema de geolocalización.

**Figura 5**

*Representación de direcciones para encender y apagar dispositivo del nodo recolector*

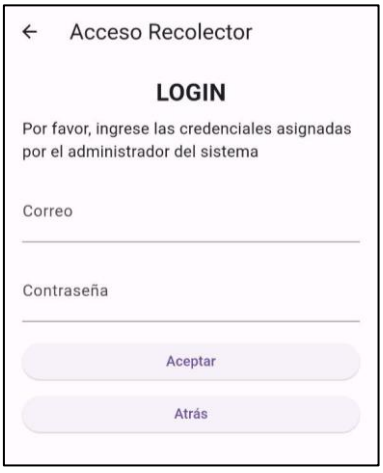



### 3.1. Login usuario “Recolector”

1) Abra la aplicación "ResiRuta".
2) En la pantalla principal, seleccione "Ingresar como Recolector".
3) En la interfaz de Login que se muestra en la Figura 6, introduzca las credenciales (correo y contraseña) proporcionadas por el administrador del sistema y presione el botón "Iniciar Sesión".

**Figura 6**

*Interfaz de Login para recolector*



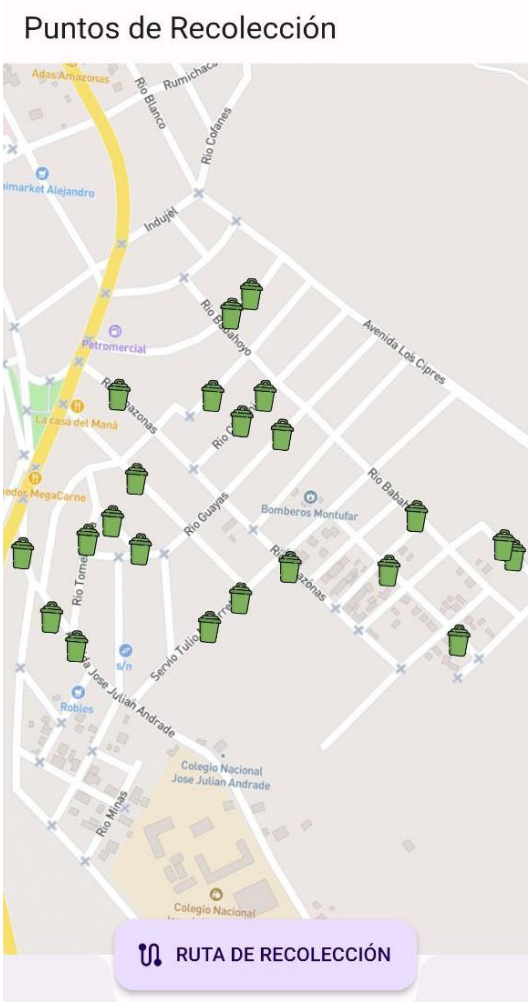

**Nota:** Si las credenciales son correctas, se visualizará la siguiente interfaz, caso contrario se emiten mensajes informando ingrese las credenciales correctas.

### 3.2. Guía de Interfaz: Puntos De Recolección

Una vez autenticado, se mostrará un mapa con los puntos de recolección registrados para el día como se visualiza en la Figura 7.

**Figura 7**

*Interfaz de puntos de recolección*




## 3.3. Proceso para generar la Ruta de Recolección

- **Para la generación de ruta de recolección:**

1) En la pantalla del mapa, pulse el botón "Generar Ruta".
2) Se mostrará una nueva interfaz con dos botones: "Comenzar la Navegación" y "Terminar Recorrido" como se visualiza en la Figura 8.

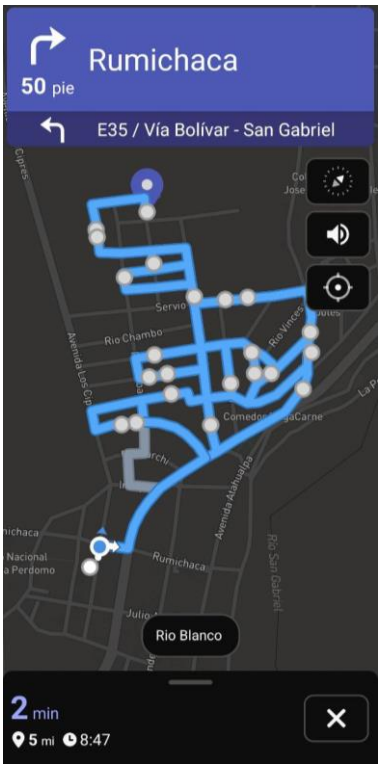
**Figura 8**

*Interfaz de puntos de recolección*


3) Pulse "Comenzar la Navegación".
4) La aplicación procesará la información y mostrará en el mapa la ruta optimizada para la recolección, considerando estadísticas de tráfico en la zona, vías alternas y guía de ruta por voz y visualmente como se visualiza en la Figura 9.

**Figura 9**

*Interfaz de navegación por la ruta de recolección de RSU.*



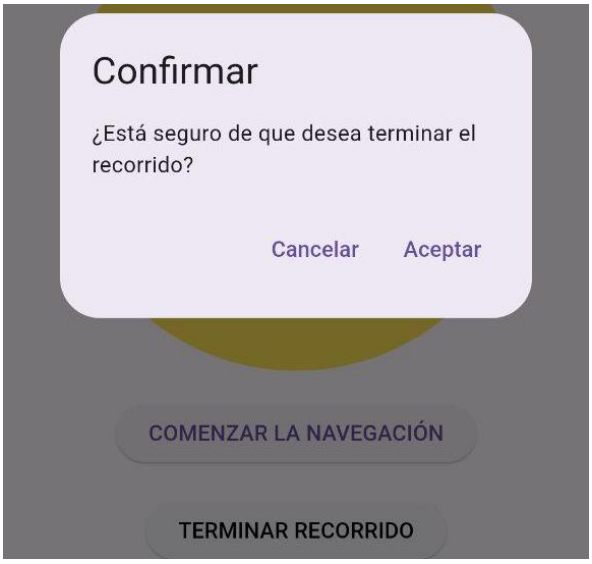

## 3.4. Proceso para finalizar el proceso de recolección

1) Una vez completada la recolección, regrese a la interfaz anterior.
2) Pulse el botón "Finalizar Ruta" y en la ventana emergente confirme el proceso(ver Figura 10)

**3)** Este proceso eliminará de la base de datos los registros de los clientes atendidos ese día

**Figura 10**

*Proceso de finalización de ruta de recolección de RSU*




## 4. Guía: Usuario Cliente

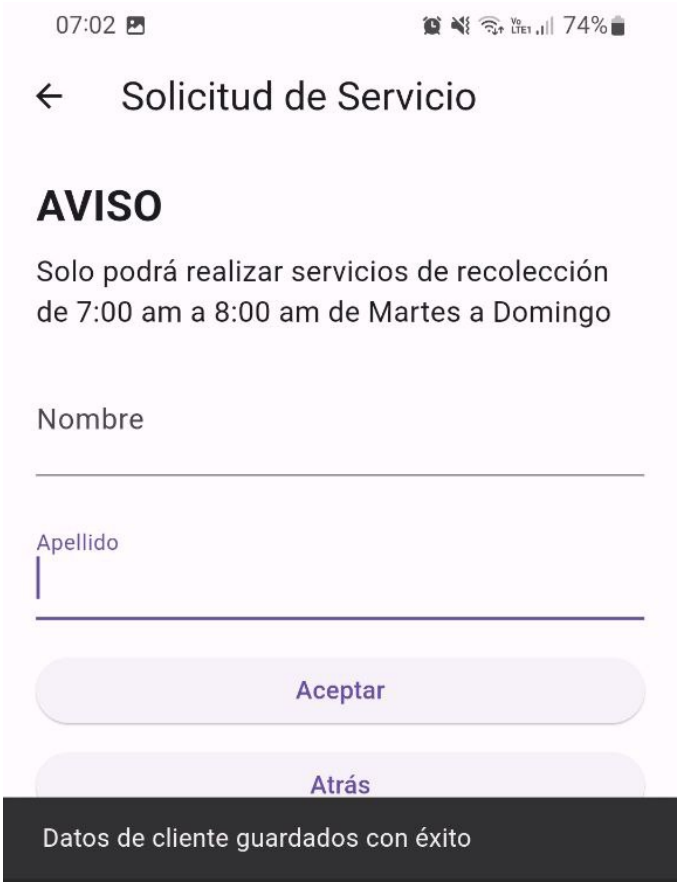
### 3.1. Proceso para generar una solicitud de recolección

1) Abra la aplicación "ResiRutas".
**2)** En la pantalla principal, seleccione "Ingresar como Cliente".
   - ✓ La aplicación verifica automáticamente si está dentro del horario de servicio (martes a domingo, de 7:00 a.m. a 8:00 a.m.). Si está dentro del horario, el botón "Enviar Solicitud" se habilitará.
3) Introduzca su nombre en el campo "Nombre" y apellido en el campo "Apellido"
4) Pulse el botón "Aceptar" para enviar la solicitud.
   - ✓ La aplicación capturará automáticamente las coordenadas de su ubicación actual y se enviarán a la base de datos en conjunto con los datos del cliente, si este proceso es exitoso se mostrará un banner con el mensaje de éxito, como se visualiza en la Figura 11, en caso contrario se generará un banner de con un mensaje de error.

**Nota:** Asegúrese de tener activada la ubicación en su dispositivo y haber otorgado los permisos correspondientes durante la instalación.

**Figura 11**

*Interfaz para la generación de solicitud de recolección de RSU*



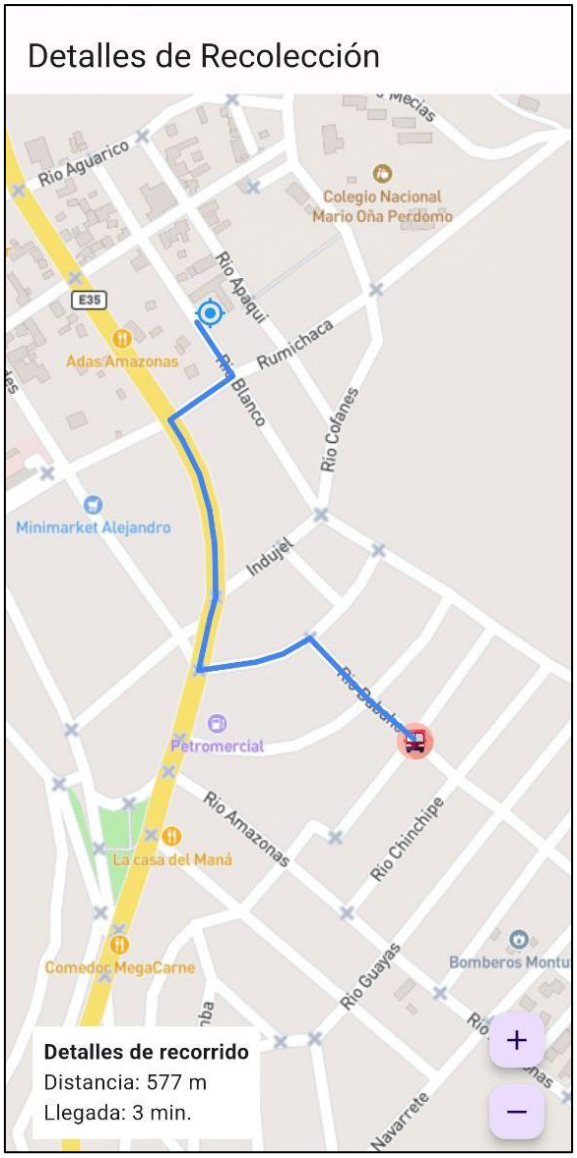

### 3.2. Guía de interfaz de visualización del estado del servicio de recolección

Después de enviar la solicitud, se mostrará un mapa como se visualiza en la Figura 12 con:

- La ubicación del recolector
- La distancia entre usted y el recolector
- El tiempo aproximado de llegada del recolector a su ubicación

**Figura 12**

*Interfaz de* visualización del estado del proceso de recolección



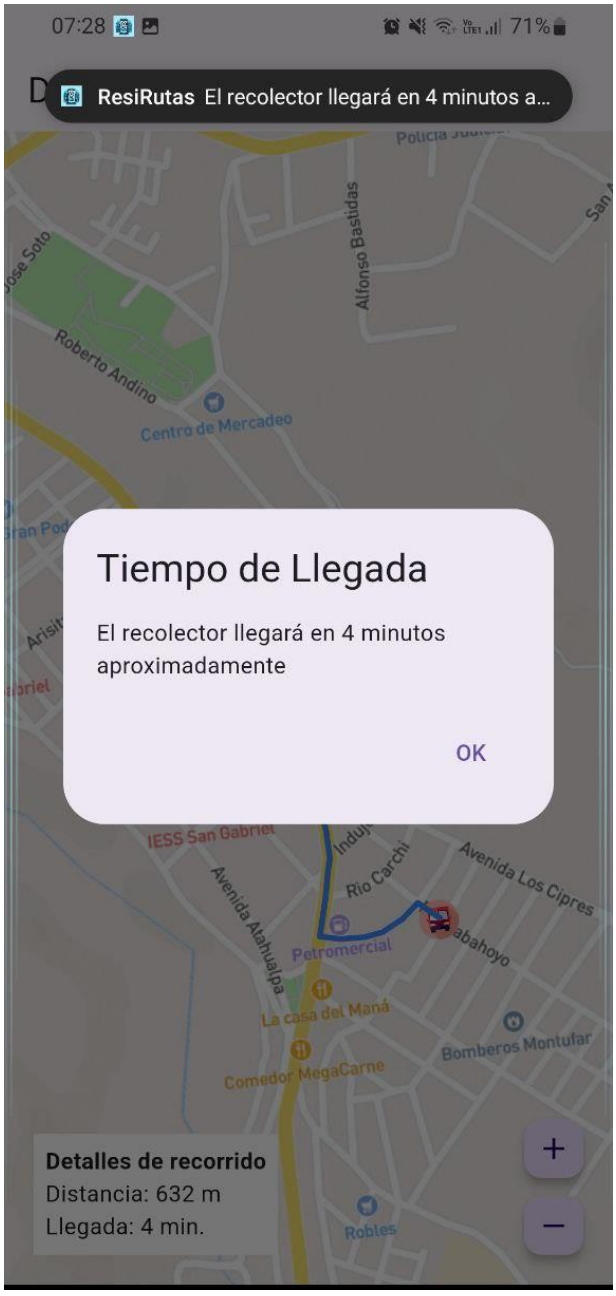

### 3.3. Guía del sistema de notificaciones sobre el estado del servicio

Durante el proceso de recolección y mientras no cierre la aplicación, recibirá notificaciones importantes sobre el proceso de recolección (ver Figura 13), como:

- Tiempo estimado de llegada
- Cercanía del recolector (menor o igual a 200 metros de distancia recolector-punto d recolección)
- Aviso de arribo del recolector al punto de recolección (menor a 10 metros)

**Figura 13**

*Sistema de notificaciones del estado del proceso de recolección*




**IMPORTANTE:** Recuerde que este servicio solo está disponible en el horario especificado.

Fuera de este horario, el botón para enviar la solicitud no estará disponible.