

**Universidad Nacional del Este.
Facultad Politécnica.**



**Sistema de rastreo e
identificación para motocicletas
empleando tecnología IoT.**

**José Mario Barreto Jara.
Mackey Ryuto Sugawara Mochinaga.**

Año 2025.

Universidad Nacional del Este.
Facultad Politécnica.

Carrera Ingeniería de Sistemas.
Cátedra Trabajo Final de Grado.

Sistema de rastreo e identificación para motocicletas empleando tecnología IoT.

**Por: José Mario Barreto Jara
y Mackey Ryuto Sugawara Mochinaga**

Profesor Orientador: **Ing. MSc. Daisy Isabel Kang Cardozo**
Profesor Coorientador: **Ing. René Andres Ayoroa Martínez**

Trabajo final de grado presentado a la Facultad Politécnica de la Universidad Nacional del Este como parte de los requisitos para optar al título Ingeniero de Sistemas.

Ciudad del Este, Alto Paraná. Paraguay.

Año 2024

FICHA CATALOGRÁFICA
BIBLIOTECA DE LA FACULTAD POLITÉCNICA
DE LA UNIVERSIDAD NACIONAL DEL ESTE

Barreto Jara,
José Mario, 2000.
Sugawara Mochinaga,
Mackey Ryuto, 2000.
Sistema de rastreo e identificación para motocicletas empleando tecnología IoT.
Orientador: Ing. Daisy Isabel Kang Cardozo.
Ciudad del Este, Alto Paraná. Año 2024.
Páginas: 118.

Área de estudio: Automatización, Sistemas Embebidos.
Carrera: Ingeniería de Sistemas.
Titulación: Ingeniería de Sistemas.
Trabajo Final de Grado. Universidad Nacional del Este,
Facultad Politécnica.

Descriptores: 1. Radio frecuencia, 2. Geolocalización, 3. LoRaWAN.
Tracking and identification system for motorcycles using IOT technology
Key words: 1. Radio frequency, 2. Geolocation, 3. LoRaWAN.

Nosotros, Daisy Isabel Kang Cardozo, documento de identidad No. 4.202.423, Profesor Orientador y René Andres Ayoroa Martínez, documento de identidad No. 2.904.591, Profesor Coorientador del TFG titulado “*Sistema de rastreo e identificación para motocicletas empleando tecnología IoT.*”, de los alumnos José Mario Barreto Jara, documento de identidad No. 5.931.713, y Mackey Ryuto Sugawara Mochinaga, documento de identidad No. 4.791.872 de la carrera Ingeniería de Sistemas de la Facultad Politécnica de la Universidad Nacional del Este; certifico que el mencionado Trabajo Final de Grado ha sido realizado por dichos alumnos, de lo cual damos fe y en nuestra opinión reúne las condiciones para su presentación y defensa ante la Mesa Examinadora designada por la institución.

Ciudad del Este, _____ de _____ de 2024

Daisy Isabel Kang Cardozo

René Andres Ayoroa Martínez

Nosotros, los miembros de la Mesa Examinadora del Trabajo Final de Grado titulado “*Sistema de rastreo e identificación para motocicletas empleando tecnología IoT.*”, de la carrera Ingeniería de Sistemas de la Facultad Politécnica de la Universidad Nacional del Este, hacemos constar que el citado trabajo ha sido evaluado en fondo y forma por esta Mesa, la que por _____ ha resuelto asignar la calificación _____

Ciudad del Este, _____ de _____ de 2024

Profesor _____
Presidente de la Mesa Examinadora

Profesor _____

Miembro de la Mesa Examinadora

Profesor _____

Miembro de la Mesa Examinadora

Hoja de Aprobación

**Universidad Nacional del Este
Facultad Politécnica**

**Sistema de rastreo e identificación para
motocicletas empleando tecnología IoT**

Trabajo de Fin de Grado presentado por:

**José Mario Barreto Jara
Mackey Ryuto Sugawara Mochinaga**

Tribunal Examinador:

**Prof. [Nombre del Director]
Director**

Prof. [Nombre del Examinador]
Examinador

Prof. [Nombre del Examinador]
Examinador

Aprobado en: _____ de _____ de 2025

Quiero dedicar este logro a las personas que han sido fundamentales en mi vida y que han sido mi apoyo incondicional. Este logro es también de ustedes. Gracias por ser parte de mi historia.

Quiero expresar mi más sincero agradecimiento a todos los funcionarios de la Facultad Politécnica - UNE por su constante apoyo. Agradezco a los docentes, bibliotecarios y personal de limpieza por haber creado un entorno propicio para mi estudio. A mis amigos y compañeros de carrera, gracias por su amistad y apoyo incondicional.

“Si crees que puedes, o crees que no puedes, tienes razón. La clave está en creer en uno mismo; esa confianza es lo que nos lleva al éxito.” - Henry Ford

Resumen

Actualmente, la inseguridad y el robo de motocicletas representan una preocupación creciente en Paraguay, especialmente para las personas que dependen de este medio de transporte para su vida cotidiana. En respuesta a esta problemática, se desarrolló un sistema de seguridad y rastreo para motocicletas utilizando tecnologías de *IoT*, integrando componentes como *RFID* para la autenticación de usuarios, *LoRaWAN* para la transmisión de datos y la red *Helium* como infraestructura de conectividad. Este sistema permite que solo usuarios autorizados puedan encender y utilizar la motocicleta, y facilita el monitoreo de su ubicación en tiempo real para favorecer su recuperación en caso de robo.

El objetivo del sistema es permitir la autenticación segura del usuario mediante *RFID* y habilitar el arranque del motor solo para usuarios autorizados, además de rastrear en tiempo real la ubicación de la motocicleta y enviar alertas en caso de intento de robo. La metodología incluyó el diseño de un prototipo de hardware, el desarrollo de *firmware* para la interacción entre módulos de autenticación, geolocalización y transmisión de datos, y pruebas en entornos controlados y en campo. Las pruebas realizadas, tanto en laboratorio como en campo, demostraron una tasa de éxito del 95.56 %, destacando la estabilidad y efectividad del sistema incluso en movimiento. Comparado con métodos de seguridad tradicionales, esta solución combina autenticación continua y monitoreo en tiempo real, mostrando su viabilidad como alternativa para la seguridad vehicular.

Descriptores: Seguridad vehicular, *IoT*, *RFID*, *LoRaWAN*, *Helium*, Geolocalización .

Abstract

Currently, insecurity and motorcycle theft represent a growing concern in Paraguay, especially for people who rely on this means of transport for their daily lives. In response to this issue, a security and tracking system for motorcycles was developed using IoT technologies, integrating components such as RFID for user authentication, LoRaWAN for data transmission, and the Helium network as connectivity infrastructure. This system allows only authorized users to start and operate the motorcycle and facilitates real-time location monitoring to aid in recovery in case of theft.

The objective of the system is to enable secure user authentication through RFID and to allow motor start-up only for authorized users, as well as to track the motorcycle's location in real-time and send alerts in case of attempted theft. The methodology included the design of a hardware prototype, firmware development for interaction between authentication, geolocation, and data transmission modules, and testing in controlled and field environments. Tests conducted in both laboratory and field environments demonstrated a 95.56 % success rate, highlighting the system's stability and effectiveness, even during movement. Compared to traditional security methods, this solution combines continuous authentication and real-time monitoring, proving its feasibility as an alternative for vehicle security.

Keywords: Vehicle security, IoT, RFID, LoRaWAN, Helium, Geolocation.

Índice general

Resumen	IX
Abstract	X
Índice de Figuras	XI
Índice de Tablas	XII
Acrónimos y símbolos	XII
1. Introducción	1
1.1. Motivación	2
1.2. Definición del problema	3
1.3. Objetivos	3
1.3.1. Objetivo General	3
1.3.2. Objetivos Específico	3
1.4. Hipótesis	4
1.5. Justificación	4
1.6. Delimitación del alcance del trabajo	5
1.7. Descripción de los contenidos por capítulo.	5
2. Conceptos fundamentales, teoría y antecedentes	7
2.1. Internet de las cosas	7
2.2. LoRA (<i>Long Range</i>)	9
2.3. LoRaWAN (<i>Long Range Wide Area Network</i>)	10
2.4. Helium Networks	15
2.5. RFID	15
2.6. Sistema de Geolocalización	20
2.7. Servicios de Entrega y Logística	20
2.8. Antecedentes	21
2.8.1. Lokalisering av sensorer med LoRaWAN på Kalmar Länssjukhus (Año 2021)	21

2.8.2.	LoRaWAN for tracking inland routes of plastic waste: Introducing the smart TRACKPLAST bottle (Año 2023)	22
2.8.3.	Smart Security System for Two-Wheelers (Año 2020) .	23
2.8.4.	Smart Security System for Vehicles using Internet of Things (IoT) (Año 2018)	24
2.8.5.	Development of an IoT-Based (LoRaWAN) Tractor Trac- king System (2022)	25
2.8.6.	Use of RFID Technology as a Reporting Mechanism in Vehicle Tracking System (2016)	26
3.	Tecnologías utilizadas	28
3.1.	Componentes de <i>Hardware</i>	28
3.1.1.	Placa de Desarrollo WiFi LoRa 32 (V3)	28
3.1.2.	Lector RFID MFRC522	31
3.1.3.	Módulo GNSS GP-02 Kit	32
3.1.4.	Placa de Desarrollo ESP-WROOM-32	33
3.2.	Componentes de <i>Software</i>	34
3.2.1.	<i>Helium Console</i>	34
3.2.2.	Thingsboard	36
3.2.3.	Arduino IDE	40
3.2.4.	Fritzing	41
3.2.5.	Blender	42
4.	Método	44
4.1.	Definición de arquitectura	44
4.1.1.	Requisitos clave	45
4.1.2.	Estructura interna del prototipo	45
4.1.3.	Arquitectura de comunicación externa	46
4.2.	Módulo de autenticación y control	47
4.2.1.	Requisitos del sistema	47
4.2.2.	Conexión del módulo <i>RFID</i>	48
4.2.3.	Primera versión del código: implementación básica en el bucle principal	48
4.2.4.	Segunda versión del código: integración de FreeRTOS y modularización	49
4.3.	Módulo de captura y transmisión de datos	50
4.3.1.	Implementación básica para obtención de datos de las coordenadas en tiempo real	51
4.3.2.	Implementación y configuración del módulo LoRaWAN para la transmisión de datos	53

4.3.3. Implementación de la comunicación entre los módulos <i>GNSS</i> y <i>LoRaWAN</i>	55
4.4. Módulo de interfaz gráfica y monitoreo	57
4.4.1. Configuración de <i>ThingsBoard</i>	57
4.4.2. Integración de <i>ThingsBoard</i> con <i>Helium</i>	64
4.5. Integración de componentes y encapsulamiento	68
4.5.1. Implementación Básica con Integración Inicial de los Módulos RFID y LoRaWAN	68
4.5.2. Implementación alternativa con la Integración ajustada entre los Módulos.	68
4.5.3. Incorporación de Corte de Energía y de Arranque	71
4.5.4. Modelado y Diseño de Encapsulado 3D.	72
5. Pruebas y resultados	76
5.1. Pruebas en entornos controlado	76
5.1.1. Pruebas del Módulo de autentificación y control	77
5.1.2. Pruebas del Módulo de captura y transmisión de datos	79
5.1.3. Pruebas del Módulo de Interfaz Gráfica y Monitoreo .	87
5.1.4. Integración del sistema completo	92
5.2. Pruebas de Campo	95
5.2.1. Evaluación Inicial	95
5.2.2. Pruebas en Movimiento	96
5.2.3. Resultados Cuantitativos	97
5.2.4. Interpretación de los Resultados	99
5.3. Prototipo del sistema desarrollado	99
5.3.1. <i>Hardware</i> del prototipo	99
6. Conclusión	104
6.1. Discusión	104
6.2. Análisis de la hipótesis	105
6.3. Principales logros alcanzados	105
6.4. Sugerencias para futuras investigaciones	106
Glosario	107
Anexo	108

Índice de figuras

2.1.	<i>Internet de las Cosas</i> (crédito: Imagen por macrovector en <i>Freepik</i>).	9
2.2.	Estructura de capas de <i>LoRa</i> y <i>LoRaWAN</i> [?].	11
2.3.	Una arquitectura típica de red <i>LoRaWAN</i> [?].	12
2.4.	Clases de Dispositivos en <i>LoRaWAN</i> [?].	13
2.5.	Diagrama de Seguridad en <i>LoRaWAN</i> [?].	14
2.6.	Modelo de Sistema <i>RFID</i> [?].	16
2.7.	Modelo de comunicación entre el lector y los tags en un sistema <i>RFID</i> [?].	17
2.8.	Flujo de datos en un sistema <i>RFID</i> basado en el modelo maestro-esclavo [?].	18
3.1.	Placa de desarrollo Heltec	29
3.2.	Lector <i>RFID</i>	31
3.3.	Módulo de desarrollo <i>GP-02 Kit</i>	33
3.4.	Placa de desarrollo <i>ESP32</i>	34
3.5.	Diagrama de Arquitectura ThingsBoard	37
3.6.	Cadena de reglas raíz ThingsBoard	38
3.7.	Interfaz del Arduino IDE 2 [?].	41
3.8.	Entorno de diseño de Fritzing mostrando el esquema de una placa [?].	42
3.9.	Entorno de desarrollo en Blender.	43
4.1.	Fases principales del sistema.	44
4.2.	Estructura del prototipo.	45
4.3.	Arquitectura de comunicación y monitoreo.	46
4.4.	Diagrama de flujo de autenticación y control.	47
4.5.	Diagrama de conexiones entre WiFi LoRa 32 V3 y RFID RC522.	48
4.6.	Diagrama de conexiones entre WiFi LoRa 32 V3 y GNSS. . .	51
4.7.	Entorno para crear dispositivo y configuración de las credenciales necesarias para el nodo.	54

4.8.	Ajuste de los parámetros necesarios en el dispositivo para la comunicación con el servidor <i>LoRaWAN</i>	55
4.9.	Diagrama de flujo de transmisión de datos por <i>LoRaWAN</i>	55
4.10.	Detalles del dispositivo ThingsBoard.	58
4.11.	Perfil de dispositivo ThingsBoard.	59
4.12.	Cliente con usuario creado en ThingsBoard.	59
4.13.	Asignación de dispositivos a clientes en ThingsBoard.	60
4.14.	<i>Widgets</i> disponibles en <i>ThingsBoard</i>	61
4.15.	<i>Widgets</i> utilizados en <i>ThingsBoard</i>	62
4.16.	Configuraciones en <i>Widgets</i> de Mapa.	63
4.17.	Diagrama del motor de reglas <i>MQTT</i> en <i>ThingsBoard</i>	64
4.18.	Menú de integraciones en la consola de <i>Helium</i>	65
4.19.	Detalles de conexión <i>MQTT</i> configurados en la consola de <i>Helium</i>	66
4.20.	Flujo de integración entre el dispositivo PruebaHeltec y <i>MQTT_TB</i>	66
4.21.	Configuraciones en el <i>router</i> de la red local.	67
4.22.	Diagrama de conexiones entre RFID RC522, ESP-WROOM-32, WiFi LoRa 32 V3 y GNSS.	69
4.23.	Diagrama de conexiones de todos los módulos.	72
4.24.	Disposición estructural de los componentes.	73
4.25.	Base del encapsulado.	74
4.26.	Parte superior del encapsulado.	74
4.27.	Modelado 3D del encapsulado para el módulo RFID.	75
5.1.	Prueba del módulo RFID.	78
5.2.	Pruebas del módulo GNSS.	79
5.3.	Comprobación de las coordenadas del módulo GNSS.	80
5.4.	Comprobación con teléfono convencional.	81
5.5.	Medición de distancias con Google Earth - Coordenadas específicas.	82
5.6.	Distancia calculada con Omni Calculator.	82
5.7.	Pruebas de ubicación en Google Earth con múltiples puntos adquiridos.	83
5.8.	Visualización en la consola helium, mensaje de conexión y subida.	83
5.9.	Visualización en la consola helium.	84
5.10.	Visualización de cobertura en la consola helium.	84
5.11.	Visualización en la página helium maps.	85
5.12.	Estructura del Mensaje enviado, donde payload serían las coordenadas en Base64.	86

5.13. Interpretación de Base64 a Hexadecimal.	87
5.14. Interpretación de las coordenadas de latitud y longitud transmítidas.	87
5.15. Visualización de la ubicación en el mapa <i>OpenStreet</i> con trayectoria.	88
5.16. Series temporales de telemetría.	89
5.17. Prueba de decodificación de mensajes en formato <i>Base64</i> a valores hexadecimales.	90
5.18. Prueba de interpretación de valores hexadecimales según su tamaño.	91
5.19. Uso de <i>Mosquitto</i> para simular mensajes durante las pruebas. .	91
5.20. Integración Helium-ThingsBoards.	92
5.21. Visualización en la consola Helium de los paquetes de datos recibidos del UID.	94
5.22. Visualización en la consola Helium de los paquetes de datos recibidos de las coordenadas.	95
5.23. Diagrama esquemático de conexiones.	96
5.24. Trayectoria registrada durante las pruebas de campo.	97
5.25. Disposición estructural.	100
5.26. Encapsulado 3D.	101
5.27. Encapsulado 3D para el módulo <i>RFID</i>	101
5.28. Prototipo Final.	102
5.29. Visualización de datos en tiempo real.	103
6.1. Validación del Sistema Integrado en la Motocicleta.	108
6.2. Proceso de impresión del encapsulado.	109
6.3. Proceso de medición de continuidad en el sistema eléctrico de una motocicleta.	109

Índice de Tablas

3.1.	Tabla comparativa entre Placas de Desarrollo	30
3.2.	Tabla comparativa entre módulos <i>RFID</i>	32
3.3.	Comparativa de plataformas LoRaWAN	36
3.4.	Tabla comparativa de Plataformas IoT	39
5.1.	Resultados de las pruebas de detección y comparación del UID	78
5.2.	Resultados de las Pruebas por Versión del Código	94
5.3.	Resultados de las pruebas de campo	98

Acrónimos y símbolos

π raz la circunferencia del culo a su ditro. 5

LAN Local Area Network. 9, 10, 12

SOR Sistema Operativo de Red. 10, 11

SVM Support Vector Machine. 5

Capítulo 1

Introducción

En los últimos años, el robo de motocicletas ha aumentado de manera alarmante en Paraguay, afectando especialmente a los trabajadores de entrega que dependen de este medio de transporte para su sustento diario. Este problema no solo representa una pérdida económica significativa, sino también un riesgo para la seguridad personal y la estabilidad laboral de los afectados. Frente a esta situación, surge la necesidad de desarrollar soluciones innovadoras y accesibles que permitan mejorar la seguridad y facilitar la recuperación de motocicletas robadas. La presente trabajo se centra en el desarrollo de un sistema de seguridad antirrobo para motocicletas, empleando tecnologías como Identificación por Radiofrecuencia (RFID), Red de Área Amplia de Largo Alcance (LoRaWAN), integradas en el contexto del Internet de las Cosas (IoT). El IoT ha revolucionado la manera en que los dispositivos se comunican e interactúan entre sí, permitiendo la creación de redes inteligentes y eficientes. Las redes de Radiofrecuencia (RF) están experimentando un crecimiento significativo en su uso debido a su capacidad para conectar una amplia variedad de dispositivos a largas distancias con bajo consumo de energía [?]. Dentro del marco del IoT, la tecnología RFID juega un papel crucial en la identificación y autenticación de usuarios. RFID es una tecnología de identificación por radiofrecuencia que permite el reconocimiento de objetos y personas mediante el uso de etiquetas y lectores RFID. Esta tecnología se caracteriza por su capacidad para almacenar información en un microchip conectado a una antena, que puede ser leído por un dispositivo lector, permitiendo la identificación automática sin intervención humana y minimizando errores [?]. Para la comunicación y transmisión de datos en el sistema, se utiliza la tecnología LoRaWAN. LoRaWAN es una red de área amplia de baja potencia que utiliza la modulación Largo Alcance (LoRa) para la transmisión de datos a largas distancias con un bajo consumo de energía. Esta tecnología es ideal para aplicaciones IoT debido a

su capacidad para conectar dispositivos en áreas urbanas y rurales con una excelente cobertura y eficiencia energética [?]. Complementando este sistema, la red Helium proporciona una infraestructura de red global distribuida, diseñada para la conectividad de dispositivos IoT. Helium utiliza Hotspots LoRaWAN para crear una cobertura inalámbrica pública de largo alcance, incentivando a los usuarios a desplegar y mantener la infraestructura de red mediante la compensación en criptomonedas Helium [?]. Este trabajo se estructura en varios capítulos que abordan diferentes aspectos del desarrollo y la implementación del sistema de seguridad. Se inicia con una introducción al problema del robo de motocicletas y la necesidad de soluciones innovadoras. Luego, se profundiza en el marco teórico, explicando detalladamente las tecnologías utilizadas: RFID, LoRaWAN y otras relevantes. Además, se presentan los antecedentes y la metodología del proyecto, describiendo el diseño y las especificaciones del sistema, así como los resultados y análisis obtenidos durante la implementación. Finalmente, se concluye con una evaluación de los logros del proyecto y las recomendaciones para futuros trabajos en esta área.

1.1. Motivación

La motivación para la realización de este Trabajo Final de Grado no solo tiene un propósito práctico y social, sino también un componente académico y personal significativo. En primer lugar, el desarrollo de este sistema de seguridad antirrobo permite explorar y aplicar tecnologías emergentes en el ámbito del (IoT), como LoRaWAN y RFID, que están transformando la forma en que se gestionan y aseguran los recursos en diversas industrias. La implementación de estas tecnologías en un contexto real ofrece la oportunidad de expandir el conocimiento y habilidades técnicas en áreas clave de la ingeniería. Además, este proyecto sirve como un medio para alcanzar el objetivo académico de culminar la carrera de ingeniería, demostrando la capacidad para integrar conocimientos teóricos y prácticos adquiridos a lo largo de la formación universitaria. La realización de un proyecto de esta envergadura no solo contribuye al crecimiento profesional, sino que también fortalece la preparación para enfrentar desafíos futuros en el campo de la ingeniería. Finalmente, la motivación personal se ve impulsada por el deseo de ayudar a un amigo que enfrenta problemas de seguridad con su motocicleta. Este componente humanitario añade un valor adicional al proyecto, haciendo que el esfuerzo invertido tenga un impacto tangible y positivo en la vida de otros.

1.2. Definición del problema

Las motocicletas se han convertido en un componente esencial para la dinámica cotidiana, especialmente entre los trabajadores de entrega a domicilio. Sin embargo, la creciente incidencia de robos de motocicletas en Paraguay representa una amenaza para la seguridad y el sustento de los trabajadores que dependen de estas ágiles máquinas de dos ruedas para sus operaciones diarias. Según datos del Registro de Automotores, el número de motocicletas registradas ha experimentado un aumento constante en los últimos años, pasando de 855,797 en 2019 a 1,090,633 en 2023 [?]. Además, estadísticas de delitos registrados en comisarías revelan un incremento en los casos de robo de motocicletas. En 2019, se reportaron 3,270 casos de robo de motocicletas [?], mientras que en 2021, los números aumentaron a 3,551 casos [?]. Es importante destacar que, a pesar de estos alarmantes números, la tasa de esclarecimiento de estos delitos ha sido significativamente baja [?].

Estos datos representan un desafío significativo que requiere medidas para salvaguardar la vida y el sustento de quienes dependen de este medio de transporte.

Definición del problema de investigación

¿Cómo desarrollar un sistema de seguridad y rastreo de motocicletas, a través de la combinación de tecnologías *RFID*, *LoRaWAN*, como una alternativa a los sistemas de seguridad con rastreo disponibles?

1.3. Objetivos

1.3.1. Objetivo General

Desarrollar un prototipo de seguridad destinado a motocicletas utilizadas en servicios de entrega.

1.3.2. Objetivos Específico

1. Definir la estructura para el sistema de seguridad, considerando factores como integración entre tecnologías y cobertura.
2. Desarrollar un firmware para el microcontrolador que facilite la interacción entre los dispositivos de comunicación y control.
3. Desarrollar una interfaz para usuario.
4. Ensamblar un prototipo hardware incluyendo la integración de sensores, actuadores y sistema de alimentación.

5. Desarrollar un encapsulado 3D a medida para el prototipo.
6. Evaluar el rendimiento del sistema mediante pruebas en entornos controlados y en campo.

1.4. Hipótesis

La integración de tecnologías *RFID* con la red *LoRaWAN* representa una alternativa ventajosa a los sistemas de seguridad vehicular actuales para combatir el robo de motocicletas en entornos urbanos.

1.5. Justificación

En el mundo de hoy, donde la movilidad es esencial y las motocicletas desempeñan un papel fundamental en el transporte y la entrega de bienes, la seguridad de estos vehículos se ha vuelto más crítica que nunca [?, ?]. Para abordar eficazmente el diseño de soluciones en un contexto que abarca seguridad vehicular y tecnologías de IoT, es esencial sumergirse en el estado actual del conocimiento. Esta inmersión implica una exploración de la literatura existente con el fin de identificar las mejores prácticas y enfoques que puedan informar y enriquecer el desarrollo del sistema propuesto.

Este trabajo se basa en la necesidad de abordar la problemática de los robos de motocicletas en Paraguay, como lo demuestran los informes de la Agencia Nacional de Tránsito y Seguridad Vial (ANTSV) [?, ?]. La propuesta de investigación se centra en la implementación de identificadores *RFID* y su conexión a la red *Helium* mediante la tecnología *LoRaWAN*, dado que esta red ofrece un bajo costo y está en potencial crecimiento, aprovechando el alcance y las capacidades para proporcionar una alternativa efectiva para el rastreo y la vigilancia, como se evidencia en [?], con el objetivo de evaluar su viabilidad en el contexto paraguayo.

Este enfoque busca no solo generar conocimiento académico relevante, sino también tener un impacto positivo en la sociedad paraguaya. Se espera que la implementación de este sistema contribuya a mejorar la seguridad ciudadana y proteger los medios de subsistencia de trabajadores y empresas que dependen de motocicletas. Además, se anticipa que ayudará a reducir la tasa de robos y aumentar la tasa de recuperación, lo que resultaría en un entorno más seguro y confiable para la movilidad urbana y los servicios de entrega [?].

Desde una perspectiva personal, la motivación para llevar a cabo este proyecto surge de la convicción de abordar una problemática significativa

en la sociedad paraguaya. La oportunidad de aplicar conocimientos y habilidades para contribuir al bienestar social y mejorar los servicios de entrega representa una motivación intrínseca. En síntesis, este proyecto busca integrar la generación de conocimiento académico con la solución de problemas sociales, con el objetivo de ofrecer soluciones tecnológicas significativas para la comunidad.

1.6. Delimitación del alcance del trabajo

La investigación se centró en el desarrollo de un prototipo de seguridad para motocicletas, con un enfoque particular en la integración de tecnologías de identificación mediante RFID y la conectividad a la red LoRa a través de LoRaWAN. El estudio se llevó a cabo en un entorno controlado, lo que permitió una evaluación detallada del funcionamiento del prototipo bajo condiciones específicas.

El trabajo incluyó la implementación de un sistema que asegura el acceso y uso de las motocicletas únicamente por usuarios autorizados, además de la capacidad de monitorear la ubicación de la motocicleta, facilitando así la recuperación en caso de robo. Esta investigación estableció una base sólida para futuras exploraciones y desarrollos en el ámbito de la seguridad de motocicletas.

1.7. Descripción de los contenidos por capítulo.

A continuación, se muestran los distintos capítulos abarcados:

- En el capítulo 1 se plantea el inicio de la parte textual del informe del trabajo, como la motivación, la problemática, los objetivos trazados, la hipótesis, la justificación y el alcance del trabajo.
- En el capítulo 2 se presenta el marco teórico con los conceptos más importantes, frecuentemente mencionados y discutidos a lo largo del libro. También se hace mención a los trabajos similares que anteceden a la realización de este, que sirvieron como inspiración y soporte.
- En el capítulo 3 se describen las herramientas, y otros tipos de tecnologías utilizadas para el desarrollo de la aplicación.
- En el capítulo 4 se detalla la metodología que permitió alcanzar los resultados, incluyendo aspectos del funcionamiento de la aplicación, la metodología del desarrollo y las partes más importantes del código.

- En el capítulo 5 se expone el producto del análisis y pruebas llevadas a cabo, de forma a evaluar en resultado la aplicación.
- En el capítulo 6 se interpreta y sintetizan los resultados expuestos previamente, además de evaluar la validez de la hipótesis, el logro de los objetivos y realizar recomendaciones para futuros estudios.

Capítulo 2

Conceptos fundamentales, teoría y antecedentes

En este capítulo se desarrolla el marco teórico conceptual y referencial de este trabajo de investigación. En primer lugar, se presentan las definiciones básicas asociadas al estudio, como las relacionadas con la tecnología de geolocalización y radiofrecuencia. Por último, se presentan los antecedentes de trabajos anteriormente realizados en el área sobre el cual está basado este trabajo final de grado.

2.1. Internet de las cosas

Diversos autores han definido el Internet de las Cosas (*IoT*) de maneras que resaltan distintos aspectos de su funcionamiento e impacto. Un artículo publicado por Alejandro Cama y colaboradores describe el *IoT* como un sistema que utiliza (*redes de sensores inalámbricos (WSN)*) para integrar dispositivos de bajo costo en una red conectada a internet. Desde esta perspectiva, el *IoT* permite la comunicación y el control de dispositivos a través de internet, creando una red de objetos que pueden interactuar con su entorno y responder a cambios en tiempo real [?]. Por su parte, en el libro titulado *Internet de las Cosas*, los autores Rivera Berrio y Lopera Sánchez describen el *IoT* como una red de dispositivos cotidianos que abarca desde electrodomésticos hasta sistemas industriales, los cuales se conectan a internet para recopilar, analizar y compartir datos de manera continua. Para estos autores, el *IoT* facilita la interacción autónoma entre objetos, optimizando el control remoto y la automatización de procesos en múltiples contextos [?].

Kevin Ashton, quien acuñó el término “*Internet of Things*” en los años 90, planteó que si los objetos de la vida diaria estuvieran conectados a internet

y contaran con identificadores únicos, como etiquetas, podrían intercambiar información entre sí y ser gestionados eficazmente por computadoras, transformando la cadena de suministro. Ashton describió el *IoT* como una “red de cosas” que fusiona el mundo físico con el digital, permitiendo que la tecnología de sensores convierta los objetos físicos en una extensión del mundo de la información. En otro trabajo, Mora González define el *IoT* como un ecosistema digital que interconecta objetos inteligentes, permitiendo aplicaciones en sectores como la salud, la seguridad y las ciudades inteligentes. Según esta autora, el *IoT* tiene el potencial de transformar el entorno humano al conectar dispositivos que facilitan la toma de decisiones informadas y automatizadas en tiempo real, impulsando así el desarrollo de entornos inteligentes [?].

A partir de estas definiciones, el *IoT* puede sintetizarse como una red de objetos interconectados que, mediante sensores y dispositivos de comunicación, recopila, transmite y analiza datos en tiempo real. Este sistema facilita la comunicación y la automatización de procesos entre objetos físicos y digitales, permitiendo optimizar el control y la eficiencia en diversas áreas, desde el hogar hasta la industria. Aunque el *IoT* ofrece numerosas ventajas en términos de eficiencia, automatización y seguridad, también plantea desafíos significativos en cuanto a la privacidad y la protección de datos, factores esenciales para su implementación responsable y segura en la sociedad actual [?].

En este contexto, los objetos adquieren un papel activo en el mundo real, respondiendo a eventos específicos y desencadenando acciones. La aparición del *IoT* está impulsando una transformación que genera nuevos modelos de negocio, productos y compañías. Se espera que esta tecnología aporte numerosos beneficios, como la optimización de la cadena de abastecimiento, reducción de costos, mejoras en la experiencia del consumidor y beneficios en seguridad y servicios de emergencia [?].

Una representación gráfica del concepto de *IoT* se muestra en la Figura 2.1, donde se ilustra la interconexión de dispositivos y la transmisión de datos en una red de objetos inteligentes.

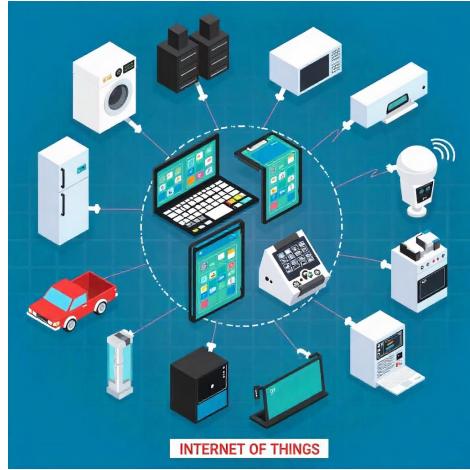


Figura 2.1: *Internet de las Cosas* (crédito: Imagen por macrovector en *Freepik*).

2.2. LoRA (*Long Range*)

LoRa es una tecnología de modulación de radiofrecuencia (*RF*) diseñada para redes de área amplia de baja potencia (*LPWAN*). Derivada de la técnica *Chirp Spread Spectrums (CSS)*, codifica información en ondas de radio mediante chirridos, similar a la comunicación de delfines y murciélagos. Esta técnica permite comunicaciones de largo alcance, hasta tres millas (cinco kilómetros) en áreas urbanas y hasta 10 millas (15 kilómetros) o más en áreas rurales con línea de visión. *LoRa* proporciona comunicaciones de largo alcance con requisitos de energía ultrabajos, lo que permite que los dispositivos funcionen con baterías durante hasta 10 años [?, ?, ?]. Según Bertoleti [?], esta tecnología es ideal para aplicaciones (*IoT*), dado que opera en bandas de frecuencia sin licencia y es resistente a interferencias.

Bandas de Frecuencia y Configuración Regional

LoRa opera en bandas de frecuencia sub-gigahertz específicas para cada región para evitar interferencias. En América, *LoRa* utiliza la banda de 902-928 MHz; en Europa, 863-870 MHz; y en China, 779-787 MHz. Los planes de frecuencia específicos y parámetros de cada región están detallados en los documentos [?, ?].

Ventajas y Limitaciones de LoRa

LoRa ofrece varias ventajas clave [?, ?]:

- Alcance Extenso y Eficiencia Energética: *LoRa* permite la comunicación de datos a grandes distancias con bajo consumo de energía.
- Inmunidad a Interferencias: Al operar en bandas de baja frecuencia y utilizar *CSS*, *LoRa* minimiza la interferencia de otros dispositivos.
- Costo-Efectividad: Comparado con otras tecnologías, *LoRa* reduce los costos de infraestructura y mantenimiento.

Sin embargo, *LoRa* tiene limitaciones en la tasa de transmisión de datos, con un máximo de alrededor de 37.5 kbps, lo que la hace inadecuada para aplicaciones que requieren transmisión de grandes volúmenes de datos, como video. Además, *LoRa* no incorpora encriptación de datos en su capa física, por lo que los desarrolladores deben implementar medidas de seguridad en capas superiores [?].

2.3. *LoRaWAN (Long Range Wide Area Network)*

LoRaWAN es el protocolo de red que permite organizar y gestionar redes de dispositivos que utilizan *LoRa* como capa física. Este protocolo, administrado por la *LoRa Alliance*, es de código abierto y permite una estructura de red en la cual miles de dispositivos pueden conectarse y enviar datos a un servidor de red a través de *gateways* (puntos de acceso). *LoRaWAN* facilita una comunicación bidireccional segura, escalable y de bajo consumo energético, adaptándose a los requerimientos de aplicaciones *IoT* a gran escala [?, ?]. La Figura 2.2 muestra la estructura de capas de *LoRa* y *LoRaWAN*, detallando las opciones *MAC*, *Media Access Control* y su integración con la capa física de *LoRa*.



Figura 2.2: Estructura de capas de *LoRa* y *LoRaWAN* [?].

LoRa se puede comparar con los cables físicos que interconectan los dispositivos en una red *Ethernet*, mientras que *LoRaWAN* representa el protocolo de comunicación que gestiona el intercambio de datos entre los dispositivos, similar a cómo una red *Ethernet* utiliza direcciones *MAC* e *IP* para coordinar la comunicación entre sus componentes.

Arquitectura de LoRaWAN

La arquitectura de *LoRaWAN* presenta una topología conocida como “Estrella de Estrellas” (*Star of Stars*), compuesta por cuatro elementos principales: dispositivos finales (nodos), *gateways* (puertas de enlace), servidor de red y servidor de aplicación. Los dispositivos finales, generalmente constituidos por sensores o actuadores, utilizan la capa física *LoRa* para compartir información adquirida con el *gateway*. El *gateway*, a su vez, recibe esta información y la transmite al servidor de red mediante una comunicación *Backhaul* (una red de retorno), que puede ser a través de redes *Wi-Fi*, *Ethernet* o redes móviles celulares [?].

El servidor de red se encarga de recibir los paquetes de datos enviados por los nodos, decodificarlos y aplicar protocolos de seguridad, permitiendo que cada aplicación reciba los datos necesarios del servidor de red. Este protocolo facilita la recopilación de datos en un solo *gateway* desde varios nodos ubicados a diferentes distancias, incluso kilómetros, a través de una comunicación unidireccional, aunque en algunos casos se utiliza la comunicación bidireccional entre nodo y *gateway*. Finalmente, los datos se envían al servidor de aplicaciones, encargado de procesar la carga útil. Esta arquitectura permite administrar miles de dispositivos en una sola red, optimizando la

escalabilidad y reduciendo los costos de infraestructura [?, ?]. En la Figura 2.3 se muestra la arquitectura de red previamente descrita.

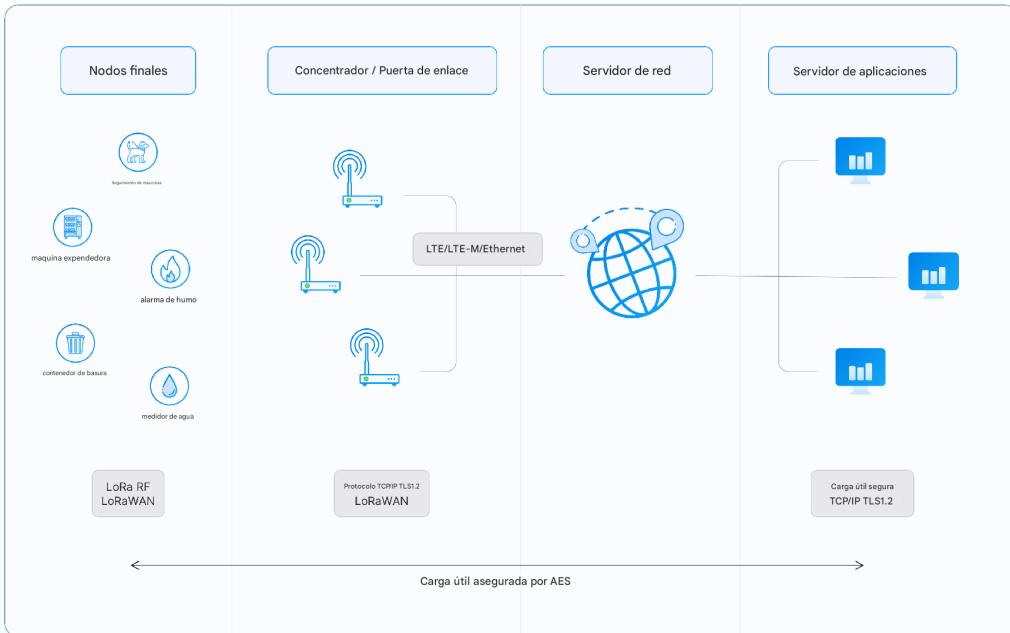


Figura 2.3: Una arquitectura típica de red *LoRaWAN* [?].

Clases de Dispositivos en LoRaWAN

LoRaWAN define tres clases de dispositivos, cada una optimizada para diferentes perfiles de uso y consumo energético [?, ?]:

- **Clase A:** Dispositivos que solo reciben datos después de transmitir, ideal para sensores de bajo consumo.
- **Clase B:** Dispositivos con ventanas de recepción adicionales sincronizadas mediante balizas.
- **Clase C:** Dispositivos en escucha continua, adecuado para dispositivos conectados a una fuente de energía constante.

La Figura 2.4 ilustra las características de cada clase de dispositivo en *LoRaWAN*, mostrando el momento de transmisión y las ventanas de recepción para cada clase. En la clase A, los dispositivos abren dos ventanas de recepción después de la transmisión, con retardo entre ellas, lo cual permite eficiencia energética. En la clase B, se añaden balizas de sincronización, lo

que permite ventanas de recepción programadas en momentos específicos. La clase C, por otro lado, está en escucha continua, lo que minimiza la latencia de recepción de datos, pero aumenta el consumo de energía. Esta figura ayuda a visualizar las diferencias en el comportamiento de las tres clases y su adecuación para distintos tipos de aplicaciones.

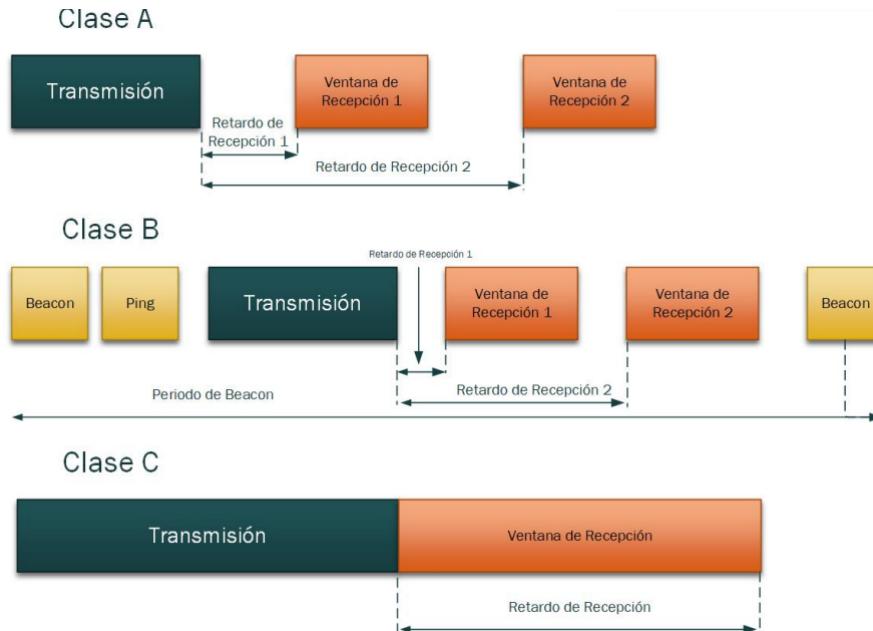


Figura 2.4: Clases de Dispositivos en *LoRaWAN* [?].

Seguridad en LoRaWAN

La seguridad es un componente esencial en *LoRaWAN*, diseñado para garantizar la autenticación, integridad y confidencialidad en las comunicaciones *IoT*. *LoRaWAN* implementa una doble capa de protección con tres claves principales:

- Autenticación en la Capa de Red:** La Network Session Key (*nwkskey-acronym*) autentica cada dispositivo, asegurando que solo nodos autorizados puedan acceder a la red. También valida la integridad de cada mensaje mediante un código de integridad (*MIC*) para prevenir alteraciones en los datos.
- Encriptación en la Capa de Aplicación:** La Application Session Key, (*Clave de Sesión de Aplicación (AppSKey)*) cifra y descifra los datos transmitidos, asegurando que solo el dispositivo y el servidor de aplicación puedan acceder al contenido, lo cual protege la privacidad del usuario [?, ?, ?].

3. **Clave de Aplicación (*AppKey*):** En el proceso de activación, los dispositivos que utilizan activación por aire (*OTAA*) generan las claves de sesión (*NwkSKey* y *AppSKey*) a partir de una clave de aplicación única (*AppKey*). Esto permite que las claves de sesión se regeneren en cada activación, añadiendo una capa adicional de seguridad dinámica [?, ?].

Para prevenir ataques de repetición, *LoRaWAN* emplea contadores de tramas (*frame counters*), evitando que los mensajes antiguos se retransmitan. Además, utiliza cifrado *Estándar de Cifrado Avanzado (AES)*, (*Advanced Encryption Standard*) en modo *CTR (Counter Mode)* para proteger los datos y *CMAC (Cipher-based Message Authentication Code)* para verificar la integridad, ambos métodos aprobados por el *NIST (National Institute of Standards and Technology)*.

La Figura 2.5 ilustra este flujo de seguridad en *LoRaWAN*. En ella, cada dispositivo obtiene las claves de sesión *NwkSKey* y *AppSKey* del *Join Server* durante el proceso de activación. La *NwkSKey* asegura la comunicación con el *LoRaWAN Network Server*, mientras que la *AppSKey* protege los datos entre el dispositivo y el *Application Server*. Los *gateways* actúan como intermediarios en este proceso, transmitiendo datos entre los dispositivos y el servidor de red, lo cual fortalece la seguridad de extremo a extremo en la red *LoRaWAN*.

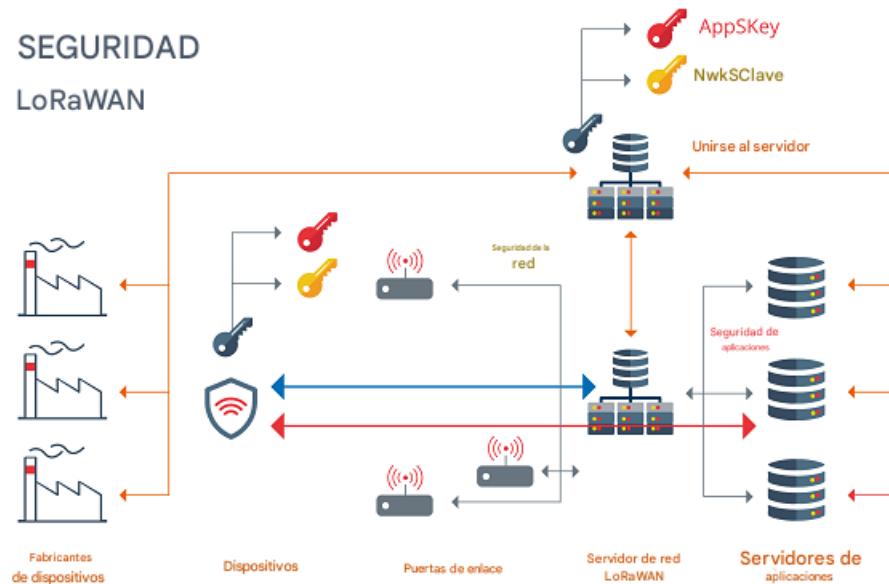


Figura 2.5: Diagrama de Seguridad en *LoRaWAN* [?].

Tasa de Datos Adaptativa (Adaptive Data Rate) en LoRaWAN

LoRaWAN incorpora el mecanismo de (Tasa de Datos Adaptativa (ADR)) que ajusta automáticamente el factor de dispersión, el ancho de banda y la potencia de transmisión según la distancia al *gateway* y la calidad de la señal. Esto permite que dispositivos cercanos al *gateway* usen menos potencia y una mayor tasa de datos, mientras que los dispositivos más alejados incrementan el factor de dispersión para mejorar la sensibilidad.

Parámetros Regionales y Configuración

El protocolo *LoRaWAN* también permite adaptarse a normativas regionales específicas, detalladas en [?]. Estas regulaciones incluyen los límites de potencia de transmisión y las bandas de frecuencia permitidas. [?] es una referencia para comprender las configuraciones regionales.

LoRa y *LoRaWAN* constituyen una combinación robusta para aplicaciones de *IoT* que requieren comunicación de largo alcance y bajo consumo. *LoRa* permite comunicación de largo alcance y eficiente en consumo, mientras que *LoRaWAN* estructura la red con seguridad, escalabilidad y gestión de dispositivos.

2.4. *Helium Networks*

Helium es una red global distribuida de puntos de acceso que crean cobertura inalámbrica pública de largo alcance para dispositivos *IoT* y celulares habilitados para *LoRaWAN*. Los *Hotspots LoRaWAN™* (puntos calientes) producen y se compensan en *IoT*, la criptomoneda nativa de la cadena de bloques *Helium*. La cadena de bloques *Helium* es una nueva cadena de bloques pública de código abierto creada enteramente para incentivar la creación de redes inalámbricas físicas y descentralizadas. Hoy en día, *Helium IoT Network* y sus cientos de miles de *Hotspots* brindan acceso a la red *LoRaWAN* más grande del mundo [?].

2.5. *RFID*

La identificación por radiofrecuencia, o *RFID*, es un término genérico para denominar a las tecnologías que utilizan ondas de radio. *RFID* es un método simple y automático de recolección de datos sobre un activo o producto (identificación, ubicación, estado, fecha y hora, etc.) de forma más rápida y fácil, sin requerir intervención humana y evitando errores.

Existen diversos métodos para realizar la identificación de objetos utilizando tecnología *RFID*, pero el más común consiste en almacenar un número de serie que identifica un producto (y posiblemente otro tipo de información adicional) en un microchip adjunto a una antena. El chip y la antena juntos conforman un *Tag RFID*. La antena permite al chip transmitir la información almacenada a un lector. Este lector convierte las ondas de radio de las etiquetas *RFID* en datos que pueden ser transmitidos a un sistema informático [?]. En la Figura 2.6, se ilustra la Interfaz Lector-Etiqueta y la Interfaz Lector-Sistema de un sistema *RFID*.

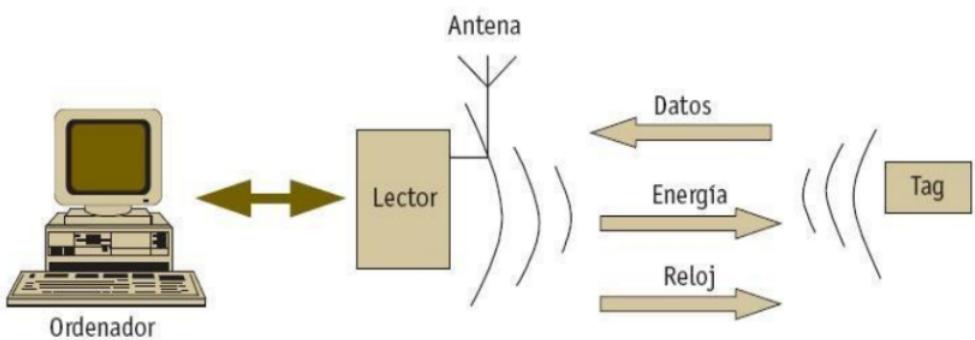


Figura 2.6: Modelo de Sistema *RFID* [?].

Componentes Principales de un Sistema *RFID*

Un sistema *RFID* consta de tres componentes esenciales:

1. Etiquetas *RFID* (*Tags*):

- Contienen un microchip y una antena que permiten la transmisión de datos.
- Clasificación:
 - *Activas*: Equipadas con batería interna, permiten comunicación a distancias mayores.
 - *Pasivas*: Sin fuente de energía interna, se activan por el campo electromagnético del lector.
- Almacenan información única que identifica el objeto al que están asociadas [?].

2. Lector *RFID* (*Reader*):

- Dispositivo que emite señales de radio para activar las etiquetas y leer los datos almacenados.
- Está compuesto por una antena y un procesador que interpreta la información transmitida por las etiquetas.

3. Middleware y Sistema de Gestión:

- Gestiona y procesa los datos recopilados por los lectores para integrarlos en sistemas de administración y monitoreo en tiempo real [?].

La Figura 2.7 muestra el flujo interno de comunicación entre un lector RFID y una etiqueta. Incluye los componentes claves.

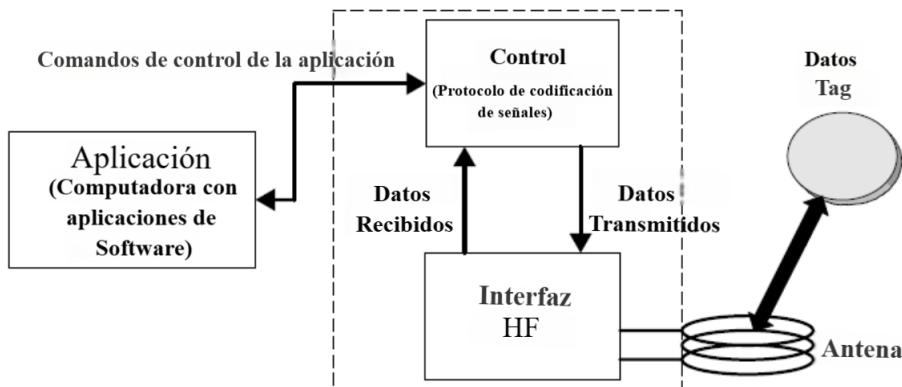


Figura 2.7: Modelo de comunicación entre el lector y los tags en un sistema *RFID* [?].

Funcionamiento de *RFID*

El lector genera un campo electromagnético que activa las etiquetas pasivas, permitiendo la transferencia de datos a través de la modulación de ondas de radio. Los datos recibidos por el lector son procesados y enviados a un sistema de gestión. En el caso de etiquetas activas, estas utilizan su batería para transmitir información de forma autónoma, logrando mayores alcances [?].

En algunos sistemas RFID, la comunicación entre los componentes se organiza siguiendo un modelo maestro-esclavo. En este modelo, los principales actores tienen roles claramente definidos:

- Aplicación (maestro): Envía comandos al lector para iniciar la comunicación y recibe los datos procesados desde las etiquetas.

- Lector: Actúa como intermediario entre la aplicación y las etiquetas. Recibe los comandos de la aplicación, los traduce a señales de radio para las etiquetas, y devuelve las respuestas de las etiquetas a la aplicación.
- Tags (esclavas): Responden a las señales del lector, enviando la información almacenada en su memoria interna.

La Figura 2.8 ilustra este flujo de datos, mostrando cómo las etiquetas y el lector interactúan bajo la dirección de la aplicación. Este modelo es común en aplicaciones que requieren un control jerárquico, como sistemas de control de acceso o gestión logística centralizada.

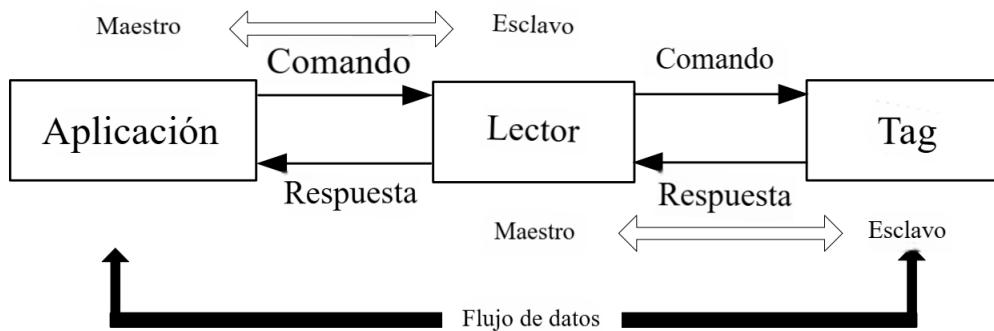


Figura 2.8: Flujo de datos en un sistema *RFID* basado en el modelo maestro-esclavo [?].

Sin embargo, no todos los sistemas *RFID* utilizan este modelo. Algunos sistemas optan por arquitecturas diferentes, dependiendo de sus necesidades y del tipo de etiquetas empleadas:

- Sistemas Autónomos (Peer-to-Peer): El lector opera de manera autónoma, identificando etiquetas dentro de su alcance sin necesidad de comandos desde una aplicación maestra. Este enfoque es común en sistemas de inventarios masivos.
- Tags Activas con Comunicación Directa: En redes *RFID* activas, las etiquetas con batería interna pueden comunicarse directamente con un servidor o receptor, eliminando la necesidad de un lector intermediario. Este modelo es ideal para rastreo de activos o vehículos en grandes áreas.

- Sistemas Distribuidos o en Red: Varios lectores pueden operar como nodos independientes en una red, reportando datos directamente a un servidor central sin jerarquías estrictas.

Cada modelo tiene sus ventajas y limitaciones. Mientras que el modelo maestro-esclavo es efectivo para aplicaciones que requieren control centralizado, los sistemas autónomos o distribuidos son más adecuados para entornos flexibles y escalables. La elección del modelo dependerá del caso de uso específico, el tipo de etiquetas y la infraestructura disponible.

Características Clave de *RFID*

- **Identificación sin contacto:** No requiere línea de vista directa, a diferencia de los códigos de barras.
- **Capacidad de lectura simultánea:** Puede identificar múltiples etiquetas en una sola operación.
- **Variedad de frecuencias:**
 - *LF (Low Frequency)*: 125-134 kHz, corto alcance (10 cm), ideal para control de acceso.
 - *HF (High Frequency)*: 13.56 MHz, alcance de hasta 1 metro, común en tarjetas inteligentes.
 - *UHF (Ultra High Frequency)*: 860-960 MHz, para logística e inventarios, con alcance de varios metros.
- **Durabilidad:** Las etiquetas pasivas tienen una vida útil prolongada, ya que no requieren baterías.
- **Seguridad:** Implementa medidas de autenticación y encriptación para proteger los datos transmitidos [?, ?].

Aplicaciones de *RFID*

1. Logística y Gestión de Inventarios:

- Seguimiento automatizado de productos y monitoreo en tiempo real en cadenas de suministro.

2. Control de Acceso:

- Uso en sistemas de seguridad para edificios y eventos.

3. Salud y Farmacéutica:

- Monitoreo de equipos médicos y rastreo de medicamentos para prevenir falsificaciones.

4. IoT y Ciudades Inteligentes:

- Integración en soluciones inteligentes para rastreo y gestión en tiempo real [?, ?].

RFID es una tecnología versátil y eficiente, especialmente en combinación con *IoT*. Su capacidad para identificar objetos de forma inalámbrica y sin contacto la convierte en una solución clave en sectores como logística, seguridad y salud. Al operar en diversas frecuencias y con opciones de configuración activa y pasiva, *RFID* se adapta a una amplia variedad de aplicaciones [?, ?].

2.6. Sistema de Geolocalización

La geolocalización es el reconocimiento de la posición de un dispositivo en el espacio real. El (*Sistema de Posicionamiento Global (GPS)*, por sus siglas en inglés) es la forma más conocida para obtener la localización geográfica, ubicando el dispositivo con una precisión de unos pocos metros. Los datos obtenidos de la posición del *GPS* no dependen exclusivamente de la conexión a la red de Internet, sino también de la geolocalización del dispositivo cuando no se encuentra fuera de la cobertura de los satélites geoestacionarios. Esta tecnología no es recomendable para uso en interiores como edificios, casas o lugares cerrados, ya que la cobertura de los satélites *GPS* es nula [?, ?].

2.7. Servicios de Entrega y Logística

La Gestión de la Entrega de Servicios, también conocida como gestión de la prestación de servicios, es un área clave en las empresas orientadas a los servicios. Consiste en un conjunto de procesos y prácticas destinados a garantizar la entrega eficaz y de alta calidad de servicios a los clientes. El Gerente de Entrega de Servicios es el responsable encargado de supervisar y coordinar estas actividades. El papel principal de la Gestión de la Entrega de Servicios es asegurarse de que los servicios proporcionados cumplan con las expectativas de los clientes en cuanto a calidad, plazos y rendimiento. Esto implica una gestión proactiva de proyectos, recursos y relaciones con los clientes [?].

2.8. Antecedentes

2.8.1. Lokalisering av sensorer med LoRaWAN på Kalmar Länssjukhus (Año 2021)

Desarrollado por: Nathalie Wetterskog, Jeanette Marie Victoria Skepeland Hole.

En [?] se evaluó un sistema de posicionamiento basado en sensores conectados a *LoRaWAN* para localizar sillas de ruedas en el hospital de Kalmar. Este sistema ayudaría al personal del hospital a encontrar sillas de ruedas perdidas, mejorando la eficiencia operativa. La metodología empleada consistía en la instalación de sensores en las sillas de ruedas, que estaban conectados al sistema de rastreo *Traxmate*, una plataforma *IoT* diseñada para rastrear dispositivos mediante la recolección de datos de ubicación y la visualización en tiempo real [?].

Los principales parámetros evaluados fueron: la precisión de la ubicación geográfica, la fuerza de la señal recibida (*RSSI*) y la latencia de transmisión. La precisión de la ubicación se determinó mediante trilateración usando las direcciones *MAC* de los puntos de acceso *Wi-Fi* cercanos, lo que subraya la metodología empleada en [?]. La latencia de transmisión se calculó como el tiempo de retraso entre la señal emitida por el sensor y su recepción por el sistema.

Los resultados mostraron que *LoRaWAN* es adecuado para su uso en el entorno hospitalario debido a su capacidad para operar tanto en interiores como en exteriores, aunque se encontraron deficiencias en la exactitud del sistema de posicionamiento [?].

Es importante destacar que ambos trabajos utilizan la tecnología de *LoRaWAN* para la geolocalización, pero presentan diferencias significativas en su enfoque: este utiliza trilateración con puntos de acceso *Wi-Fi* para determinar la ubicación, mientras que la propuesta actual se apoya en una combinación de *RFID* y *Sistema Global de Navegación por Satélite (GNSS)* para mejorar la seguridad y la validación del propietario [?]. Estas diferencias subrayan la versatilidad de las tecnologías de geolocalización en distintos contextos.

2.8.2. LoRaWAN for tracking inland routes of plastic waste: Introducing the smart TRACKPLAST bottle (Año 2023)

Desarrollado por: Stavros Ponis, George Plakas, Eleni Aretoulaki, Dimitra Tzanetou, Theodoros Nikolaos Maroutas.

En [?], se desarrolló un sistema para rastrear el flujo de residuos plásticos desde tierra hasta el mar, utilizando la tecnología *LoRaWAN*. A través del desarrollo de una botella inteligente *TRACKPLAST*, se buscó emular y seguir el comportamiento de los residuos plásticos en el medio ambiente.

El diseño y la fabricación de la botella inteligente implicó la integración de sensores *IoT* en una botella de plástico comercial. Los sensores fueron integrados de manera que no afectaran la apariencia o el comportamiento de la botella en el ambiente. Para la comunicación de las botellas inteligentes, se desarrolló una red *LoRa* específica. Esta red incluyó la instalación de tres *gateways* en la región de *Attiki*, cerca de Atenas, que eran utilizados para la trilateración y el seguimiento preciso de las botellas [?]. Se realizaron pruebas de conectividad para asegurar el funcionamiento correcto de la red.

El proyecto destacó el uso de servicios de geolocalización de la plataforma *LoRa Cloud Geolocation*, una *API* que trabaja con un servidor de red, que recibe metadatos como *RSSI*, *SNR* (*Signal-to-Noise Ratio*) y *TOA* (*Time on Air*) de los *gateways* para calcular las posiciones de los dispositivos. La geolocalización se logró específicamente mediante la técnica *TDOA*, *Time Difference of Arrival*. Además, se creó una plataforma en la nube para procesar y visualizar los datos recolectados por los sensores. La plataforma recibía las señales a través de los *gateways*, analizaba los datos geoespaciales y los mostraba en un mapa, permitiendo a las autoridades locales rastrear las rutas de las botellas. Esta plataforma fue probada por expertos y usuarios finales para garantizar su funcionalidad y facilidad de uso [?].

La validación del sistema se realizó en condiciones reales mediante una implementación piloto en la isla de *Syros*, Grecia. Se desplegaron 69 botellas inteligentes y se siguieron sus movimientos en tiempo real desde abril hasta junio de 2022. Esta fase permitió obtener datos valiosos sobre el comportamiento y destino de los residuos plásticos.

Los resultados del proyecto fueron positivos. Las botellas *TRACKPLAST* se rastrearon con éxito durante toda su vida útil de batería, proporcionando información crucial para la gestión de residuos plásticos. Los datos obtenidos ayudaron a las autoridades locales a comprender mejor los flujos de residuos plásticos y a implementar medidas preventivas para reducir la contaminación marina [?].

Aunque ambos trabajos utilizan la tecnología de *LoRaWAN*, se enfocan en aplicaciones diferentes: el sistema de *TRACKPLAST* se centra en el rastreo y análisis del comportamiento de los residuos plásticos, mientras que el presente trabajo busca fortalecer la seguridad del propietario a través de una integración de *RFID* y *GNSS*. Esto demuestra la adaptabilidad de la tecnología de geolocalización para abordar problemáticas variadas y resaltar su potencial en distintos ámbitos.

2.8.3. Smart Security System for Two-Wheelers (Año 2020)

Desarrollado por: Niyati Rana, Pravesh Khatta, Antim Dev Mishra.

En [?], se desarrolló un sistema de seguridad para motocicletas y *scooters* utilizando la tecnología *IoT*. Este sistema incluye la creación de un prototipo que integra *hardware* y *software* para ofrecer protección contra robos. El sistema se construyó alrededor de la plataforma *Arduino Mega* y emplea varios componentes clave: un sensor de huellas dactilares, un módulo *GPS Neo-6M*, un módulo *GSM* y una alarma sonora.

El sensor de huellas dactilares se utiliza para autenticar al usuario antes de permitir el arranque del vehículo. Solo se puede arrancar el vehículo si la huella coincide con la registrada en el sistema. El módulo *GSM* se encarga de la comunicación entre el vehículo y el propietario, enviando mensajes *SMS* en caso de detección de actividad sospechosa. El módulo *GSM* está conectado a la red móvil y vinculado al sistema de *Arduino*. Si el sistema detecta un intento de manipulación no autorizado, el módulo *GSM* envía un mensaje de alerta al teléfono del propietario, quien recibe la notificación indicando el posible intento de robo.

El sistema permite al propietario controlar ciertas funciones de forma remota mediante comandos *SMS*. El propietario puede enviar un comando *SMS* específico al módulo *GSM*, que lo transmite a la placa *Arduino*, y esta ejecuta los comandos. Además, el sistema activa la alarma sonora para disuadir a los ladrones. Si el sensor de huellas detecta un intento de arranque no autorizado, se activa una alarma sonora, alertando a las personas cercanas y posiblemente ahuyentando al ladrón. Adicionalmente, se acciona el módulo *GPS*, lo cual proporciona la ubicación del vehículo en tiempo real. Esto permite al propietario y a las autoridades localizar y recuperar el vehículo de manera más eficiente [?].

El proyecto logró desarrollar un prototipo funcional que proporciona una solución de seguridad eficaz y económica para vehículos de dos ruedas. Se logró un monitoreo en tiempo real, alertas inmediatas de intentos de ro-

bo, control remoto de funciones y disuasión mediante una alarma sonora, demostrando cómo la integración de tecnologías *IoT* puede mejorar significativamente la seguridad de motocicletas y *scooters* [?].

Es importante resaltar que, aunque este sistema de seguridad se centra en la protección de motocicletas y utiliza tecnologías como huellas dactilares y *GSM*, la propuesta actual se distingue al incorporar diferentes tecnologías, como *RFID* y geolocalización mediante *LoRaWAN*. Estas tecnologías adicionales buscan ofrecer un enfoque complementario en la validación de la identidad del usuario y el rastreo de la ubicación del vehículo, destacando la diversidad de soluciones tecnológicas aplicables en el ámbito de la seguridad vehicular.

2.8.4. Smart Security System for Vehicles using Internet of Things (IoT) (Año 2018)

Desarrollado por: Mithileysh Sathiyanarayanan, Santosh Mahendra, Rajesh Babu Vasu.

En [?], se describe el desarrollo de un sistema de seguridad inteligente para vehículos utilizando la tecnología *IoT*. El objetivo principal es transformar los sistemas de seguridad vehicular convencionales (*CVSS*) en sistemas de seguridad vehicular inteligentes (*SVSS*) que permitan el acceso y control remoto de los vehículos mediante un teléfono inteligente.

El sistema se construyó alrededor de varios componentes clave. Primero, un subsistema de identificación que emplea etiquetas *RFID* y un lector conectado a un microcontrolador *Arduino Uno R3*. Cuando se intenta encender el vehículo, el sistema debe validar la etiqueta para permitir el arranque del motor. Otro subsistema de detección utiliza un receptor *GPS* para obtener la ubicación en tiempo real del vehículo. Estos datos se envían al microcontrolador y se pueden ver en la aplicación móvil del propietario.

Para la salida y alerta, el sistema cuenta con indicadores *LED*, un *buzzer* y una pantalla *LCD*. Si se detecta un intento de robo, el *buzzer* emite una alarma sonora y los relés controlan el encendido del motor, bloqueando el arranque si no se autentica correctamente. Por último, el subsistema de control es manejado por el microcontrolador, que ejecuta comandos y procesa datos. Un módulo *GSM* conectado envía y recibe mensajes *SMS*. Ante actividades sospechosas, el módulo envía alertas al propietario, quien puede bloquear el motor, activar la alarma o solicitar la ubicación del vehículo mediante comandos *SMS*.

Además, cuenta con una aplicación móvil que permite al propietario ver la ubicación del vehículo, bloquear o desbloquear el motor y activar la alar-

ma. También recibe alertas en tiempo real sobre intentos de robo y otras actividades de seguridad [?].

Las pruebas del sistema se realizaron en varias etapas para asegurar su funcionalidad y efectividad. Estas pruebas demostraron que el sistema puede identificar al propietario de manera precisa mediante *RFID*, proporcionar la ubicación exacta del vehículo en tiempo real y enviar alertas instantáneas en caso de intento de robo. Además, la capacidad de control remoto a través de la aplicación móvil y comandos *SMS* añadió una capa adicional de seguridad y conveniencia.

El proyecto logró desarrollar un sistema de seguridad vehicular inteligente que utiliza tecnologías *IoT* para mejorar la protección y el control de vehículos. Este sistema ofrece una solución efectiva y económica para prevenir robos, con la ventaja adicional de proporcionar control remoto y monitoreo en tiempo real a través de teléfonos inteligentes [?].

Este enfoque de seguridad vehicular, que combina tecnologías de *IoT*, presenta características específicas que mejoran la protección de vehículos. Por otro lado, el trabajo actual utiliza tecnologías distintas, como una combinación de *RFID* y *GNSS*, lo que permite un enfoque diferente en la validación y el rastreo, mostrando así la diversidad de aplicaciones en la seguridad de vehículos y la versatilidad de las tecnologías disponibles.

2.8.5. Development of an IoT-Based (LoRaWAN) Tractor Tracking System (2022)

Desarrollado por: Çağdaş Civelek.

En [?], se describe el desarrollo de un sistema *IoT* de rastreo para tractores utilizando *LoRaWAN*, con el objetivo de apoyar la agricultura de precisión en Turquía. Este proyecto surge ante la dificultad de implementar tecnologías avanzadas de monitoreo en áreas rurales, donde la infraestructura y los altos costos de soluciones comerciales basadas en *LTE* limitan su adopción, especialmente para agricultores de pequeñas explotaciones. La solución busca proporcionar una herramienta asequible y de bajo consumo energético que permita monitorear en tiempo real la ubicación y el rendimiento de los tractores, optimizando así las operaciones agrícolas y el uso de recursos.

El sistema está construido en torno a un microcontrolador *PIC18F46K22*, al cual se le acopla un módulo *LoRa RN2483* para la transmisión de datos en la banda de frecuencia *ISM, Industrial, Scientific, and Medical*. Esto permite la transmisión de datos a largas distancias sin necesidad de infraestructura de comunicación convencional. Además, el sistema integra un módulo *GPS SE868K7-A*, que proporciona coordenadas precisas de la ubicación del trac-

tor, y diversos sensores, incluidos un medidor de flujo de combustible y un sensor de torque en la toma de fuerza (*PTO*), para obtener información detallada sobre el consumo de combustible y la eficiencia operativa del vehículo. La transmisión de los datos recolectados se realiza mediante un *gateway LoRaWAN Kerlink Wirnet*, que envía la información a un servidor en la nube. Allí, los datos se almacenan en una base de datos *MySQL* y se muestran en tiempo real a través de una interfaz web, accesible desde dispositivos móviles o computadoras, permitiendo a los usuarios visualizar la posición del tractor y otras métricas de rendimiento relevantes para la gestión agrícola.

Durante las pruebas de campo realizadas en áreas rurales cercanas a la Universidad de Çukurova, el sistema mostró un rendimiento satisfactorio, logrando transmitir datos de forma confiable hasta 15 km en zonas con baja interferencia. La precisión del *GPS*, con un margen de error de aproximadamente 3 metros, resultó adecuada para el monitoreo en entornos de cultivo, proporcionando una visión precisa de la ubicación y movimientos del tractor en el campo. Además, se destacó que el costo de producción del sistema ronda los \$55, significativamente menor que los \$795 que suelen costar las soluciones comerciales basadas en *LTE*. Este bajo costo convierte al sistema en una alternativa viable y accesible para los agricultores que desean implementar tecnologías de monitoreo sin incurrir en elevados gastos.

Este trabajo demuestra cómo *LoRaWAN* puede adaptarse a las necesidades de la agricultura de precisión, permitiendo el rastreo y la gestión de maquinaria agrícola de manera rentable. Comparado con el enfoque de seguridad vehicular del presente trabajo, que emplea *LoRaWAN* en combinación con *RFID* y *GNSS* para la autenticación y monitoreo, este sistema agrícola se enfoca en mejorar la eficiencia y sostenibilidad de la producción agrícola. Sin embargo, ambos estudios destacan la flexibilidad y efectividad de *LoRaWAN* en aplicaciones de *IoT* para distintos contextos, mostrando su potencial para transformar tanto la agricultura como otros sectores que requieren soluciones de monitoreo remoto.

2.8.6. Use of RFID Technology as a Reporting Mechanism in Vehicle Tracking System (2016)

Desarrollado por: Mamudu Hamidu.

En [?], se describe un sistema de seguridad vehicular denominado *Transport Highway Integrated Security System (THISS)*, diseñado para facilitar el rastreo en tiempo real y la notificación de incidentes en autopistas mediante tecnología *RFID*, *GPS* y *GSM*. La motivación para este desarrollo surge de la necesidad de contar con una herramienta rápida y precisa que

permite a los conductores reportar eventos críticos, como robos, accidentes o fallos mecánicos, y mejorar la seguridad en las carreteras, especialmente en situaciones donde la asistencia inmediata puede ser crucial.

El sistema utiliza un módulo *RFID* para activar la señal de emergencia de manera sencilla y rápida, junto con un *GPS* que obtiene la ubicación exacta del vehículo. Una vez que el conductor activa el sistema mediante el escaneo de una tarjeta *RFID*, el *GPS* comienza a recopilar las coordenadas, y el módulo *GSM* envía esta información a una base de datos central en tiempo real. Este proceso permite registrar y monitorear la ubicación del vehículo y clasificar el tipo de emergencia según el incidente reportado. Para su implementación, el sistema cuenta con una estructura de comunicación en la cual los datos de ubicación y emergencia se gestionan a través de la red *GSM/GPRS*, garantizando una cobertura amplia y una transmisión de datos confiable.

Los resultados obtenidos en las pruebas iniciales del sistema revelaron una alta efectividad en la notificación de incidentes y en el envío de la ubicación en tiempo real. Las alertas generadas contenían coordenadas precisas y se transmitieron de forma consistente, asegurando que los datos llegaran a la base de datos central sin demoras significativas. Este flujo continuo de información permite a las autoridades responder con mayor rapidez y precisión a los eventos reportados, reduciendo el tiempo de reacción en caso de emergencias. La infraestructura de comunicación mediante *GSM* también facilitó la recolección de datos de ubicación, consolidándolos en una base de datos para su análisis y monitoreo a largo plazo.

Este proyecto demuestra cómo el uso de *RFID*, *GPS* y *GSM* en conjunto puede crear una solución de seguridad vehicular robusta y accesible, que ofrece una alternativa práctica para la gestión de emergencias en autopistas. A diferencia del presente trabajo, que emplea *LoRaWAN* y *RFID* en combinación con *GNSS* para la autenticación y rastreo vehicular, el sistema *THIS* se enfoca en la notificación rápida de eventos y la asistencia en carretera. Ambos sistemas destacan el valor de integrar diferentes tecnologías de comunicación y localización para mejorar la seguridad y la capacidad de respuesta en distintos escenarios de uso.

Capítulo 3

Tecnologías utilizadas

En este capítulo se presentan las herramientas utilizadas para el desarrollo de este trabajo. En primer lugar, para asegurar que las herramientas seleccionadas sean adecuadas, se realiza una comparación entre alternativas considerando los factores y características más importantes dentro de las actividades.

3.1. Componentes de *Hardware*

Los componentes de hardware abarcan los dispositivos físicos necesarios para la ejecución del sistema, incluyendo microcontroladores, sensores y módulos de comunicación, que se encargan de realizar las tareas físicas y electrónicas requeridas.

3.1.1. Placa de Desarrollo WiFi LoRa 32 (V3)

La *WiFi LoRa 32 (V3)* (véase Figura 3.1) es una placa de desarrollo *IoT* basada en el microcontrolador *ESP32-S3FN8*, un procesador de doble núcleo de 32 bits que opera a una frecuencia de hasta 240 MHz. Esta placa está diseñada para aplicaciones de bajo consumo y alto rendimiento en redes inalámbricas, tales como ciudades inteligentes, sistemas de seguridad, agricultura conectada y control industrial. Una de las características principales es la integración de múltiples tecnologías de comunicación, incluyendo *Wi-Fi*, *Bluetooth Low Energy (BLE)* y *LoRa*, lo que le permite ser utilizada en una variedad de entornos [?].

La placa cuenta con un transceptor de radio *SX1262*, que permite la transmisión de datos en redes de largo alcance. Soporta modulaciones *LoRa* y *FHSS* para aplicaciones de redes de baja potencia y larga distancia

(LPWAN), además de la modulación (G)FSK para aplicaciones más tradicionales. Este transceptor cubre un rango de frecuencias de 150 MHz a 960 MHz, permitiendo el uso en bandas *ISM* globales. El SX1262 está diseñado para cumplir con los requisitos de la capa física de la especificación *LoRaWAN* y se ajusta a las regulaciones internacionales de transmisión de radiofrecuencia, lo que lo hace adecuado para su implementación en diversas regiones del mundo [?].



Figura 3.1: Placa de desarrollo Heltec

La selección de la placa *WiFi LoRa 32 (V3)* para esta investigación se basa en un análisis de diversas placas de desarrollo y herramientas de hardware, con el objetivo de identificar las tecnologías más adecuadas para aplicaciones de (*IoT*). Se revisaron microcontroladores y transceptores *LoRa*, considerando su funcionalidad y rendimiento.

Según [?], los microcontroladores son componentes centrales en la transición de los sistemas embebidos hacia el *IoT*. Estos dispositivos permiten manejar múltiples procesos de forma autónoma, lo que es esencial para la conectividad y automatización de dispositivos.

Las placas de desarrollo deben cumplir ciertos requisitos fundamentales, como conectividad y compatibilidad con diversas plataformas. En este contexto, se comparan las características clave de tres placas populares: el *Wio E5 Mini*, el *ESP32 Heltec LoRa WiFi V3* y el *NÚCLEO-WL55JC*. Estas placas son reconocidas por su versatilidad y eficiencia en monitoreo y comunicación, como se documenta en diversos estudios [?, ?].

Con el fin de seleccionar la placa de desarrollo, se establecieron una serie de criterios clave de evaluación que permiten analizar y comparar las opciones disponibles.

- **Conectividad:** Se refiere a la capacidad del dispositivo para soportar diferentes tipos de comunicación y protocolos.

- **Frecuencia LoRa:** Considera la compatibilidad del dispositivo con la frecuencia operativa del protocolo *LoRa* para esta región.
- **Características adicionales:** Incluye funcionalidades extra que pueden mejorar el rendimiento o la versatilidad del dispositivo, como sensores o interfaces de comunicación.
- **Compatibilidad con consolas:** Evalúa qué tan bien la placa puede integrarse con las plataformas de administración y control de dispositivos *LoRaWAN* que permiten la gestión y supervisión del dispositivo.
- **Tamaño/Dimensiones:** Este criterio evalúa la adecuación del tamaño del dispositivo ya que se requiere un diseño compacto.
- **Entornos de desarrollo (Facilidad):** Considera la facilidad de uso del entorno de desarrollo, lo que puede afectar la curva de aprendizaje y el tiempo de implementación.

Para reflejar el desempeño relativo de cada dispositivo, se empleó una escala del 1 al 3:

- **3:** Indica el mejor rendimiento o la opción más favorable en ese criterio.
- **2:** Representa un rendimiento intermedio, adecuado pero no el mejor.
- **1:** Refleja un rendimiento limitado en comparación con los otros dispositivos.

A continuación, se presenta la tabla 3.1 comparativa que resume las características técnicas más relevantes de cada dispositivo, facilitando así la selección de la opción más adecuada para este proyecto.

Tabla 3.1: Tabla comparativa entre Placas de Desarrollo

Criterio/Placa	Wio E5 Mini
Conectividad	2
Frecuencia LoRa	2
Características adicionales	1
Compatibilidad con consolas	3
Tamaño/Dimensiones	3
Entornos de desarrollo (Facilidad)	2
Total	13

De la tabla comparativa se determina que el *WiFi LoRa V3* es el dispositivo más adecuado para la investigación. Este dispositivo fue seleccionado para las pruebas debido a su equilibrio entre conectividad y compatibilidad con diversos entornos de desarrollo, lo que lo convierte en una opción ideal para la fase de prototipado.

3.1.2. Lector RFID MFRC522

El *MFRC522* (ver Figura 3.2) es un circuito integrado de alto rendimiento diseñado para la lectura y escritura de tarjetas de identificación por radiofrecuencia (*RFID*) a 13.56 MHz, compatible con los estándares ISO/IEC 14443 A y tecnologías *MIFARE* y *NTAG*. Este módulo es ideal para aplicaciones de control de acceso, sistemas de pago sin contacto, y sistemas de seguimiento por identificación de objetos, brindando un diseño compacto y eficiente.

El *MFRC522* está equipado con un transmisor integrado que permite la comunicación con tarjetas *RFID* sin necesidad de circuitos adicionales, proporcionando una distancia de operación de hasta 50 mm (dependiendo del diseño y tamaño de la antena) [?].



Figura 3.2: Lector *RFID*

En el ámbito de los sistemas de control de acceso basados en *RFID*, diversos trabajos han explorado el uso de estas tecnologías por su facilidad de implementación y capacidad para integrarse en proyectos de *IoT* y seguridad. Por ejemplo, investigaciones previas [?, ?] resaltan la viabilidad y flexibilidad de los módulos *RFID* en, aplicaciones prácticas.

Para la selección del módulo *RFID* más adecuado para el presente proyecto, se consideraron criterios clave como la disponibilidad en el mercado,

el costo, la simplicidad de implementación y la compatibilidad con plataformas de desarrollo comunes. A continuación, se presenta la tabla comparativa 3.2, que incluye tres módulos *RFID*: *PN532 NFC RFID Module V3*, *RC522 RFID Module* y *R200 UHF RFID Reader*.

Tabla 3.2: Tabla comparativa entre módulos *RFID*

Criterio / Módulos	PN532
Disponibilidad en el mercado	
Costo	
Simplicidad de implementación	
Compatibilidad con plataformas populares	
Adecuación para pruebas y prototipado	
Total	

De acuerdo con los resultados obtenidos en la tabla comparativa, el *RC522 RFID Module* ha sido seleccionado como la opción más adecuada para el proyecto en curso. Su alta disponibilidad, bajo costo y facilidad de implementación lo convierten en el candidato ideal para llevar a cabo pruebas y prototipos. Además, su compatibilidad con plataformas populares como *Arduino* y su sencilla integración en sistemas de control de acceso lo destacan frente a los otros módulos evaluados.

En contraste, aunque el *PN532* ofrece características avanzadas como la compatibilidad *NFC*, su implementación es más compleja. El *R200 UHF*, aunque destaca por su capacidad de lectura a larga distancia, es menos apropiado para un proyecto de bajo costo y prototipado rápido debido a su precio más elevado y la necesidad de antenas externas.

3.1.3. Módulo GNSS GP-02 Kit

El *GP-02-Kit* (ver Figura 3.3) es un módulo de desarrollo altamente integrado que incorpora un receptor de navegación por satélite multimodo de alto rendimiento. El módulo utiliza el chip de posicionamiento por satélite *AT6558R*, el cual combina un frontal de radiofrecuencia, un procesador de banda base digital, una CPU *RISC* de 32 bits, gestión de energía y funciones de detección y protección de antenas activas.

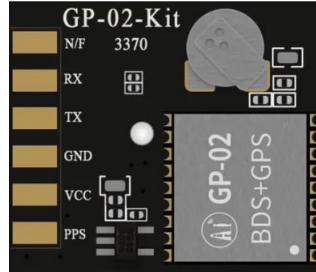


Figura 3.3: Módulo de desarrollo *GP-02 Kit*

Este módulo es compatible con múltiples sistemas de navegación por satélite, incluyendo el sistema de navegación por satélite *BeiDou* de China, el sistema *GPS* de Estados Unidos y el *GLONASS* de Rusia, lo que le permite realizar un posicionamiento conjunto multi-sistema, garantizando alta precisión en aplicaciones críticas de geolocalización.

El *GP-02-Kit* sigue el protocolo *NMEA* para la transmisión de datos y se comunica con otros dispositivos a través de una interfaz serial *UART*, la cual soporta tasas de transferencia de hasta 256000 bps. Además, cuenta con una antena cerámica integrada, lo que simplifica su uso en entornos móviles o aplicaciones de seguimiento donde el tamaño y la eficiencia son esenciales [?].

3.1.4. Placa de Desarrollo ESP-WROOM-32

El *ESP32-WROOM-32* (ver Figura 3.4) es un módulo *MCU* de alto rendimiento que incluye *Wi-Fi*, *BT* y *BLE*, ideal para una variedad de aplicaciones, desde redes de sensores de bajo consumo hasta tareas avanzadas como codificación de voz, transmisión de música y decodificación de *MP3*.

En el núcleo de este módulo se encuentra el chip *ESP32-D0WDQ6**, diseñado para ofrecer escalabilidad y flexibilidad. Posee dos núcleos de *CPU*, controlables individualmente, con una frecuencia de reloj ajustable entre 80 MHz y 240 MHz. Además, permite apagar la *CPU* principal y usar el coprocesador de bajo consumo para monitorear los periféricos de forma constante. Entre sus múltiples periféricos, el *ESP32* integra sensores táctiles capacitivos, sensores *Hall*, interfaz para tarjeta *SD*, *Ethernet*, *SPI* de alta velocidad, *UART*, *I2S* e *I2C* [?].

Este microcontrolador fue elegido debido a su flexibilidad y disponibilidad, así como por sus capacidades de manejo de periféricos y transmisión de datos, lo que lo convierte en una opción adecuada para las demandas de comunicación en este sistema.



Figura 3.4: Placa de desarrollo *ESP32*

3.2. Componentes de *Software*

Los componentes de software proporcionan las herramientas y entornos de programación necesarios para diseñar, compilar y cargar el código en el hardware. Además, facilitan la visualización de los datos recolectados, permitiendo una supervisión eficiente y en tiempo real del sistema. No solo posibilitan la integración con el hardware, sino que también proporciona una plataforma para interpretar y presentar la información

3.2.1. *Helium Console*

Helium Console es una plataforma desarrollada por Helium que permite la gestión de organizaciones y dispositivos en redes LoRaWAN a través de una interfaz basada en navegador web. Actúa como un servidor de red LoRaWAN (LNS, por sus siglas en inglés, LoRaWAN Network Server) compatible con la blockchain de Helium, conocido como Router, que facilita la comunicación entre dispositivos LoRaWAN y la red de Helium. Helium Console simplifica la integración y operación de dispositivos, proporcionando a los usuarios una forma eficiente y directa de administrar sus dispositivos IoT (Internet of Things, o Internet de las Cosas) dentro de la infraestructura de la red Helium [?].

Una de las principales ventajas de Helium Console es su enfoque “plug-and-play” (lista para usar), lo que significa que los usuarios pueden configurar y operar su propia Organización Única de Identificación (OUI, Organizationally Unique Identifier) en la red Helium de manera rápida y sin necesidad de permisos especiales. Dado que la red de Helium es descentralizada y abierta, la operación de una OUI es completamente permissionless (sin necesidad de permisos), lo que facilita la incorporación de nuevos dispositivos IoT sin las barreras típicas que se encuentran la mayoría de redes.

Se evaluaron varias plataformas de servidor de red LoRaWAN (LNS) con el objetivo de elegir la más adecuada para el trabajo. El proceso de selección se basó en analizar criterios específicos que afectan directamente la funcionalidad y adaptabilidad de cada plataforma a los requisitos del proyecto.

Criterios considerados

- Costo: Considera el gasto involucrado en utilizar la plataforma, desde suscripciones hasta la infraestructura adicional que pueda requerirse. Cuanto menor es el costo, mejor es la evaluación.
- Facilidad de Uso: Mide la simplicidad y accesibilidad de la interfaz y su curva de aprendizaje, ideal para usuarios con diversos niveles de experiencia.
- Escalabilidad: Evalúa la capacidad de la plataforma para adaptarse y crecer, tanto en usuarios como en dispositivos, sin que el rendimiento disminuya.
- Cobertura: Analiza la disponibilidad geográfica de la plataforma y su cobertura en distintas áreas.
- Control y Personalización: Valora el nivel de control que ofrece la plataforma para adaptar la infraestructura y configuraciones específicas.
- Privacidad y Seguridad: Mide las medidas de seguridad y cifrado en la transmisión de datos. Mayor puntaje si la plataforma ofrece seguridad avanzada y privacidad de datos.
- Integración: Evalúa la facilidad de integración de la plataforma con otros sistemas y su capacidad para conectarse con tecnologías IoT.
- Documentación: Analiza la calidad y accesibilidad de la documentación proporcionada, indispensable para configurar la plataforma.

En la tabla 3.3 se comparan plataformas de servidor de red LoRaWAN (LNS) como: Helium Console, The Things Network (TTN), ChirpStack, Loriot para determinar la plataforma más adecuada.

Tabla 3.3: Comparativa de plataformas LoRaWAN

Criterios	Helium Console
Costo	Bueno
Facilidad de Uso	Excelente
Escalabilidad	Excelente
Cobertura	Bueno
Control y Personalización	Regular
Privacidad y Seguridad	Bueno
Integraciones	Excelente
Documentación	Excelente

Helium Console destaca principalmente como la opción ganadora debido a su cobertura, un aspecto crucial en el análisis comparativo. Aunque otras plataformas ofrecen excelentes características en varias áreas, Helium sobresale por su capacidad de brindar una red descentralizada y global. Además, otras características como su facilidad de uso, integraciones y documentación excelente también refuerzan su posición.

3.2.2. Thingsboard

ThingsBoard es una plataforma IoT de código abierto que facilita la recopilación, procesamiento, visualización y gestión de dispositivos de datos. La plataforma destaca por su capacidad para manejar grandes volúmenes de datos y dispositivos, lo que la convierte en una opción robusta y flexible para diversas aplicaciones en el ámbito de la Internet de las Cosas. Su diseño modular permite a los desarrolladores integrar y gestionar dispositivos de forma eficiente, sin necesidad de construir una infraestructura desde cero.

La arquitectura de ThingsBoard es clave para su funcionalidad, y se puede implementar en dos modalidades: monolítica y microservicios. En su modo monolítico, todos los componentes se ejecutan dentro de una única máquina virtual Java (JVM), optimizando el uso de recursos del sistema operativo y simplificando su implementación en entornos con limitaciones de memoria y procesamiento. Este enfoque centralizado facilita el desarrollo y el prototipado de soluciones IoT, al requerir menos configuración y memoria. Por otro lado, la modalidad basada en microservicios distribuye los componentes del sistema en nodos independientes, lo que permite una escalabilidad horizontal y una mayor tolerancia a fallos, ideal para proyectos de gran escala donde es

crucial gestionar grandes cantidades de dispositivos y datos de manera eficiente. La figura 3.5 ilustra cómo ThingsBoard organiza sus componentes y cómo interactúan para manejar la comunicación entre dispositivos y sistemas externos [?].

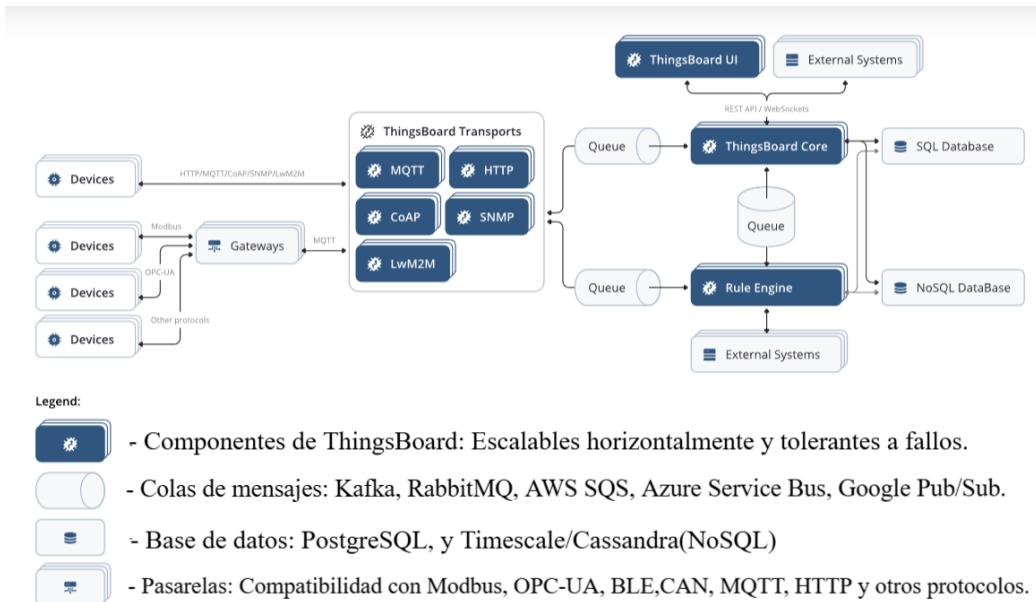


Figura 3.5: Diagrama de Arquitectura ThingsBoard

Uno de los componentes esenciales de ThingsBoard es su motor de reglas (Rule Engine), que juega un papel central en la automatización y procesamiento de eventos dentro de la plataforma. Este motor permite procesar los datos entrantes desde los dispositivos IoT y ejecutar acciones automáticas basadas en la lógica definida por el usuario. El sistema procesa los datos a través de una serie de nodos que realizan diferentes operaciones, como almacenamiento, generación de alertas o control remoto de dispositivos mediante llamadas a procedimientos remotos RPC. Esto posibilita la creación de flujos de trabajo complejos que reaccionan en tiempo real a los eventos generados por los dispositivos, permitiendo a la plataforma tomar decisiones y ejecutar acciones automáticamente sin intervención humana. En la Figura 3.6 , donde se puede observar cómo se configuran las cadenas de reglas para tomar decisiones basadas en los mensajes recibidos, por ejemplo, enviando comandos de RPC o almacenando datos en bases de datos [?].

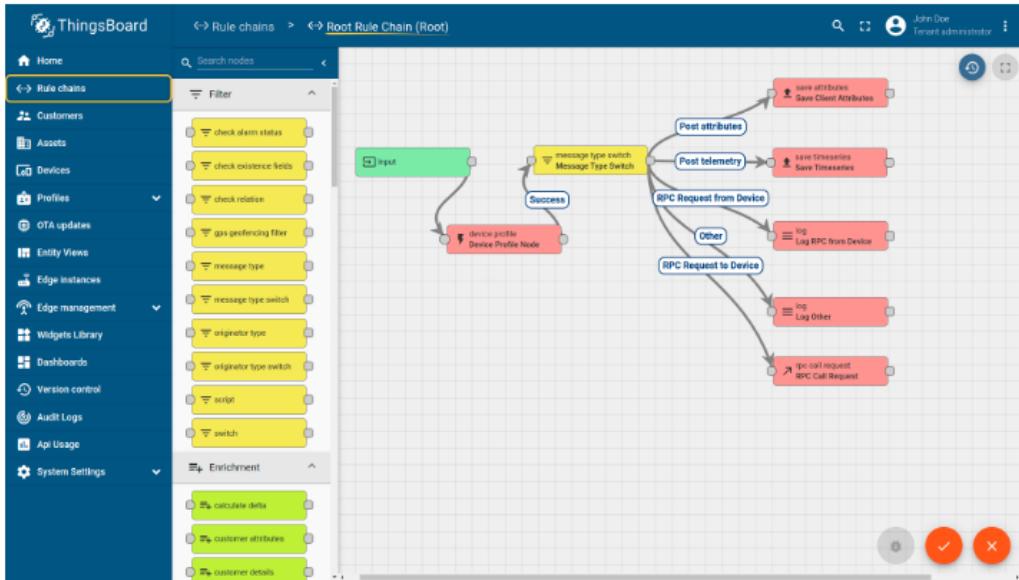


Figura 3.6: Cadena de reglas raíz ThingsBoard

Una visualización adecuada de los datos adquiridos es esencial ya facilita el procesamiento de los mismos, lo que contribuye a la precisión y efectividad de las decisiones tomadas a partir de esta información. Con el objetivo de facilitar esta visualización y gestión de datos, se seleccionó una plataforma IoT, basándose en estudios que resaltan el uso de estas herramientas en contextos de monitoreo y análisis de datos [?, ?, ?, ?, ?, ?, ?].

Para determinar la plataforma más adecuada, se evaluaron Ubidots, ThingSpeak y ThingsBoard bajo criterios técnicos relevantes para entornos de prototipado, incluyendo:

- Escalabilidad: Este criterio evaluó la capacidad de cada plataforma para manejar un aumento en el número de dispositivos y datos a medida que el proyecto crece.
- Protocolos Soportados: Se analizó la diversidad de protocolos de comunicación IoT que cada plataforma puede gestionar.
- Visualización de Datos: Este criterio evaluó la calidad de las herramientas de visualización de datos disponibles en cada plataforma.
- Automatización de Reglas: Se evaluó la capacidad de cada plataforma para configurar reglas y flujos de trabajo automáticos.

- Costo: Este criterio consideró los costos asociados al uso de las plataformas.
- Soporte y Comunidad: La disponibilidad de soporte técnico y la existencia de una comunidad activa fueron evaluadas.
- Instalación Local (On-Premise): Este criterio evaluó la opción de implementar la plataforma en un servidor propio.
- Facilidad de Uso: Finalmente, se evaluó la intuitividad y facilidad de uso de la interfaz para usuarios tanto novatos como avanzados.

Cada plataforma fue calificada en una escala de 1 a 3, permitiendo una comparación cuantitativa de sus capacidades, donde:

- 1 indica un rendimiento bajo.
- 2 un rendimiento medio.
- 3 un rendimiento alto.

La tabla 3.4 resume los resultados obtenidos, y constituye la base para seleccionar la plataforma que mejor responde a los requerimientos de visualización y control de datos en el proyecto.

Tabla 3.4: Tabla comparativa de Plataformas IoT

	Criterio/Plataforma	Ubidots	ThingSpeak
	Escalabilidad	2	2
	Protocolos Soportados	2	2
	Visualización de Datos	3	2
	Automatización de Reglas	2	2
	Costo	3	3
	Soporte y Comunidad	3	3
	Instalación Local (On-Premise)	1	1
	Facilidad de Uso	3	3
	Total	17	16

De acuerdo con los criterios evaluados, ThingsBoard se destaca como la plataforma IoT más adecuada para el presente proyecto. Su versión Community Edition permite una instalación local sin costos asociados, lo cual

resulta especialmente ventajoso en un entorno de prototipado. Además, su capacidad para escalar y su soporte para una amplia gama de protocolos lo convierten en una opción robusta para gestionar y visualizar información en tiempo real. Una de las razones clave para elegir ThingsBoard fue su integración nativa con MQTT [?], un protocolo ligero que utiliza un modelo de publicación/suscripción para optimizar la transmisión de datos entre dispositivos IoT. Esto permite una comunicación eficiente, en tiempo real y con bajo consumo de recursos, asegurando la escalabilidad del sistema. Estas características, junto con la posibilidad de uso gratuito y sin limitaciones de licenciamiento comercial, posicionan a ThingsBoard como la solución óptima, permitiendo una implementación sostenible y adaptable.

3.2.3. Arduino IDE

El Arduino IDE o Entorno de Desarrollo Integrado de Arduino, es un entorno de desarrollo integrado que facilita la programación y carga de código en microcontroladores mediante una interfaz amigable [?]. Permite a los usuarios escribir y editar “sketches” (programas o códigos) en un lenguaje basado en C/C++, simplificado para el control de hardware, lo que agiliza el desarrollo de proyectos de electrónica.

Una de sus ventajas es la flexibilidad en la configuración de hardware, gracias a su “Gestor de Placas” que adapta el código a distintos modelos de microcontroladores y permite la instalación de librerías adicionales. En este proyecto, el Arduino IDE es especialmente útil debido a su compatibilidad con la placa Heltec LoRa WiFi 32 V3, que cuenta con librerías y referencias específicas de Heltec, lo que simplifica la integración de *LoRaWAN* y WiFi.

Además, el IDE permite la carga del código de manera directa desde una conexión USB o inalámbrica, haciendo más ágil el ciclo de pruebas y depuración, gracias a sus herramientas básicas para detectar errores en el código. Estas características hacen del Arduino IDE una opción versátil y eficiente para proyectos con microcontroladores, adecuada para usuarios de todos los niveles.

La interfaz del Arduino IDE se muestra en la Figura 3.7, donde es posible observar sus principales herramientas para la edición y carga de código.

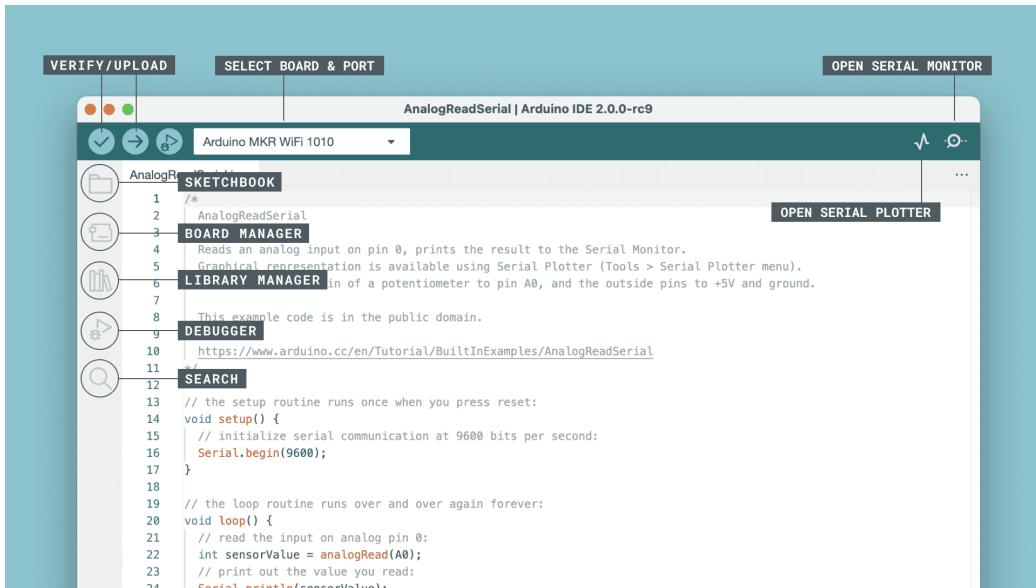


Figura 3.7: Interfaz del Arduino IDE 2 [?].

3.2.4. Fritzing

Fritzing es un software de automatización del diseño electrónico (Electronic Design Automation, EDA) orientado a diseñadores, artistas, investigadores y cualquier persona interesada en la electrónica y el desarrollo de prototipos. Su propósito principal es ofrecer herramientas que faciliten la documentación y el intercambio de proyectos, permitiendo a los usuarios crear y diseñar circuitos de forma visual y accesible [?].

El programa se destaca por su interfaz intuitiva, que permite a los usuarios diseñar circuitos, crear esquemas de circuitos impresos y compartir sus prototipos. Además, Fritzing es compatible con otras herramientas de diseño como Processing y Arduino, formando un ecosistema en el que los usuarios pueden documentar y compartir sus proyectos con facilidad. Esto lo convierte en una herramienta valiosa tanto para la enseñanza de electrónica como para la creación de prototipos destinados a la fabricación [?].

La Figura 3.8 muestra la interfaz de Fritzing, en la cual es posible visualizar y editar circuitos de manera sencilla y gráfica. Este entorno facilita tanto el diseño de circuitos como la creación de esquemas listos para fabricación, adaptándose a las necesidades de diversos usuarios y proyectos.

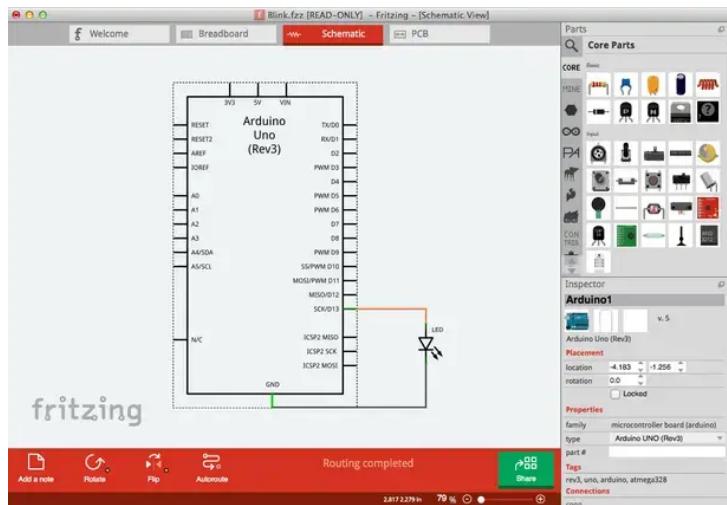


Figura 3.8: Entorno de diseño de Fritzing mostrando el esquema de una placa [?].

3.2.5. Blender

Blender es una suite de creación 3D gratuita y de código abierto que soporta toda la cadena de producción 3D: modelado, rigging, animación, simulación, renderizado, composición, seguimiento de movimiento, edición de video e incluso creación de juegos. Los usuarios avanzados pueden utilizar la API de Blender para scripting en Python, permitiendo personalizar la aplicación y desarrollar herramientas especializadas que a menudo se incluyen en futuras versiones.

Una de las grandes ventajas de Blender es su accesibilidad y versatilidad. Al ser gratuito y estar disponible para múltiples plataformas como Linux, Windows y macOS, permite que tanto individuos como pequeños estudios accedan a herramientas profesionales sin incurrir en altos costos. Su interfaz unificada y el uso de OpenGL garantizan una experiencia consistente en todos los sistemas operativos [?].

Además, Blender es impulsado por una comunidad activa bajo la Licencia Pública General de GNU (GPL), lo que significa que cualquier persona puede contribuir al desarrollo del software. Esto conduce a la incorporación constante de nuevas funciones, correcciones rápidas de errores y mejoras en la usabilidad. La colaboración comunitaria fomenta la innovación y asegura que el software evolucione para satisfacer las necesidades cambiantes de sus usuarios [?].

Blender no solo ofrece una solución completa para la creación de contenido 3D, sino que también promueve un entorno colaborativo y abierto. Su

combinación de potencia, flexibilidad y gratuidad lo convierte en una opción ideal para aquellos que buscan desarrollar proyectos creativos sin las limitaciones de los software propietarios.

En la figura 3.9, se muestra la interfaz de Blender, donde es posible dibujar directamente en una ventana gráfica 3D. Esta funcionalidad abre una libertad de flujo de trabajo sin igual para creadores de guiones gráficos y artistas 2D. Blender permite combinar elementos 2D y 3D directamente en la ventana gráfica. Además, incluye capas y colores para trazos y rellenos, así como la capacidad de esculpir pinceladas y asociarlas a objetos 3D.



Figura 3.9: Entorno de desarrollo en Blender.

Capítulo 4

Método

Este capítulo describe el desarrollo del trabajo de investigación, enfocado en la creación de un prototipo de control para motocicletas mediante tecnologías *RFID* y *LoRaWAN*, con el propósito de implementar un sistema de autenticación y localización para prevenir robos. El trabajo constituye una investigación aplicada al desarrollo tecnológico de un sistema de seguridad antirrobo, con un enfoque cuantitativo y un diseño experimental y transversal en el tiempo, abarcando el desarrollo de un prototipo funcional. En las siguientes secciones se detallan los módulos desarrollados, cada uno representando una fase clave en el funcionamiento del sistema (ver Figura 4.1).



Figura 4.1: Fases principales del sistema.

4.1. Definición de arquitectura

En esta primera fase, se definieron los requisitos técnicos orientados a proporcionar seguridad y monitoreo remoto en tiempo real mediante tecnologías *IoT*. El objetivo principal fue desarrollar un sistema que autentique al propietario de la motocicleta a través de un módulo *RFID* y, en caso de intento de robo o alejamiento del vehículo sin el *tag* autorizado, interrumpa el encendido y envíe una alerta a un servidor remoto. Para satisfacer estos requisitos, se diseñó una arquitectura que integra comunicación de baja potencia me-

diente *LoRaWAN* y un dispositivo de geolocalización *GNSS*, permitiendo el monitoreo y la visualización de datos en una plataforma gráfica accesible.

4.1.1. Requisitos clave

- Autenticación del propietario mediante *RFID*.
- Corte de corriente en el sistema de encendido de la motocicleta cuando el *tag* autorizado se encuentre fuera de rango.
- Envío de datos de geolocalización (coordenadas *GNSS*) y estado de autenticación mediante *LoRaWAN* hacia la red Helium.
- Monitoreo remoto de la ubicación y estado de la motocicleta a través de una interfaz gráfica.

Con base en estos requisitos, se definieron dos aspectos principales: la estructura interna del prototipo y la arquitectura de comunicación externa hacia el servidor.

4.1.2. Estructura interna del prototipo

La Figura 4.2 ilustra el diseño del hardware y el flujo de control dentro del prototipo.

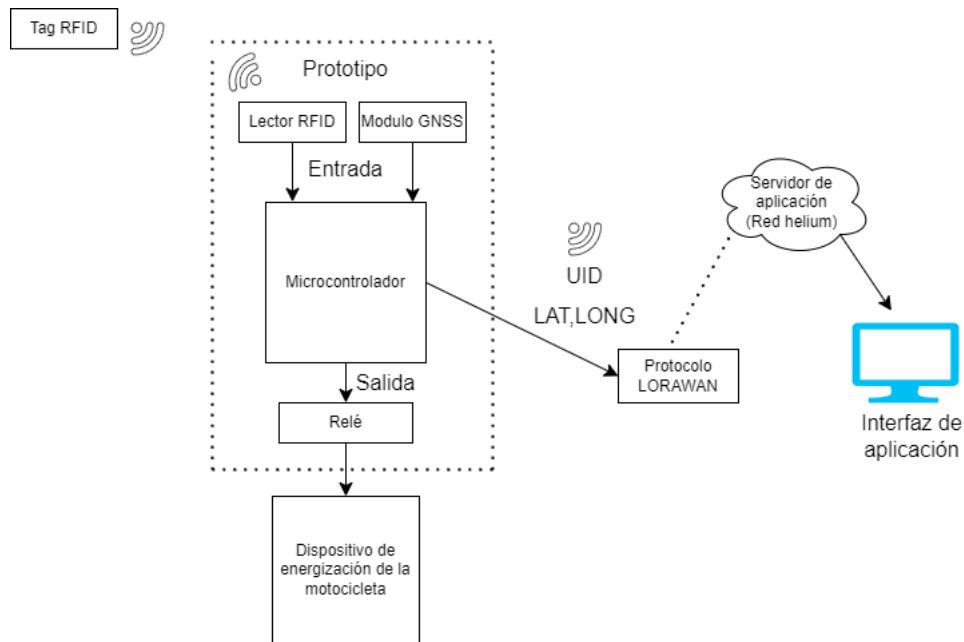


Figura 4.2: Estructura del prototipo.

- **Microcontrolador (Heltec WiFi LoRa 32 V3):** Procesa las entradas de los módulos *RFID* y *GNSS*, y envía datos mediante *LoRaWAN*.
- **Módulo *RFID*:** Detecta y autentica el *tag* del usuario. Si el *UID* del *tag* coincide con uno de los usuarios autorizados, se habilita el sistema de arranque.
- **Módulo *GNSS*:** Proporciona las coordenadas de geolocalización, que se envían junto con el *UID* autenticado para monitoreo remoto.
- **Relé:** Controla el flujo de energía hacia el encendido de la motocicleta. Si el *tag* *RFID* no es reconocido o se retira del rango, el relé corta la energía, evitando que la motocicleta arranque.

4.1.3. Arquitectura de comunicación externa

La Figura 4.3 muestra cómo los datos capturados por el prototipo se procesan para cumplir con los requisitos de monitoreo remoto en tiempo real.

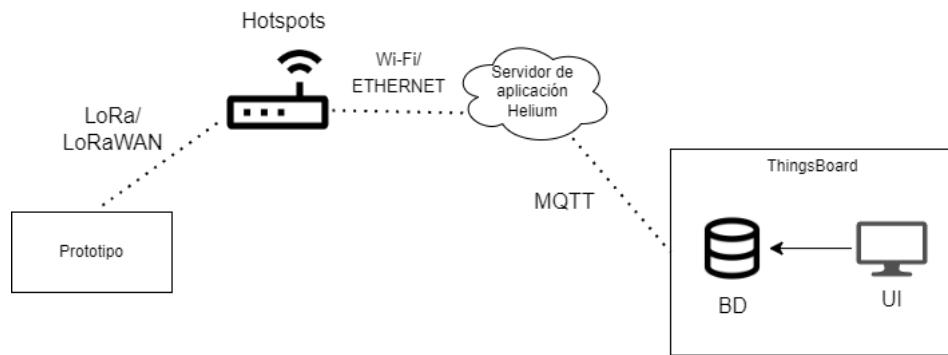


Figura 4.3: Arquitectura de comunicación y monitoreo.

- **Prototipo:** Recopila datos del *RFID* y las coordenadas *GNSS*. El microcontrolador envía esta información utilizando el protocolo *LoRaWAN*.
- **Hotspots Helium:** Actúan como puntos de acceso, recibiendo los datos enviados por el prototipo y transmitiéndolos al servidor de aplicación.
- **Servidor Helium:** Procesa los datos recibidos y los envía mediante el protocolo *MQTT* a la plataforma *ThingsBoard*.

- **ThingsBoard (UI):** Plataforma que permite la visualización gráfica de los datos en tiempo real. Los usuarios pueden monitorear la ubicación de la motocicleta, su estado de autenticación, y recibir alertas en caso de eventos inusuales, como intentos de robo.

4.2. Módulo de autenticación y control

En esta fase, se desarrollaron las funciones principales para garantizar la autenticación del usuario antes de permitir el arranque de la motocicleta. El sistema está diseñado para validar la presencia de un *tag RFID* autorizado; solo cuando se detecta un *tag* registrado se permite el arranque del motor de la motocicleta. Si el *tag* no es detectado o es removido en algún momento, el sistema corta el suministro de corriente al motor, bloqueando su funcionamiento. Este mecanismo proporciona una primera capa de seguridad para prevenir el uso no autorizado de la motocicleta.

4.2.1. Requisitos del sistema

Para implementar esta funcionalidad, se definieron los siguientes requisitos:

- **Comparación de UID:** El sistema debe comparar el UID del *tag* detectado con una lista de UID autorizados.
- **Verificación en rango:** Una vez autorizado el *tag*, el sistema realiza una lectura constante para verificar que el *tag* permanezca en rango.

Considerando estos requisitos, se diseñó el diagrama de flujo que representa el funcionamiento del sistema de autenticación y control (Figura 4.4).

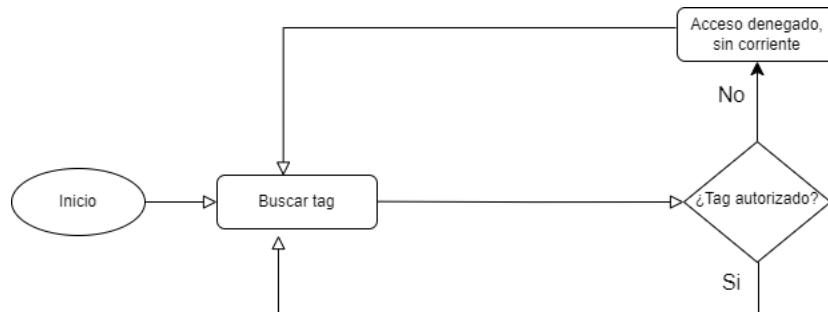


Figura 4.4: Diagrama de flujo de autenticación y control.

4.2.2. Conexión del módulo *RFID*

Para la comunicación entre el módulo *RFID RC522* y la placa de desarrollo *WiFi LoRa 32 V3* se empleó el protocolo SPI. Los pines de conexión utilizados fueron:

- **CS:** Pin 34.
- **MOSI:** Pin 35.
- **CLK:** Pin 36.
- **MISO:** Pin 26.
- **RESET:** Pin 33.
- **VCC y GND:** Pines de alimentación.

En la Figura 4.5 se muestra el diagrama de conexiones utilizadas para esta configuración.

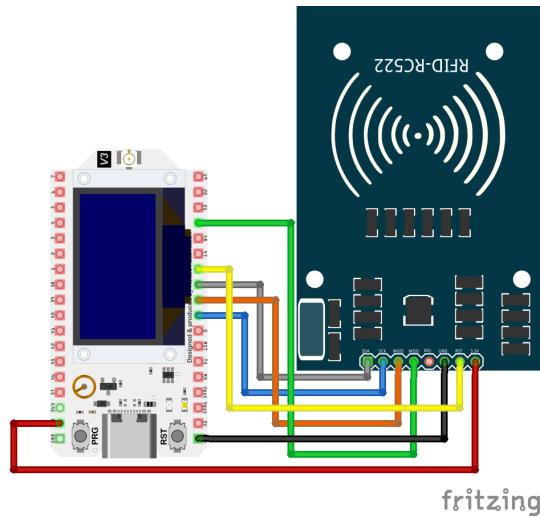


Figura 4.5: Diagrama de conexiones entre WiFi LoRa 32 V3 y RFID RC522.

4.2.3. Primera versión del código: implementación básica en el bucle principal

La primera versión del código implementa un sistema de autenticación y monitoreo continuo de un *tag RFID*, permitiendo el acceso únicamente cuando se detecta la presencia de un *tag* autorizado. En este caso, se definieron

dos UIDs correspondientes a *Usuario1* y *Usuario2*. Al leer el UID del *tag*, el sistema lo compara con estos valores autorizados para conceder o denegar el acceso.

Autenticación del *tag*

- Si **tagAutenticada** es **falso**, el sistema permanece en espera, buscando la presencia de un *tag RFID*.
- Cuando se detecta un *tag*, se lee su UID y se utiliza la función **comparaUID()** para verificar si coincide con uno de los usuarios autorizados. Si coincide, se muestra un mensaje de bienvenida y **tagAutenticada** se establece en **verdadero**.
- Si el UID no es reconocido, se muestra un mensaje indicando que el *tag* no es válido.

Verificación de presencia del *tag*

- Si **tagAutenticada** es **verdadero**, el sistema verifica constantemente si el *tag* autorizado permanece en rango.
- Si el tiempo desde la última detección del *tag* excede un límite definido (**tiempoCorte**), el sistema corta el suministro de corriente al motor.
- Si el *tag* sigue en rango, se actualiza el tiempo de última lectura y el motor continúa en funcionamiento.

4.2.4. Segunda versión del código: integración de FreeRTOS y modularización

En esta versión, se incorporó FreeRTOS para gestionar de manera concurrente la autenticación del *tag RFID* y la verificación de su presencia. Además, el código se modularizó en archivos específicos para facilitar su mantenimiento y escalabilidad:

- **Archivo rfidfunciones.h:** Declaraciones de variables y funciones relacionadas con el módulo *RFID*.
- **Archivo rfidfunciones.cpp:** Implementación de funciones y tareas de FreeRTOS para autenticación y monitoreo.
- **Archivo principal RFIDFreeRTOS.ino:** Inicialización de tareas de FreeRTOS y configuración de parámetros principales.

Tareas implementadas con FreeRTOS

Tarea 1: tareaDeteccionTag

- Busca la presencia de un *tag RFID*. Si se detecta, verifica su autenticidad mediante la función `autenticarTag()`.
- Monitorea la permanencia del *tag* en rango. Si el *tag* es detectado nuevamente, actualiza el tiempo de última lectura.
- Notifica a la tarea de verificación del tiempo mediante un semáforo (`tagDetectadaSemaphore`).

Tarea 2: tareaVerificarTiempo

- Espera la señal del semáforo para ejecutar la función `verificarTiempo()`, que comprueba si el tiempo transcurrido desde la última detección supera el límite definido (`tiempoCorte`).
- Si el tiempo excede el límite, se corta el suministro de corriente y se establece `tagAutenticada` en falso.

Esta versión proporciona una estructura modular, mejorando la eficiencia y permitiendo un control más preciso del sistema. El uso de *FreeRTOS* optimiza la gestión de tareas críticas, asegurando la sincronización y escalabilidad del sistema.

4.3. Módulo de captura y transmisión de datos

Para llevar a cabo la captura y transmisión de datos en tiempo real, se desarrollaron códigos específicos que permiten la adquisición de coordenadas de ubicación mediante el módulo *GNSS* y su transmisión eficiente a través de *LoRaWAN*, integrando ambos subsistemas en el funcionamiento general del prototipo.

Este proceso se dividió en tres fases:

- Implementación básica para obtención de datos de las coordenadas en tiempo real.
- Implementación y configuración del módulo *LoRaWAN* para la transmisión de datos.

- Integración de la comunicación entre los módulos *GNSS* y *LoRaWAN*.

Para la comunicación entre el módulo *GNSS* y la placa de desarrollo *WiFi LoRa 32 V3*, se empleó el protocolo *UART*. Los pines de conexión utilizados fueron el pin 45 (RX), pin 46 (TX), y los pines de alimentación *VCC* y *GND*. En la Figura 4.6, se muestra el diagrama de conexiones utilizado.

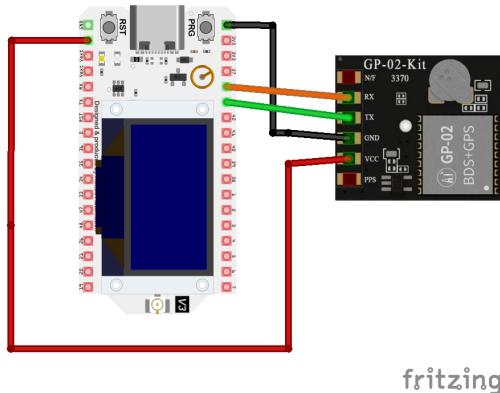


Figura 4.6: Diagrama de conexiones entre *WiFi LoRa 32 V3* y *GNSS*.

4.3.1. Implementación básica para obtención de datos de las coordenadas en tiempo real

Una vez conectado el módulo *GNSS* a la placa de desarrollo, se implementó un código para obtener y mostrar en tiempo real los datos de ubicación (latitud y longitud), junto con la fecha y hora actual. Este código se basa en la librería *TinyGPSPlus*, que permite interpretar los datos en formato *NMEA* recibidos desde el GPS. A continuación, se describe la implementación:

Función `setup()`

Durante la fase de inicialización, la función `setup()` se encarga de:

- Configurar el monitor serial (`Serial`) a 115200 baudios para facilitar la visualización de los datos en una terminal serial.
- Configurar `Serial1`, el puerto serial dedicado a la comunicación con el GPS, con los pines y velocidad previamente definidos.

Proceso de captura de datos en la función `loop()`

En la función `loop()`, el sistema verifica continuamente si hay datos disponibles en el puerto `Serial1`, donde el módulo *GPS* envía sus mensajes de datos. El flujo de trabajo en esta función es el siguiente:

Recepción y decodificación de datos:

- Cuando `Serial1` tiene datos disponibles, estos se procesan utilizando la función `gps.encode()`. Esta función decodifica los mensajes de datos en formato *NMEA* provenientes del *GPS* y almacena la información en la instancia `gps` de la clase *TinyGPSPlus*.

Verificación de la validez de los datos:

- Si `gps.encode()` detecta un mensaje completo de datos *GPS* válido, se llama a la función `displayInfo()`, que muestra los datos procesados (ubicación, fecha y hora) en el monitor serial.
- Si no se reciben suficientes caracteres de *GPS* dentro de los primeros 5 segundos (menos de 10 caracteres), se muestra un mensaje de error en el monitor serial indicando que no se detecta el *GPS*. Esto podría deberse a un problema de cableado, en cuyo caso el sistema se detiene para evitar lecturas incorrectas.

Función `displayInfo()`

La función `displayInfo()` se encarga de presentar los datos de ubicación, fecha y hora en el monitor serial.

Visualización de la ubicación (latitud y longitud):

- Si los datos de ubicación son válidos (`gps.location.isValid()`), se muestran la latitud y longitud con una precisión de hasta seis decimales. Esta precisión permite obtener coordenadas exactas del dispositivo en el mapa.
- Si los datos no son válidos, se muestra el mensaje “**INVALIDO**” en lugar de las coordenadas.

Visualización de la fecha:

- Si el *GPS* proporciona una fecha válida (`gps.date.isValid()`), esta se presenta en formato MM/DD/AAAA.
- Si la fecha no es válida o no se ha recibido, se muestra “INVALIDO” en su lugar.

Visualización de la hora:

- La hora se muestra en formato HH:MM:SS.CC (horas, minutos, segundos y centésimas de segundo) si los datos son válidos (`gps.time.isValid()`).
- Si la hora no es válida, se muestra “INVALIDO”.

4.3.2. Implementación y configuración del módulo LoRaWAN para la transmisión de datos

En esta fase, se llevó a cabo la configuración del sistema de transmisión de datos mediante la consola *Helium (Legacy)* [?], junto con la placa de desarrollo Heltec WiFi LoRa 32 V3. A continuación, se detallan los pasos y configuraciones necesarios para establecer la comunicación entre el dispositivo y la red *Helium*, así como los procesos requeridos para su correcto funcionamiento.

Uso de la consola *Helium*

La consola *Helium* se empleó para configurar los parámetros de los nodos, verificar el estado de la red y monitorear las transmisiones de datos. A través de esta plataforma, se prepararon los entornos necesarios para visualizar la conectividad entre los dispositivos y la consola, incluyendo el registro de los dispositivos y la verificación de su comunicación con los *gateways*.

El proceso comenzó con el acceso a una organización en la que se habían otorgado permisos de usuario. Tras recibir estos permisos, se procedió a crear una cuenta en la plataforma *Helium*. Completado el registro y acceso a la cuenta, fue posible crear un nodo en la red *Helium*, configurado específicamente para recibir los datos transmitidos por el dispositivo. La Figura 4.7 ilustra el entorno de configuración en la consola *Helium*, donde se han resaltado con recuadros negros los elementos clave, incluyendo las credenciales principales necesarias para establecer la conexión.

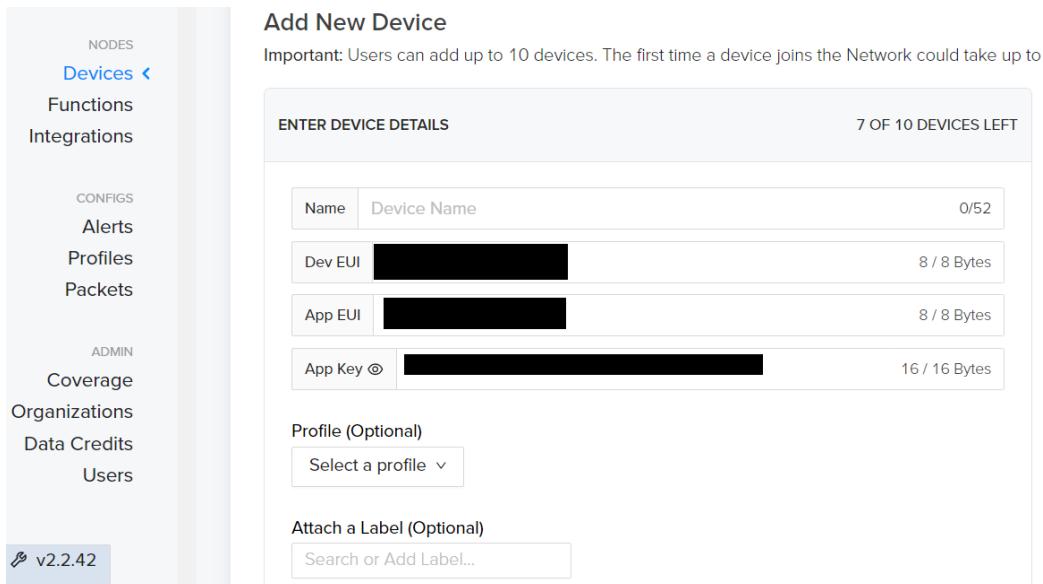


Figura 4.7: Entorno para crear dispositivo y configuración de las credenciales necesarias para el nodo.

Configuración del dispositivo Heltec WiFi LoRa 32 V3

Luego de crear el nodo y obtener las credenciales necesarias, el siguiente paso fue la configuración de la placa de desarrollo Heltec WiFi LoRa 32 V3. Para la implementación inicial, se utilizó la documentación oficial de *Heltec Automation* [?], que proporciona instrucciones detalladas paso a paso sobre la configuración del módulo *LoRaWAN*, así como las librerías requeridas para su funcionamiento.

Una vez completada esta configuración, se consultó la guía oficial de *Helium* [?], para su integración en el entorno Arduino IDE, adaptándola a las especificaciones particulares de la placa. Esta guía proporcionó los pasos necesarios para registrar las credenciales generadas, permitiendo que el dispositivo enviara datos en *uplink* a la red *Helium*. En la Figura 4.8, se muestran los ajustes de los parámetros necesarios para la comunicación con el servidor *LoRaWAN*.

```

2
3 #include "LoRaWan_APP.h"
4
5 /* OTAA para*/
6 uint8_t devEui[] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 };
7 uint8_t appEui[] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 };
8 uint8_t appKey[] = { 0x88, 0x66, 0x01 };
9
10 /* ABP para*/
11 uint8_t nwkKey[] = { 0x15, 0xb1, 0xd0, 0xef, 0xa4, 0x63, 0xdf, 0xbe, 0x3d, 0x11, 0x18, 0x1e, 0x1e, 0xc7, 0xda, 0x85 };
12 uint8_t appKeyExt[] = { 0x17, 0x3c, 0x78, 0x75, 0x9c, 0x0d, 0x6a, 0xb5, 0x5c, 0x43, 0x77, 0x9d, 0x16, 0x67 };

```

Figura 4.8: Ajuste de los parámetros necesarios en el dispositivo para la comunicación con el servidor *LoRaWAN*.

4.3.3. Implementación de la comunicación entre los módulos *GNSS* y *LoRaWAN*

Una vez configurados y validados de manera independiente tanto el módulo *GNSS* como el sistema de comunicación *LoRaWAN*, se procedió con la integración de ambos componentes. El objetivo de esta fase fue garantizar que las coordenadas obtenidas por el módulo *GNSS* se transmitieran de manera eficiente a través de la red *LoRaWAN*. Para ello, se realizaron los ajustes necesarios en el código, asegurando que la captura de los datos de ubicación y su envío a través de *LoRaWAN* ocurrieran de manera sincronizada y sin pérdida de información.

En esta sección, se describen en detalle los procesos y el funcionamiento del código ajustado para integrar los sistemas de geolocalización y transmisión de datos.

Como primer paso, se diseñó un diagrama de flujo para representar el funcionamiento del sistema integrado, el cual se puede observar en la Figura 4.9.

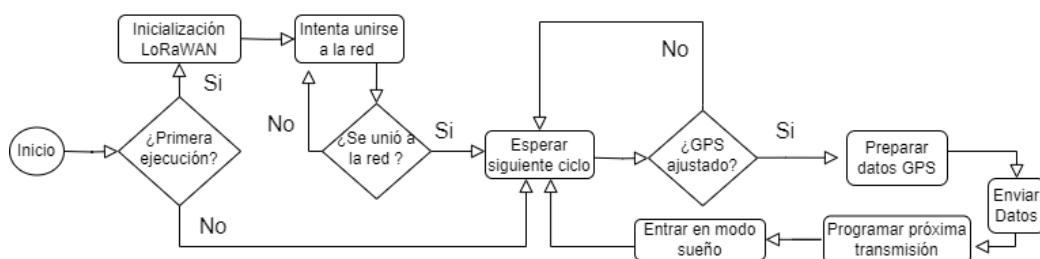


Figura 4.9: Diagrama de flujo de transmisión de datos por *LoRaWAN*.

Una vez diseñado el diagrama de flujo, se procedió a desarrollar una nueva versión del código, reutilizando los fragmentos de código de las fases

anteriores e incorporando los ajustes necesarios. A continuación, se describe la implementación de este código.

El programa inicia con la inclusión de las librerías esenciales: `LoRaWANAPP.h`, para gestionar la comunicación mediante *LoRaWAN*, y `TinyGPSPlus.h`, para manipular los datos obtenidos del módulo *GNSS*. Luego, se configuran los pines de acuerdo con las definiciones establecidas en la primera fase del proyecto.

A continuación, se definen los parámetros de conexión de *LoRaWAN*. La conexión puede configurarse de dos maneras: mediante *OTAA* (*Over-The-Air Activation*) o *ABP* (*Activation By Personalization*). Para ello, se especifican las claves necesarias para cada método (`devEui`, `appEui`, `AppKey` para *OTAA*, y `NwkSKey`, `AppSKey`, `devAddr` para *ABP*). También se definen los canales de comunicación y la región de operación.

Posteriormente, se establece un ciclo de transmisión de datos cada 60 segundos (`appTxDutyCycle` a 60000) y se especifican otros parámetros, como el uso de *ADR* (*Adaptive Data Rate*) y la configuración de los mensajes como confirmados o no confirmados, según los requisitos del sistema.

La función `prepareTxFame` se encarga de preparar los datos *GPS* para su envío. Primero, el programa espera hasta obtener una ubicación válida (`gps.location.isValid()`). Una vez que la señal es válida, la función extrae la latitud y longitud y las almacena en un *buffer* de datos (`appData`) en el formato adecuado para su transmisión mediante *LoRaWAN*.

Al comenzar el ciclo `setup`, el programa inicializa la comunicación serie y *LoRaWAN*. Adicionalmente, se utiliza una variable (`firstrun`) para verificar si es la primera vez que el dispositivo se configura en el *MCU*, mostrando un mensaje en pantalla para confirmar la inicialización.

En el ciclo `loop`, la lógica del programa sigue una estructura de máquina de estados, que permite gestionar cada fase de la operación *LoRaWAN*. Los estados principales son:

- **DEVICE STATE INIT:** Inicializa el módulo *LoRaWAN* y realiza las configuraciones básicas.
- **DEVICE STATE JOIN:** Intenta unirse a la red *LoRaWAN*, mostrando mensajes de estado.
- **DEVICE STATE SEND:** Llama a `prepareTxFame` para obtener y empaquetar los datos de *GPS*, y luego envía la información.
- **DEVICE STATE CYCLE:** Calcula el tiempo hasta la próxima transmisión, usando `appTxDutyCycle` como base.

- **DEVICE STATE SLEEP:** Pone al dispositivo en modo de bajo consumo para conservar energía hasta la siguiente transmisión.

Finalmente, la función `displayInfo` muestra en el monitor serie la información de ubicación, fecha y hora del *GPS*. En caso de que los datos no estén disponibles, se muestra un mensaje de *INVALIDO*.

4.4. Módulo de interfaz gráfica y monitoreo

El módulo de interfaz gráfica y monitoreo constituye un componente esencial del sistema, proporcionando una plataforma interactiva *IoT* para visualizar en tiempo real la ubicación de la motocicleta y otros parámetros relevantes. Para esta implementación, se utilizó *ThingsBoard Community Edition*, que se conectó a la consola *Helium*, permitiendo la recepción y visualización de los datos transmitidos desde la red. La estructura del módulo incluye un servidor local que procesa y organiza la información en una interfaz gráfica accesible para el usuario final.

La implementación de este módulo se dividió en dos fases principales. La primera fase consistió en la configuración de *ThingsBoard* como receptor de datos, habilitando el procesamiento y la visualización de la información recibida. La segunda fase abordó la integración de la consola *Helium* con *ThingsBoard* mediante el protocolo *MQTT*, configurando un flujo de datos en tiempo real entre el dispositivo y el servidor local.

En las siguientes secciones, se detallan cada una de estas fases:

4.4.1. Configuración de *ThingsBoard*

En esta fase, se llevó a cabo la instalación y configuración de *ThingsBoard* de manera local, siguiendo las instrucciones de la documentación oficial para asegurar una implementación inicial estable [?]. Una vez completada esta instalación, se realizaron configuraciones personalizadas con el objetivo de adaptar la plataforma a los requisitos específicos del sistema de monitoreo.

Los pasos realizados en esta etapa incluyeron:

Creación de dispositivos y configuración de conectividad

En *ThingsBoard*, se añadieron los dispositivos que transmitirían datos al sistema. En este caso, se creó un dispositivo llamado *Poli_Moto* (ver Figura 4.10) y se le asignó un perfil *MQTT*. Esta configuración permite que el dispositivo envíe datos de telemetría a través de la consola *Helium* hacia *ThingsBoard*.

Se definió un perfil de transporte *MQTT* con los temas de telemetría y atributos configurados para recibir datos en formato *JSON* (ver Figura 4.11). Esto asegura que los datos de ubicación y estado se transmitan de manera continua y en tiempo real.

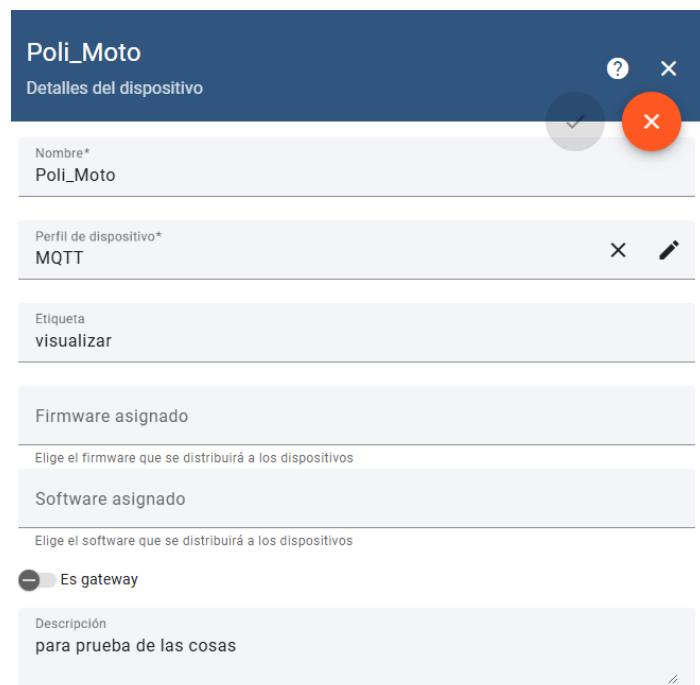


Figura 4.10: Detalles del dispositivo ThingsBoard.

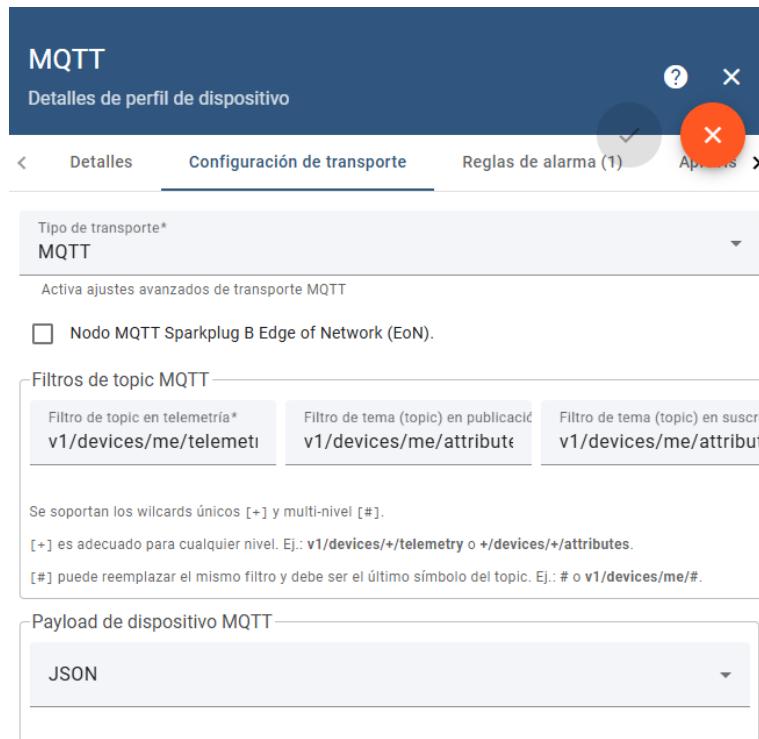


Figura 4.11: Perfil de dispositivo ThingsBoard.

Gestión de Usuarios y Roles de Acceso

Se crearon diferentes usuarios y clientes para facilitar el acceso y administración de los datos en el sistema. En la configuración de clientes, se creó un cliente de prueba llamado *Cliente_Prueba*, al cual se le asignaron dispositivos específicos y se configuraron sus permisos de acceso (ver Figuras 4.12 y 4.13).

Se configuraron usuarios con permisos exclusivos para acceder a dispositivos asignados, permitiendo una gestión controlada y personalizada de la información disponible.

	Nombre	Apellido	Email
<input type="checkbox"/>	Mario	Barreto	kutumaio@gmail.com

Figura 4.12: Cliente con usuario creado en ThingsBoard.

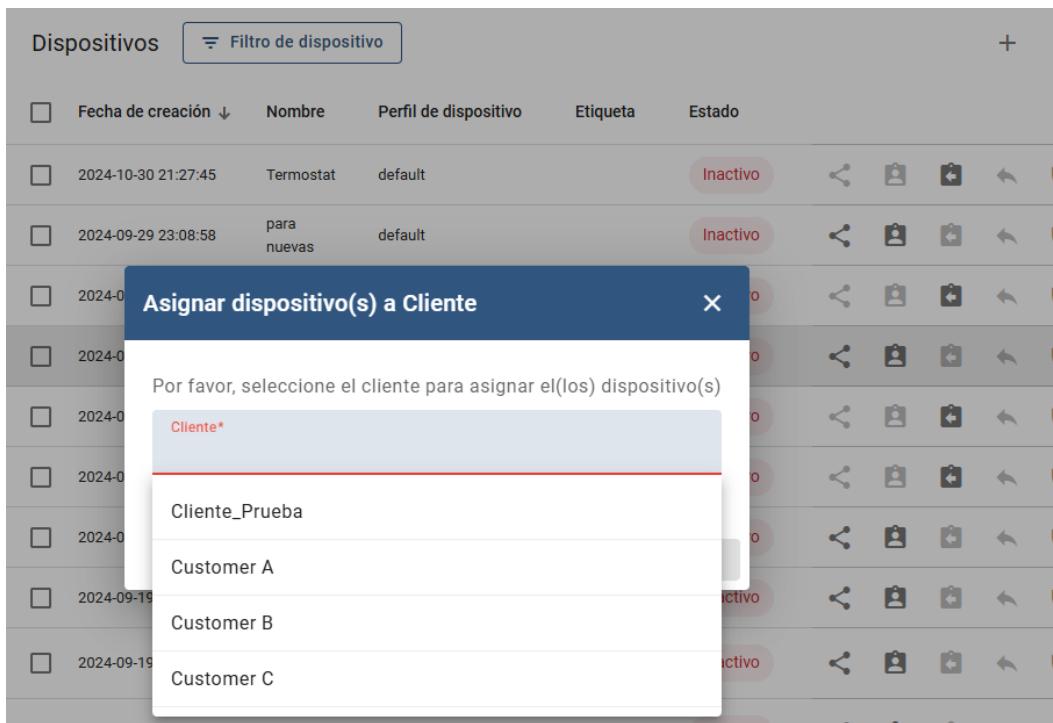


Figura 4.13: Asignación de dispositivos a clientes en ThingsBoard.

Diseño del Tablero Interactivo

Se diseñó un tablero personalizado titulado *Mapa_Mot*, donde se agruparon los datos de los dispositivos en widgets específicos para su monitoreo en tiempo real. Este tablero permite centralizar y visualizar la información de ubicación y otros datos relevantes de los dispositivos conectados.

Se exploraron los *widgets* disponibles en ThingsBoard para elegir aquellos que mejor se adaptaran a las necesidades del sistema (ver Figura 4.14). Se seleccionaron los *widgets* de mapas y gráficos de series de tiempo, esenciales para monitorear la posición geográfica y el historial de movimiento del dispositivo en tiempo real.

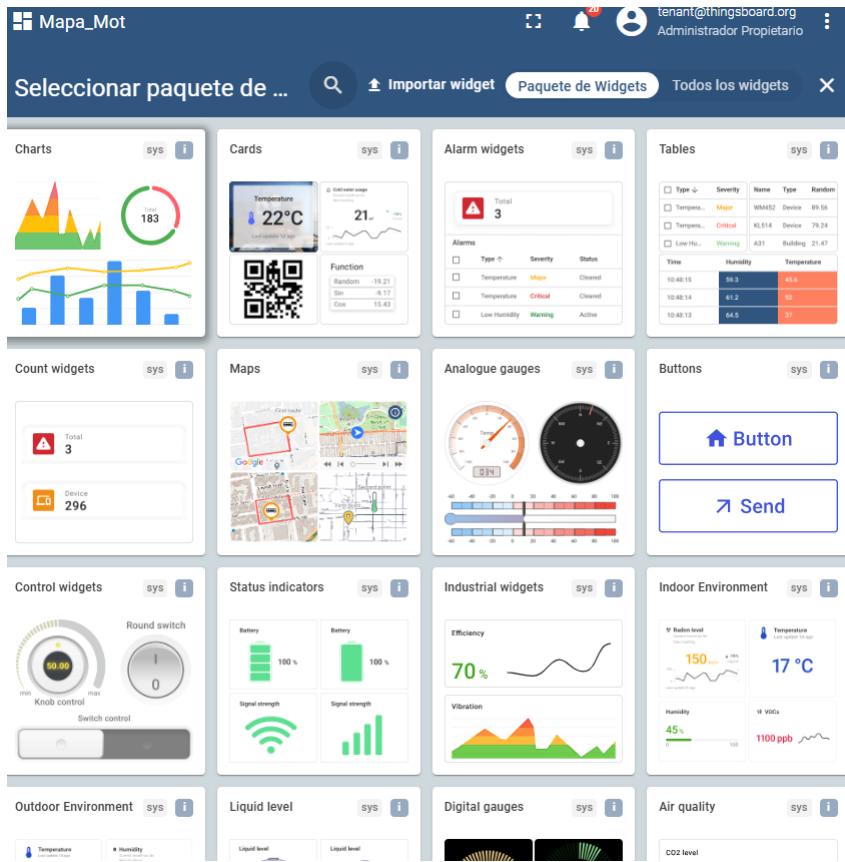


Figura 4.14: *Widgets* disponibles en *ThingsBoard*.

Se añadieron *widgets* de mapa, como *Route Map - OpenStreet* y *Route Map - Google*, que ofrecen una visualización interactiva de la ubicación en tiempo real (ver Figura 4.15). Además, se incorporó un *widget* de tabla de series temporales para analizar datos históricos como latitud y longitud.

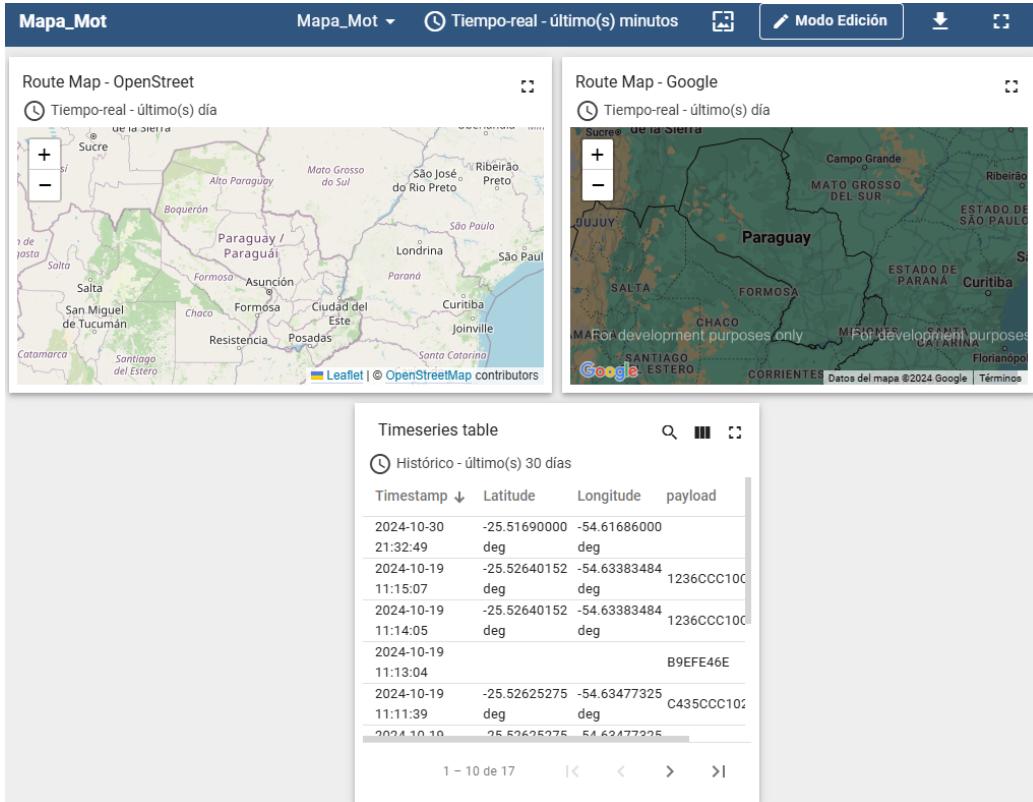


Figura 4.15: *Widgets* utilizados en *ThingsBoard*.

Para garantizar la precisión, se configuró una ventana de tiempo en los *widgets*, visualizando información del último día. Se definieron claves como latitud, longitud y estado de actividad para asegurar que los datos mostrados sean precisos (ver Figura 4.16).

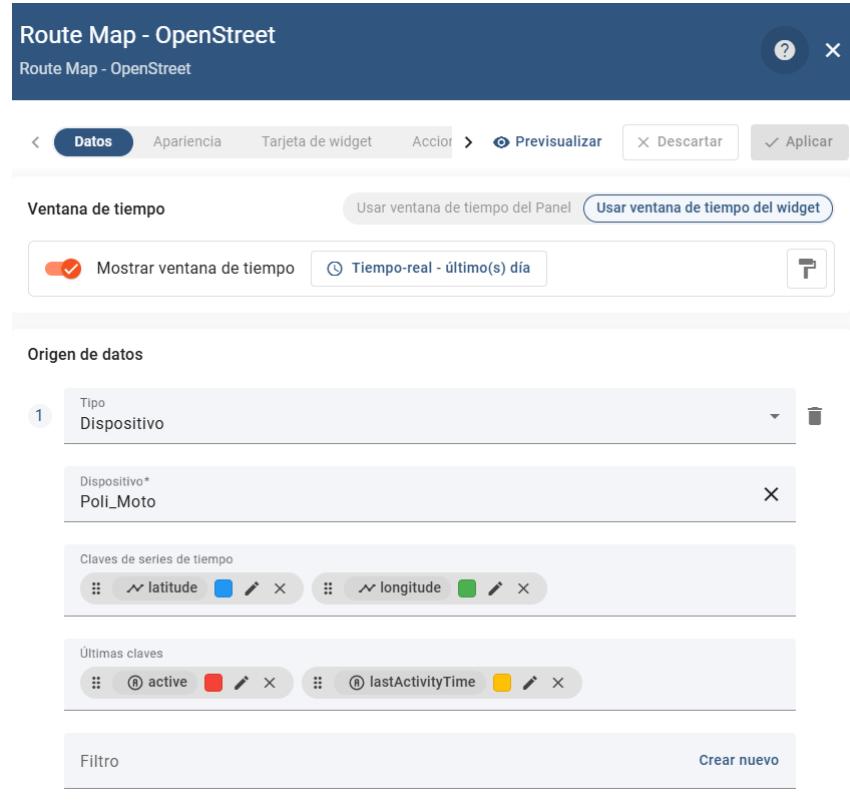


Figura 4.16: Configuraciones en *Widgets* de Mapa.

Definición de las *Rule Chains* en el Motor de Reglas

En el motor de reglas (*Rule Engine*), se configuraron cadenas de reglas personalizadas para procesar los mensajes recibidos. Se creó una cadena llamada *MQTT*, con nodos como un decodificador base64 y hexadecimal, para transformar los datos antes de almacenarlos en series temporales (ver Figura 4.17).

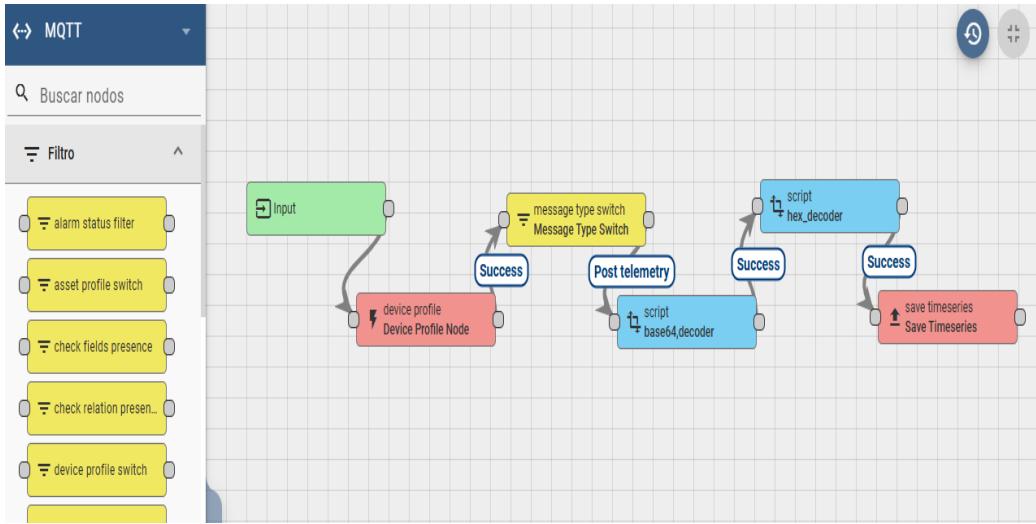


Figura 4.17: Diagrama del motor de reglas *MQTT* en *ThingsBoard*.

Configuración de Alarmas y Notificaciones

Se configuraron alarmas para generar alertas en eventos específicos, como desconexiones del dispositivo o valores de telemetría fuera de parámetros. Estas alarmas mejoran la capacidad de respuesta del sistema ante eventos críticos.

Todas las configuraciones siguieron las mejores prácticas de la documentación oficial de *ThingsBoard* [?].

4.4.2. Integración de *ThingsBoard* con *Helium*

La siguiente etapa consistió en la integración de *ThingsBoard* con la consola *Helium*. Para integrar estas herramientas y lograr una transmisión de datos en tiempo real, se configuró una integración a través del protocolo *MQTT*. Este proceso permitió recibir datos de telemetría desde dispositivos conectados a la red *Helium* y visualizarlos en el tablero de *ThingsBoard*. A continuación, se detallan los pasos realizados para establecer esta integración.

Primero, se accedió al menú de *Integrations* en la consola de *Helium*, señalado con la flecha (A) en la Figura 4.18. En este menú, se seleccionó la opción para añadir una nueva integración, lo que despliega un conjunto de opciones básicas de integración. Entre estas opciones, se encuentra *MQTT*, identificado con el círculo (B) en la Figura 4.18, que fue el protocolo elegido para transmitir los datos de los dispositivos a *ThingsBoard*.

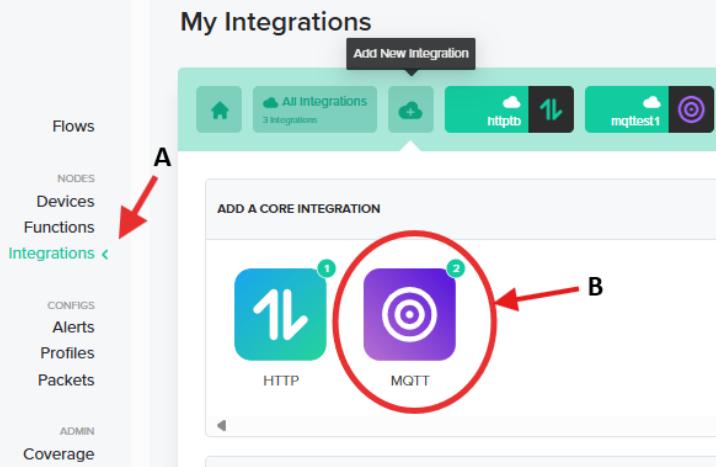


Figura 4.18: Menú de integraciones en la consola de *Helium*.

Una vez seleccionada la opción *MQTT*, se procedió a configurar los detalles de conexión en la consola de *Helium*, como se muestra en la Figura 4.19. En la sección *MQTT Connection Details*, se definieron varios parámetros clave para establecer la comunicación con el servidor de *ThingsBoard*:

En el campo Endpoint, señalado con la etiqueta (A) en la Figura 4.19, se incluyó el token único del dispositivo generado previamente en *ThingsBoard*, el cual se añadió directamente en la URL como credencial de acceso. La dirección IP pública del servidor y el puerto estándar para *MQTT* (1883), marcados con la etiqueta (B) en la Figura 4.19, fueron configurados para garantizar la correcta vinculación con el servicio remoto. Finalmente, se establecieron los temas necesarios para la comunicación señalados con la flecha (C) en la Figura 4.19. El tema *Uplink Topic*, fue configurado como v1/devices/me/telemetry, permitiendo el envío de datos de telemetría desde los dispositivos, el tema *Downlink Topic*, se configuró para recibir comandos y configuraciones desde *ThingsBoard* hacia los dispositivos. Con esta configuración, se garantizó una comunicación bidireccional entre ambos sistemas, facilitando tanto la gestión remota como el monitoreo en tiempo real de los dispositivos conectados.

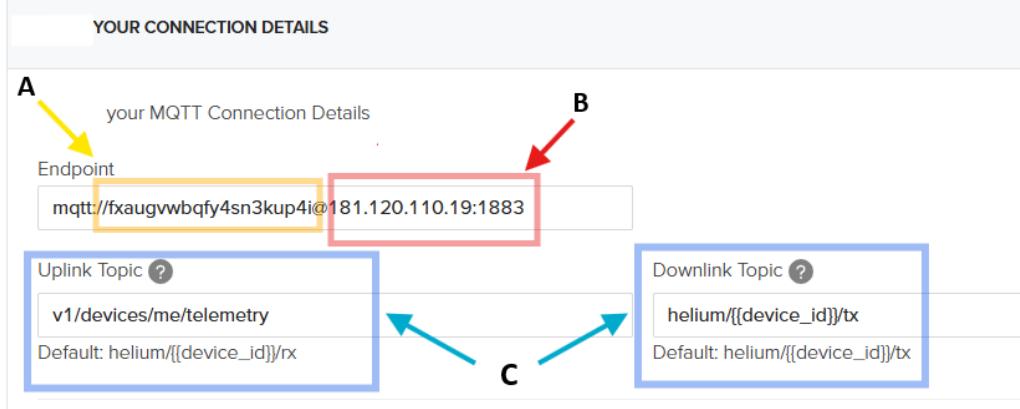


Figura 4.19: Detalles de conexión *MQTT* configurados en la consola de *Helium*.

Después de configurar los detalles de conexión, se procedió a crear un flujo en la consola de *Helium* que conecta los dispositivos a la integración *MQTT_TB*, como se observa en la Figura 4.20. En el menú Flows (A), se seleccionó la integración previamente configurada, etiquetada como *MQTT_TB* (B) en la Figura 4.20, y se vinculó con el dispositivo PruebaHeltec. Este flujo garantiza que los datos capturados por el dispositivo se envíen automáticamente a *ThingsBoard* a través de la red *Helium*.

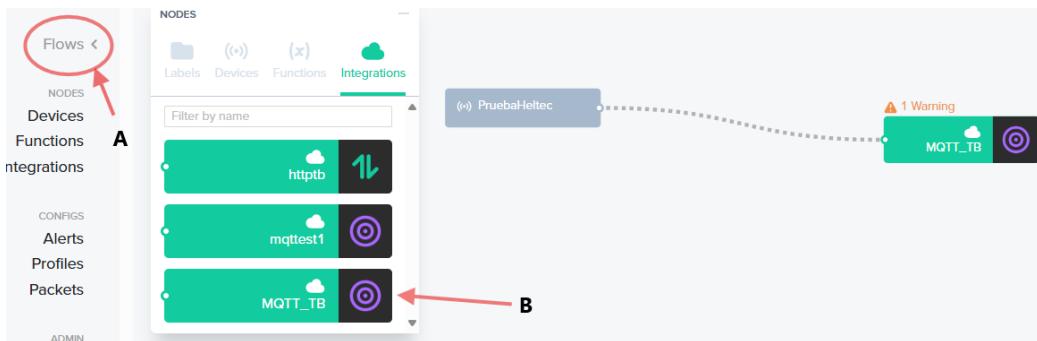


Figura 4.20: Flujo de integración entre el dispositivo PruebaHeltec y *MQTT_TB*.

Para completar la integración, fue necesario configurar la redirección de puertos en el router de la red local donde se encuentra alojado el servidor **ThingsBoard**. En la sección *IPv4 Port Mapping* del panel de administración

del *router* (ver Figura 4.21, se realizó el mapeo correspondiente para permitir el acceso externo al servidor *MQTT*. Se definió un nombre de mapeo denominado *TBMQTT*, señalado en (A) de la Figura 4.21, que asocia la dirección IP interna del servidor (192.168.100.5) con la IP pública asignada a la red (181.94.228.197) indicado con (B) de la Figura 4.21). Asimismo, identificado con el círculo (C) en la Figura 4.21 ,se estableció el puerto 1883 como punto de entrada para las conexiones externas, y se configuró el protocolo TCP, necesario para garantizar la correcta transmisión de los datos.

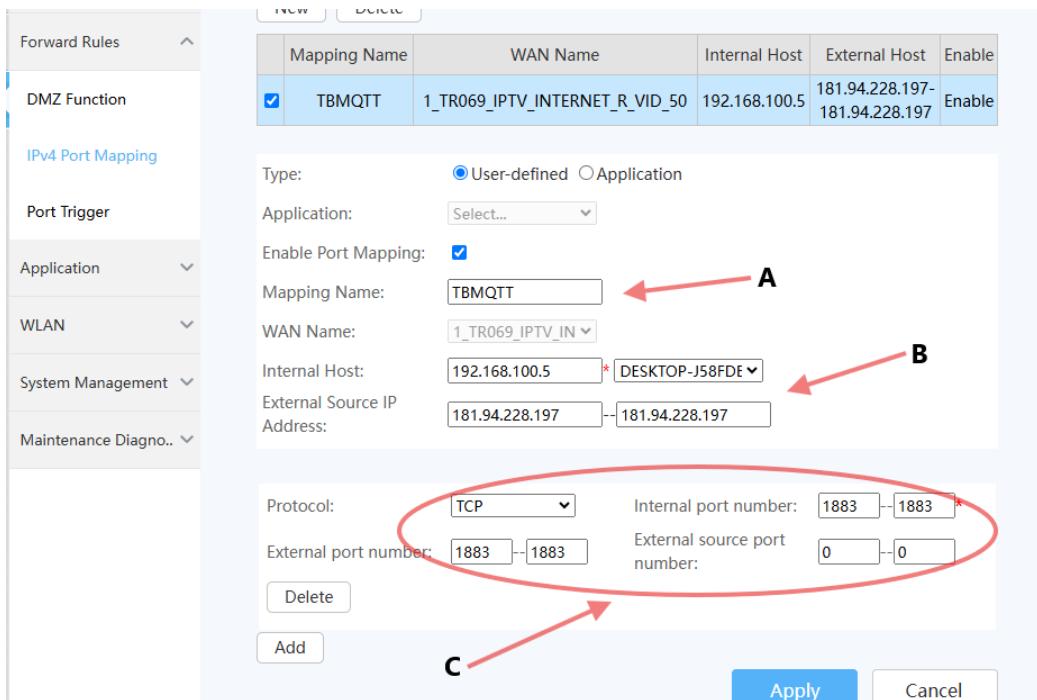


Figura 4.21: Configuraciones en el *router* de la red local.

Con estos ajustes, se completó la integración de *ThingsBoard* con *Helium*, permitiendo que los datos capturados por los dispositivos en la red *Helium* sepuedan transmitir de manera funcional al servidor *ThingsBoard* para su visualización y monitoreo en tiempo real.

4.5. Integración de componentes y encapsulamiento

En esta sección, se describe el proceso de integración de los distintos módulos desarrollados en el sistema y el encapsulamiento final del dispositivo. El objetivo de esta fase es lograr que los módulos de autenticación, control, captura y transmisión de datos trabajen de forma conjunta. Además, se diseñaron un encapsulado protector y estético para asegurar la robustez y usabilidad del dispositivo en su entorno final. A continuación, se detallan las etapas de integración y encapsulado del sistema.

Este proceso se dividió en cuatro etapas:

- Implementación Básica con la Integración Inicial entre los Módulos RFID y LoRaWAN.
- Implementación alternativa con la Integración ajustada entre los Módulos.
- Incorporación de Corte de Energía y Simulación de Arranque
- Modelado y Diseño de Encapsulado 3D.

4.5.1. Implementación Básica con Integración Inicial de los Módulos RFID y LoRaWAN

En esta primera etapa, se integraron los módulos de autenticación RFID y el sistema de comunicación LoRaWAN. El objetivo de esta fase fue asegurar que el sistema pudiera autenticar correctamente los UIDs autorizados y transmitirlos de manera confiable a través de la red LoRaWAN.

Para establecer la comunicación entre el módulo RFID y la placa de desarrollo WiFi LoRa 32 V3, se emplearon las mismas conexiones definidas previamente en el módulo de autenticación y control, lo cual permitió mantener una estructura coherente en el diseño. Esta configuración inicial proporcionó las bases para evaluar el correcto funcionamiento de los módulos integrados y realizar los ajustes necesarios en las etapas siguientes.

4.5.2. Implementación alternativa con la Integración ajustada entre los Módulos.

Tras implementar la primera versión del código, se identificaron conflictos en la comunicación SPI entre los módulos RFID y LoRaWAN. Se observó

que el módulo LoRaWAN utilizaba el protocolo SPI con conexiones definidas internamente en la placa de desarrollo, mientras que el módulo RFID, al conectarse externamente mediante SPI, generaba un solapamiento que impedía el funcionamiento coordinado de ambos sistemas.

Para abordar los problemas detectados, se diseñó una nueva versión del código que incorporó una integración alternativa con el uso de un ESP-WROOM-32 dedicado exclusivamente a gestionar el módulo RFID. Este ajuste tuvo como propósito lograr que los componentes funcionaran sin interferencias en la comunicación, permitiendo una transmisión estable y una sincronización efectiva entre los módulos.

Para establecer la comunicación entre el módulo RFID RC522 y la placa de desarrollo ESP-WROOM-32, se utilizó el protocolo SPI. Las conexiones se realizaron de la siguiente manera: el pin 4 se utilizó como CS (Chip Select), el pin 23(MOSI), el pin 18(SCK), el pin 19(MISO) y el pin 15(RESET). La alimentación se proporcionó a través de los pines VCC y GND.

Para la comunicación entre la placa ESP-WROOM-32 y la Heltec WiFi LoRa 32 V3, se empleó el protocolo UART, con las siguientes conexiones: el pin 48 de la WiFi LoRa 32 V3 se conectó al RX del ESP-WROOM-32, el pin 47 de la WiFi LoRa 32 V3 al TX del ESP-WROOM-32, y los pines GND de ambas placas se unieron para asegurar una referencia común de tierra.

El módulo GNSS se conectó utilizando las mismas conexiones definidas previamente en el módulo de captura y transmisión de datos. En la figura 4.22, se presenta el diagrama de conexiones empleado para esta configuración.

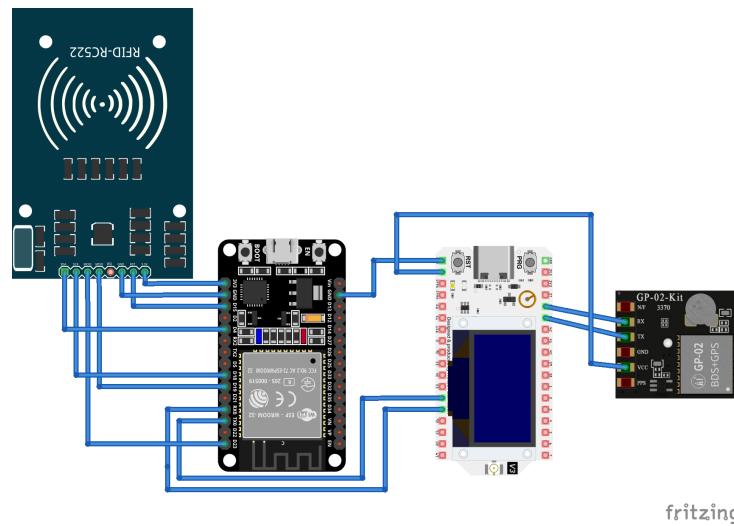


Figura 4.22: Diagrama de conexiones entre RFID RC522, ESP-WROOM-32, WiFi LoRa 32 V3 y GNSS.

Luego de realizar las conexiones, se procedió a ajustar los códigos. En la placa de desarrollo ESP-WROOM-32, se cargó una segunda versión del código previamente desarrollado para el módulo de autenticación y control. Se realizaron modificaciones en el archivo rfidfunciones.cpp, específicamente en la función mostrarUID, que fue adaptada para crear una cadena llamada uidString, la cual almacena el UID en un formato legible.

Durante el procesamiento de cada byte del UID, la función verifica si el valor es menor a 0x10, en ese caso, agrega un 0 al inicio para mantener un formato uniforme. Cada byte se convierte luego a su representación hexadecimal y se añade a uidString, utilizando : como separador entre los bytes. Además, la función almacena cada byte en un array llamado LecturaUID, permitiendo que el UID esté disponible para otras partes del programa cuando sea necesario.

Finalmente, la función imprime el uidString completo en el monitor serial, mostrando el UID del tag RFID en el formato deseado para su fácil identificación y verificación.

En esta fase, se añadieron múltiples tareas al sistema para gestionar de manera coordinada la comunicación entre módulos. Estas tareas, cargadas en la placa de desarrollo Heltec WiFi LoRa 32 V3, se organizaron de la siguiente manera:

- **Tarea UART (Recepción de Datos RFID):** Esta tarea permanece en escucha constante de los datos recibidos desde el módulo RFID a través de la UART. Al recibir un UID, este se compara con los UID previamente autorizados. En caso de que el tag recibido sea autorizado, la tarea ajusta el estado del sistema, habilitando el envío del UID por LoRaWAN y activando la tarea de transmisión de LoRaWAN.
- **Tarea LoRaWAN:** La tarea de LoRaWAN sigue una máquina de estados que controla las diferentes etapas de transmisión, incluyendo la inicialización, la unión a la red (join), el envío de datos y el ciclo de suspensión (sleep). Esta estructura permite que la tarea transmita los datos de forma programada y que el dispositivo entre en suspensión tras cada transmisión, optimizando así el consumo de energía.
- **Tarea GPS:** Esta tarea permanece inactiva hasta que se obtienen coordenadas GPS válidas. Una vez que las coordenadas de latitud y longitud han sido obtenidas y verificadas, la tarea las almacena y las marca como listas para ser enviadas en el siguiente ciclo de transmisión de LoRaWAN. Esto asegura que el sistema envíe únicamente datos de localización precisos.

Con los módulos RFID, GNSS y LoRaWAN completamente integrados, se realizaron ajustes finales en el código para definir el funcionamiento general del sistema, que quedó organizado de la siguiente manera:

- **Detección RFID:** El sistema identifica si un tag RFID autorizado está presente. Al detectar un UID autorizado, el sistema se encarga de enviarlo a través de LoRaWAN.
- **Envío de Datos por LoRaWAN:** Según el tipo de mensaje seleccionado, puede enviar el UID del RFID, otros datos relevantes o, si están disponibles, las coordenadas GPS.
- **Ciclo de LoRaWAN:** El sistema opera en un ciclo de transmisión periódico. Durante cada ciclo, envía los datos y luego entra en modo de suspensión (sleep) hasta el siguiente ciclo, optimizando así el consumo energético.
- **Validación de Coordenadas GPS:** El sistema realiza intentos continuos de obtener coordenadas GPS válidas. Una vez que estas coordenadas son capturadas con éxito, se preparan para ser enviadas en el próximo ciclo de transmisión de LoRaWAN.

4.5.3. Incorporación de Corte de Energía y de Arranque

En esta fase, se llevaron a cabo los ajustes finales mediante la incorporación de un módulo de relé, el cual se sincronizó con el módulo RFID para gestionar el corte de energía y el arranque del sistema. Para implementar esta funcionalidad, se añadió una nueva tarea al código, denominada vTaskCorriente. Esta tarea permanece a la espera de recibir una notificación que indique el estado del relé. Cuando la notificación recibida desde la tarea UART (Recepción de Datos RFID) es positiva, el relé se activa, permitiendo el arranque del sistema, en caso contrario, el relé se desactiva, interrumpiendo la corriente y apagando el sistema. Para ilustrar la estructura y conexiones del sistema, se diseñó un diagrama esquemático completo, que se presenta en la Figura 4.23.

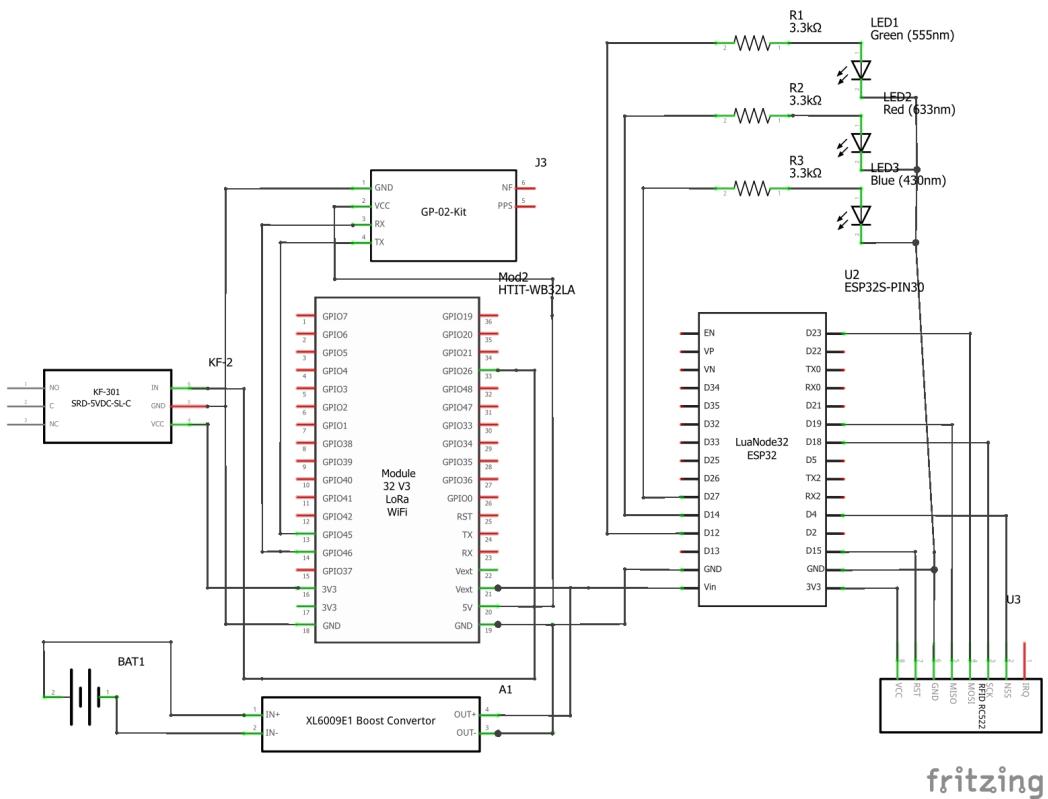


Figura 4.23: Diagrama de conexiones de todos los módulos.

Además, se integraron tres LEDs para indicar los estados del sistema. El LED rojo realiza un “self-check” del módulo RFID, indicando que este aún no está listo. Una vez completada la inicialización del RFID, el LED verde se enciende, señalizando que el sistema está preparado para la lectura. Cuando un tag es detectado y autorizado, el LED azul se enciende para confirmar el acceso. Esta señalización mediante LEDs facilita el monitoreo visual del estado y funcionamiento del sistema en tiempo real.

Por último, se añadió un nuevo tipo de mensaje al sistema de transmisión por LoRaWAN para alertar en caso de desconexión, informando sobre el estado de corte de energía.

4.5.4. Modelado y Diseño de Encapsulado 3D.

En esta fase, se comenzó definiendo la disposición y estructura de los componentes en el sistema. En primer lugar, se determinó la ubicación óptima de cada módulo, de manera que los elementos clave estuvieran organizados

de forma compacta y funcional. Este paso fue esencial para facilitar el acceso a los conectores y puertos de cada componente, optimizando el espacio y garantizando una configuración estructurada.

En la Figura 4.24, se presenta la distribución de los componentes, donde puede observarse cómo cada módulo está colocado estratégicamente para minimizar el uso de cables y asegurar una accesibilidad adecuada. Esta disposición garantiza que los puertos y conexiones principales sean accesibles para su mantenimiento.

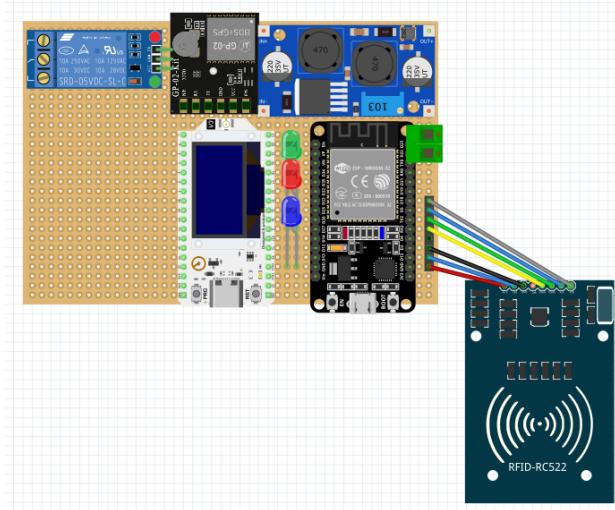


Figura 4.24: Disposición estructural de los componentes.

Con la disposición estructural establecida, se avanzó a la etapa de diseño del encapsulado 3D mediante el software Blender. Este encapsulado fue modelado específicamente para proteger los componentes, adaptándose a sus dimensiones y ubicación en el shield. La estructura fue diseñada para proporcionar una cobertura robusta y duradera, evitando el movimiento de los componentes internos y protegiéndolos de factores externos.

En la figura 4.25, se puede observar el modelo 3D del encapsulado final, que incluye aperturas específicas para cada puerto y conector, así como detalles adicionales para facilitar el montaje y desmontaje de componentes en caso de futuras modificaciones o mantenimientos.

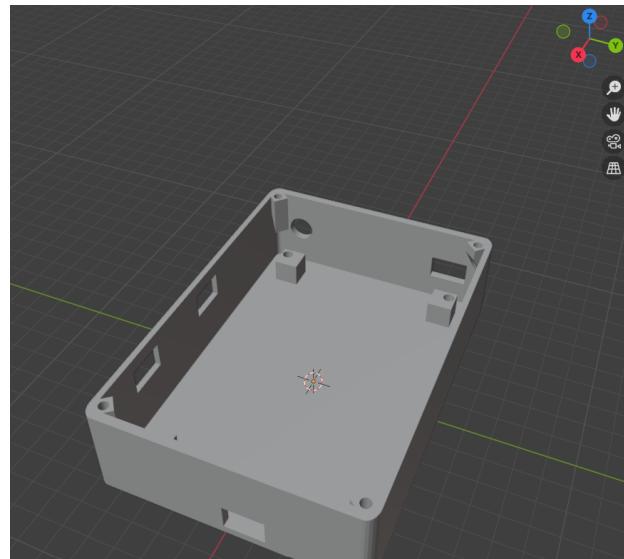


Figura 4.25: Base del encapsulado.

En la figura 4.26, se observa el diseño de una tapa destinada a cubrir el encapsulado, proporcionando una capa adicional de protección para los componentes internos. Esta tapa fue modelada para ajustarse de forma precisa al contorno del encapsulado, incluyendo accesos a los conectores que permiten la salida de los cables necesarios para el funcionamiento del sistema.

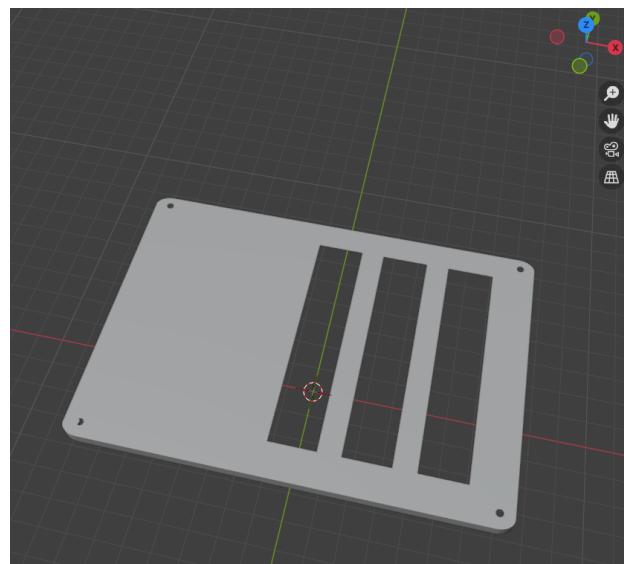


Figura 4.26: Parte superior del encapsulado.

Adicionalmente, se desarrolló un diseño 3D complementario para el módu-

lo RFID con el propósito de satisfacer la necesidad de identificación del usuario y garantizar una lectura continua del tag. En la Figura 4.27, se muestra el diseño del encapsulado, el cual incorpora un orificio estratégicamente ubicado para alojar el tag en una posición fija. Este diseño asegura su permanencia dentro del rango de lectura, optimizando la funcionalidad del sistema y mejorando su integración con los demás componentes.

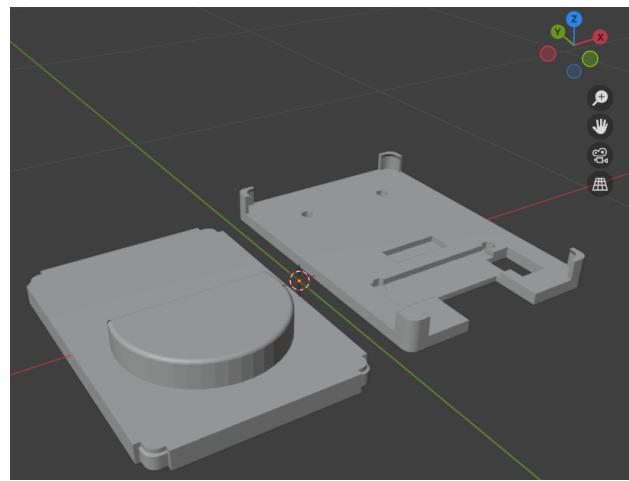


Figura 4.27: Modelado 3D del encapsulado para el módulo RFID.

Capítulo 5

Pruebas y resultados

En este capítulo se presentan las pruebas realizadas y los resultados obtenidos para cada uno de los módulos descritos en el Capítulo 4, evaluando su desempeño tanto de forma individual como integrada dentro del sistema. Estas pruebas permitieron validar la efectividad de cada componente y del sistema completo en condiciones reales. Posteriormente, se aborda el diseño y construcción del prototipo, destacando las características del hardware y software, la disposición estructural de los componentes y el encapsulado 3D, así como su montaje final en una motocicleta.

5.1. Pruebas en entornos controlado

Se aplicaron pruebas en entornos controlado con la finalidad de evaluar el funcionamiento de los distintos módulos del sistema en condiciones ideales, libres de interferencias externas o factores no previstos. Este entorno permitió realizar un análisis más preciso de la autenticación de los usuarios, la captura y transmisión de datos, así como la integración completa del sistema, garantizando que todas las funciones operen correctamente antes de ser sometidas a condiciones reales. Para la ejecución de estas pruebas, se utilizaron los siguientes materiales:

- Notebook Dell Inspiron 15 3535 equipada con un procesador AMD Ryzen 5 7530U, gráficos integrados Radeon, 8 GB de memoria RAM y cuenta con sistema operativo Windows 11 Pro.
- Entorno de desarrollo Arduino IDE version 2.3.3.
- Módulo RFID RC522.
- Módulo GNSS GP-02-KIT.

- Placa de desarrollo Heltec WiFi LoRa 32 V3.
- Placa de desarrollo ESP-WROOM-32.
- Consola Helium (Legacy).
- Thingsboard.

5.1.1. Pruebas del Módulo de autentificación y control

Se llevaron a cabo pruebas con el módulo RFID RC522, utilizando la placa desarrollo WiFi LoRa 32 V3 y el entorno de desarrollo Arduino IDE. Estas pruebas se realizaron con el objetivo de verificar el funcionamiento del módulo RFID y evaluar su capacidad para leer y comparar los identificadores únicos (UID) de tarjetas y llaveros RFID.

La prueba de la primera versión consta en tres partes, descritas a continuación:

Lectura del UID de la tarjeta/llavero

En esta fase, se acercó una tarjeta o llavero RFID al módulo RC522, permitiendo que el sistema leyera y mostrara el UID en el monitor serie en formato hexadecimal. Esta etapa inicial confirmó la capacidad del módulo para detectar y visualizar los datos de identificación de los dispositivos RFID.

Comparación de UID y control de acceso

Posteriormente, se implementó una función para comparar el UID leído con los valores previamente almacenados en el sistema. Esta verificación fue clave para determinar si el módulo podía diferenciar entre tarjetas o llaveros autorizados y no autorizados. El sistema respondió adecuadamente, permitiendo el acceso solo cuando el UID coincidía con los registros autorizados.

Combinación de lectura y comparación constante

Finalmente, se desarrolló una prueba de combinación, en la cual el sistema leía y comparaba de forma constante el UID de la tarjeta o llavero, verificando su presencia continua. Este proceso fue diseñado para probar una futura funcionalidad del sistema, en la que se definirá un tiempo específico para asegurar la autenticación continua del usuario. La Figura 5.1, muestra cómo el sistema monitorea constantemente la presencia del UID, lo cual es fundamental para mantener la autenticación activa mientras el usuario permanezca en proximidad.

The screenshot shows a terminal window titled "Serial Monitor". It displays a series of messages from a WiFi LoRa module. The messages indicate the module is welcoming a user, detecting tags within range, and eventually disconnecting a motorcycle tag that has moved outside the range.

```

Output  Serial Monitor X
Message (Enter to send message to 'WiFi LoRa 32(V3)' on 'COM8')
17:51:56.877 -> UID: B9 EF E4 6E      Bienvenido Usuario 1
17:51:59.931 -> Tag sigue dentro del rango.
17:52:00.615 -> Tag sigue dentro del rango.
17:52:00.853 -> Tag sigue dentro del rango.
17:52:01.071 -> Tag sigue dentro del rango.
17:52:01.872 -> Tag sigue dentro del rango.
17:52:02.119 -> Tag sigue dentro del rango.
17:52:02.554 -> Tag sigue dentro del rango.
17:52:12.663 -> Tag fuera de rango, desconectando la motocicleta...

```

Figura 5.1: Prueba del módulo RFID.

En la Tabla 5.1, se puede observar el número de pruebas realizadas para cada versión del código, así como los resultados obtenidos en términos de detección y comparación del UID de las tarjetas y llaveros RFID. Esta información proporciona una visión general del desempeño y funcionamiento del sistema en cada fase del desarrollo.

Tabla 5.1: Resultados de las pruebas de detección y comparación del UID

Versión del Código	Total de Pruebas	Prue...
Primera versión	10	
Segunda versión	30	

Las fallas observadas durante las pruebas fueron esporádicas y se debieron a errores humanos, problemas de conexión y ajustes manuales. En algunos casos, los cables se desconectaron accidentalmente durante el proceso, interrumpiendo temporalmente la comunicación del sistema. Otras fallas, aunque puntuales, estuvieron relacionadas con errores en el código que afectaron la detección y comparación del UID. Además, se identificaron dificultades al mantener el tag en posición fija con la mano, lo que ocasionó lecturas inestables en ciertos momentos.

5.1.2. Pruebas del Módulo de captura y transmisión de datos

Se llevaron a cabo pruebas específicas por separado para evaluar la capacidad del sistema en la captura de las coordenadas del módulo GNSS y su transmisión a través de la red LoRaWAN utilizando la consola Helium. En esta fase, se validó la correcta conexión a un hotspot, así como la correcta recepción de las coordenadas y la recepción de paquetes de datos en condiciones de laboratorio. Las pruebas se dividieron en tres partes, descritas a continuación:

- Prueba del módulo GNSS.
 - Prueba del módulo LoRa del Heltec.
 - Prueba de captura y transmisión de datos de localización.

Prueba del módulo GNSS

En la primera implementación del código, se utilizó únicamente el módulo GNSS conectado a la placa de desarrollo Heltec WiFi LoRa 32 V3. El objetivo fue obtener datos de ubicación y verificar que el sistema pudiera capturar correctamente las coordenadas en tiempo real. Durante esta prueba, se evaluó la precisión de las coordenadas obtenidas, asegurando que fueran consistentes y confiables. En la figura 5.2, se puede observar como se obtuvo las coordenadas por monitor serial de la ubicación estática.

Figura 5.2: Pruebas del módulo GNSS.

Luego, se tomaron las coordenadas obtenidas del módulo GNSS y se verificó su precisión utilizando Google Maps [?]. Este paso permitió corroborar la ubicación detectada por el sistema y evaluar la exactitud de la geolocalización en comparación con un dispositivo de referencia. Para ello, se utilizó un teléfono convencional de la marca Xiaomi con modelo Mi 11 Lite, mediante el cual se consultó la ubicación actual del lugar. Este proceso brindó un punto de comparación en cuanto a la precisión del módulo GNSS frente al teléfono, permitiendo analizar posibles diferencias en la exactitud de las coordenadas reportadas por ambos dispositivos. En las figuras 5.3 y 5.4, se muestran, respectivamente, las ubicaciones obtenidas con el módulo GNSS y con el teléfono, facilitando así la visualización comparativa de la precisión entre ambas lecturas.

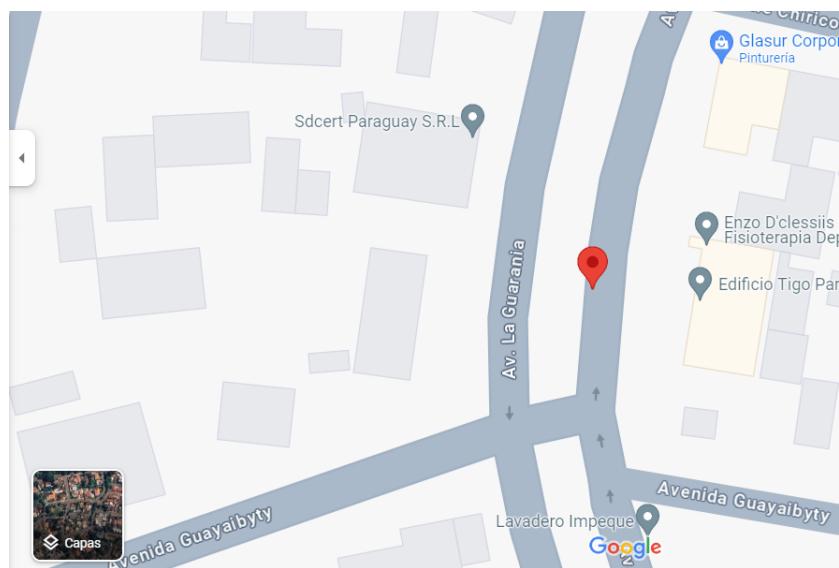


Figura 5.3: Comprobación de las coordenadas del módulo GNSS.

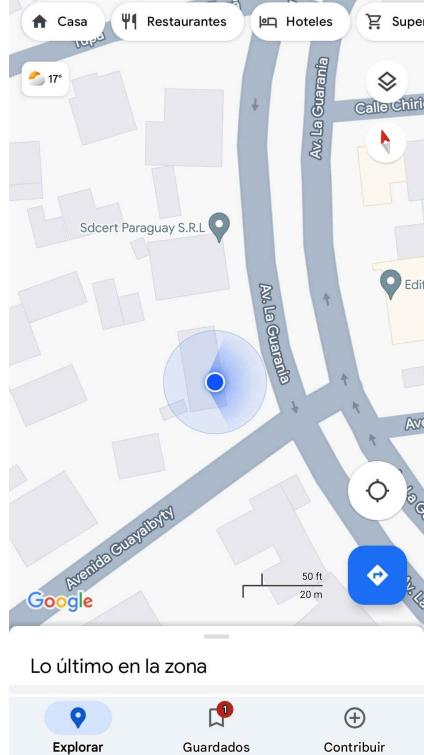


Figura 5.4: Comprobación con teléfono convencional.

A partir de la comparación de las ubicaciones obtenidas, se pudo observar una diferencia de aproximadamente 33 metros entre las coordenadas reportadas por el módulo GNSS y el teléfono Xiaomi Mi 11 Lite. Esta variación es una medida importante que permite evaluar la precisión del módulo GNSS en relación con un dispositivo de uso cotidiano, como el teléfono, que emplea servicios de localización basados en múltiples satélites y redes de datos.

Es importante destacar que estas pruebas iniciales se realizaron dentro de un recinto cerrado, lo cual pudo haber limitado la precisión del módulo GNSS debido a interferencias o bloqueos de señal. Posteriormente, se realizaron pruebas en un espacio abierto, donde la precisión del módulo mejoró notablemente. La exactitud rondó entre los 4 y 6 metros en promedio, con un error máximo de 8 metros, como se observa en la Figura 5.5, que muestra la distancia medida en Google Earth [?], para dos puntos obtenidos.



Figura 5.5: Medición de distancias con Google Earth - Coordenadas específicas.

Para validar estas distancias, se utilizó la herramienta Omni Calculator [?], con la cual se calculó la separación exacta entre las coordenadas obtenidas. El resultado de esta validación se detalla en la Figura 5.6, que confirma una diferencia de 4.9 metros entre las posiciones reportadas.

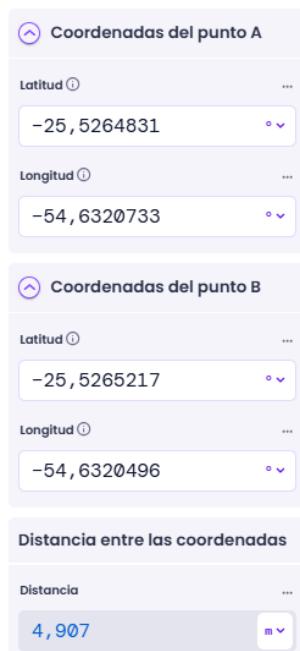


Figura 5.6: Distancia calculada con Omni Calculator.

En la Figura 5.7 se ilustran varias ubicaciones adquiridas durante las pruebas en un espacio abierto, demostrando la consistencia de las coordenadas registradas por el módulo GNSS. Estas mediciones fueron esenciales para comprobar la precisión del módulo .



Figura 5.7: Pruebas de ubicación en Google Earth con múltiples puntos adquiridos.

Prueba del módulo LoRa

En la figura 5.8, se puede observar como se establecio la conexión exitosamente, se pudo observar en la consola como se realizaban las peticiones y los envíos uplink. La consola mostró en tiempo real los datos transmitidos desde el dispositivo, confirmando que los mensajes uplink eran recibidos y procesados correctamente por el servidor. En la segunda implementación, se centró exclusivamente en la funcionalidad de LoRaWAN. Se verificó si el sistema podía enviar correctamente solicitudes de conexión a la red y transmitir mensajes. Esta prueba fue crucial para asegurarse de que el módulo LoRa estuviera configurado correctamente y pudiera establecer comunicación con la red Helium [?], garantizando así que el sistema estuviera listo para la transmisión de datos.

Event	Type	No. of Hotspots	Time
+	Uplink ↗ 0	1	Aug 6, 2024 11:05:43.716 PM
+	Join Accept	1	Aug 6, 2024 11:05:37.635 PM
+	Join Request	1	Aug 6, 2024 11:05:35.635 PM

Figura 5.8: Visualización en la consola helium, mensaje de conexión y subida.

La consola también permitió visualizar a qué hotspot se conectó el dispositivo y con cuál realizó el envío de mensajes. Como se muestra en la figura 5.9.

Hotspots (1)					
Hotspot Name	RSSI	SNR	Frequency	Spreading	Time
feisty-bronze-crane	-116	-6.00	917.00	SF11BW125	Aug 6, 2024 11:05:43.716 PM

Figura 5.9: Visualización en la consola helium.

Con esta información, se utilizó el mapa de Helium [?], para identificar la ubicación exacta del hotspot, lo que permitió estimar la distancia desde la cual el dispositivo podía conectarse. Esto facilitó la evaluación de la cobertura y el alcance de la red en el área de prueba, como se puede observa en las siguientes figuras 5.10 y 5.11.

<input type="checkbox"/>	Hotspot Name	Alias	Signal	Packets	# of Devices	Status
<input type="checkbox"/>	Unknown Hotspot ⓘ			14	1	
<input type="checkbox"/>	Faint Green Hippo			2	1	
<input type="checkbox"/>	Feisty Bronze Crane			1	1	

Figura 5.10: Visualización de cobertura en la consola helium.

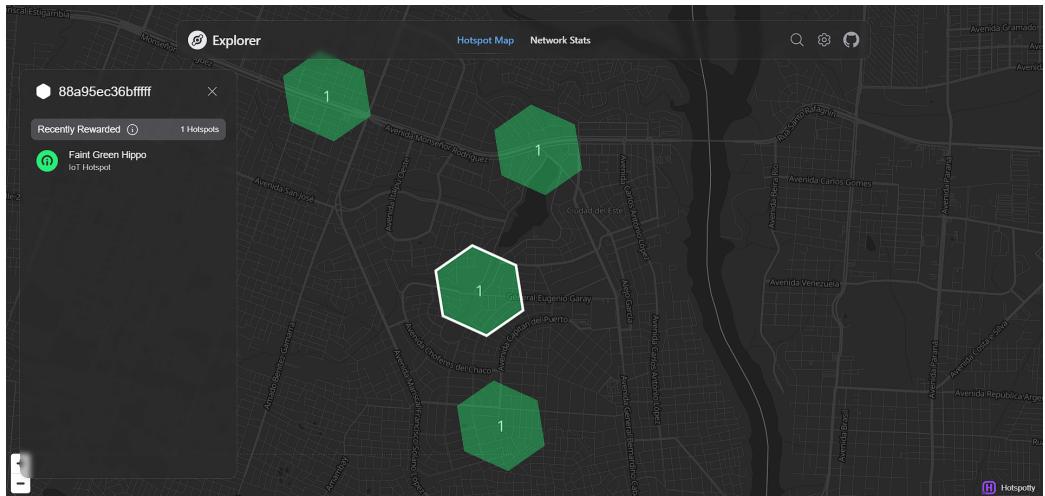


Figura 5.11: Visualización en la página helium maps.

Esta información adicional facilitó la verificación del funcionamiento de la red y permitió asegurar que la comunicación entre el dispositivo y el hotspot se estaba realizando de manera efectiva.

Prueba de captura y transmisión de datos de localización

En la tercera implementación, se combinó la obtención de datos de localización del módulo GNSS con el envío de estos datos a través de LoRaWAN. Durante esta prueba, se verificó la correcta transmisión de los payloads que contenían las coordenadas GNSS, asegurando que no hubiera pérdida de datos en el proceso. Además, se realizó una decodificación manual del payload para validar su contenido. Se analizó la recepción de los datos en el servidor, confirmando que las coordenadas se enviaran y registraran sin errores. En la Figura 5.12, se muestra la estructura del archivo JSON que se descargó de la consola Helium, donde se registran todas las acciones realizadas.

```

"hold_time": 0,
"hotspot": {
    "channel": 3,
    "frequency": 917.4,
    "id": "11cde3uWPkobSSjXNPsNzpkZbpLBRdP64eFkWb8yrhpeJh74ZUh",
    "lat": -25.5341156144535,
    "long": -54.63307668405634,
    "name": "beautiful-cinnamon-cobra",
    "rssi": -130,
    "snr": -12.199999809265137,
    "spreading": "SF10BW125"
},
"mac": [],
"payload": "gDTMwYaIWsI=",
"payload_size": 8,
"port": 2,
"raw_packet": "QC4AAEgAAAACj31oyNl8/IbPy0H6"
},
"description": "Unconfirmed data up received",

```



Figura 5.12: Estructura del Mensaje enviado, donde payload serían las coordenadas en Base64.

Para validar la precisión de las coordenadas GNSS transmitidas, se siguieron los siguientes pasos para decodificar el mensaje recibido:

Por ejemplo, el mensaje gDTMwYaIWsI= se decodifica a: Latitud: -25.5256, Longitud: -54.6333. Para la decodificación Base 64, se utilizó una herramienta en línea, como Base64 Decode [?], para decodificar el mensaje. Este proceso convierte el mensaje codificado en una serie de bytes en formato hexadecimal, lo que facilita su interpretación. Luego se dividieron los primeros 4 bytes del mensaje para obtener la latitud y los siguientes 4 bytes para la longitud. Para convertir estos bytes en números flotantes (float), se utilizó una calculadora en línea, como Hex Convert [?]. Esta herramienta permite convertir el formato hexadecimal en un valor numérico que puede ser interpretado como coordenadas.

En las Figuras 5.13 y 5.14, se muestra el proceso de conversión utilizando la herramienta mencionada, ilustrando cómo se obtuvieron las coordenadas a partir del mensaje recibido.

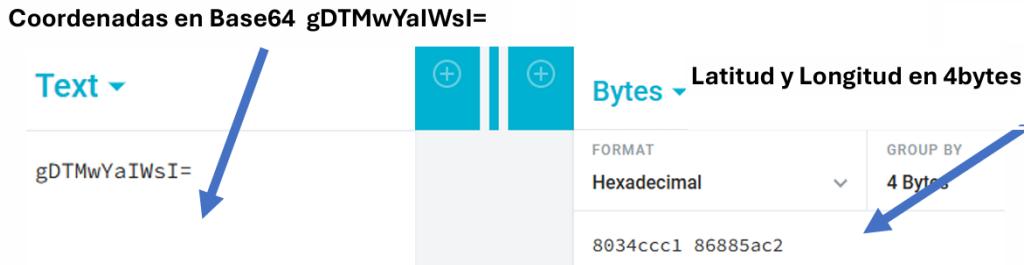


Figura 5.13: Interpretación de Base64 a Hexadecimal.

Float - Little Endian (DCBA)		
#	Raw	Float
0	C1 CC 34 80	-25.5256348

Float - Little Endian (DCBA)		
#	Raw	Float
0	C2 5A 88 86	-54.6333237

Figura 5.14: Interpretación de las coordenadas de latitud y longitud transmitidas.

5.1.3. Pruebas del Módulo de Interfaz Gráfica y Monitoreo

El módulo de Interfaz Gráfica y Monitoreo se configuró para recibir y procesar datos en tiempo real de la ubicación y estado de un dispositivo prototipo. Durante las pruebas, se verificaron las siguientes funcionalidades clave: transmisión de datos, visualización en mapas interactivos, almacenamiento de datos en series temporales, y decodificación de mensajes en formato *Base64*.

Transmisión y visualización de datos

El dispositivo prototipo fue configurado en la consola de *ThingsBoard* para enviar datos de telemetría, tales como latitud, longitud, y eventos de usuario. Los datos se transmitieron a través del protocolo *MQTT* y fueron correctamente registrados en el tablero de control del sistema, donde se visualizan en tiempo real en un mapa *OpenStreet* (ver Figura 5.15).

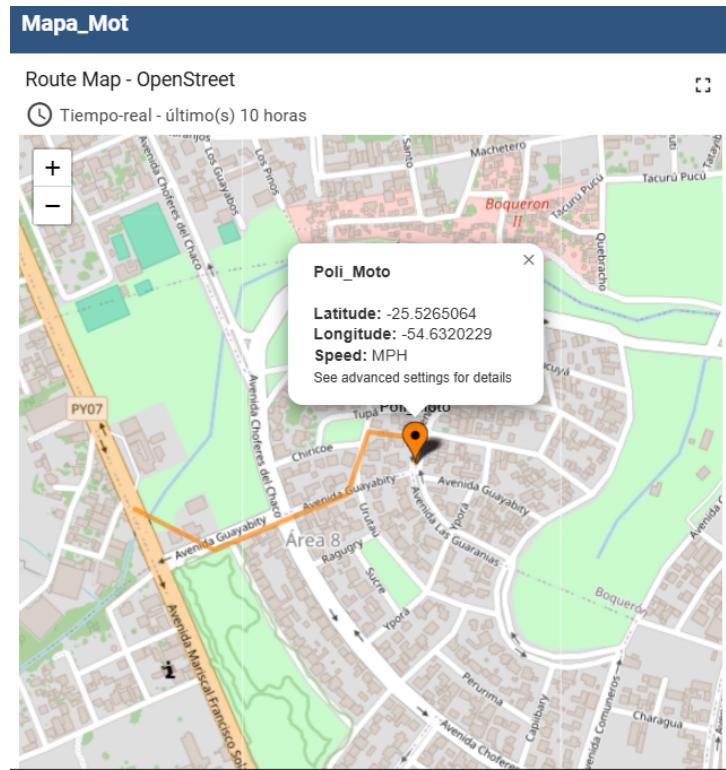


Figura 5.15: Visualización de la ubicación en el mapa *OpenStreet* con trayectoria.

Almacenamiento en tablas de series temporales

La telemetría se almacenó en una tabla de series de tiempo que registró cada evento, permitiendo un análisis detallado de la trayectoria y actividad del dispositivo (ver Figura 5.16). Esto proporcionó un historial de datos que facilita la trazabilidad y auditoría de la ubicación del dispositivo en distintos momentos del día.

Timeseries table			
⌚ Tiempo-real - último(s) día			
Timestamp ↓	latitude	longitude	payload
2024-11-15 17:41:11	-25.5138053894 deg	-54.6414070129 deg	461CCCC1CD905AC2
2024-11-15 17:40:49	-25.5161609650 deg	-54.6415176392 deg	1921CCC1EA905AC2
2024-11-15 17:40:27	-25.5162410736 deg	-54.6414985657 deg	4321CCC1E5905AC2
2024-11-15 17:40:05	-25.5165843964 deg	-54.6414947510 deg	F721CCC1E4905AC2
2024-11-15 17:39:43	-25.5175304413 deg	-54.6413383484 deg	E723CCC1BB905AC2
2024-11-15 17:39:22	-25.5176429749 deg	-54.6412849426 deg	2224CCC1AD905AC2
2024-11-15 17:39:00	-25.5181102753 deg	-54.6411170959 deg	1725CCC181905AC2
2024-11-15 17:38:40	-25.5189476013 deg	-54.6407203674 deg	CE26CCC119905AC2
2024-11-15 17:37:58	-25.5224742889 deg	-54.6390914917 deg	072ECCC16E8E5AC2
2024-11-15 17:37:36	-25.5225334167 deg	-54.6390609741 deg	262ECCC1668E5AC2
2024-11-15 17:37:14	-25.5225753784 deg	-54.6390495300 deg	3C2ECCC1638E5AC2

Powered by [Thingsboard v.3.8.1](#)

Figura 5.16: Series temporales de telemetría.

Pruebas de decodificación y transformación de datos

Para evaluar la correcta interpretación de mensajes enviados al sistema, se llevaron a cabo una serie de pruebas utilizando dos scripts desarrollados en *JavaScript*. Estas pruebas se diseñaron para asegurar la decodificación precisa de los datos transmitidos en formato *Base64* y su posterior interpretación en valores útiles para el sistema.

El proceso se dividió en dos etapas principales de prueba. En la primera etapa, se probó un script encargado de transformar los datos enviados en formato *Base64* a valores hexadecimales (ver Figura 5.17). Este script se verificó con mensajes simulados de distintos tamaños y contenido. Se realizaron 5 pruebas, cada una con un mensaje diferente, asegurando que los valores obtenidos fueran consistentes con los datos esperados.

En la segunda etapa, se utilizó un script adicional que interpretaba los valores hexadecimales obtenidos en la etapa previa. Este script clasificaba los datos según el tamaño del mensaje decodificado:

- Mensajes de 2 *bytes* representaban eventos, como alertas de desconexión o intentos de acceso no autorizado.

- Mensajes de 4 *bytes* representaban eventos, como de acceso de usuarios autorizados.
- Mensajes de 8 *bytes* correspondían a telemetrías de ubicación, como coordenadas de latitud y longitud.

Durante esta etapa, se realizaron 10 pruebas para verificar la capacidad del sistema de manejar una variedad de tamaños y contenidos diferentes en los mensajes. Los resultados de todas las pruebas fueron exitosos, demostrando es capaz de interpretar y clasificar correctamente los datos enviados. Un ejemplo de este proceso puede observarse en la Figura 5.18, donde se muestra la interpretación de un mensaje hexadecimal como denegación de acceso no autorizado.

The screenshot shows a software interface for message decoding. On the left, under the 'Mensaje' tab, there is a code editor with the following content:

```

1 <?
2   "payload": "/wE"
3 >

```

A red arrow points from the text "Mensaje de entrada en Base64" to the line "payload: "/wE"".

On the right, under the 'Salida' tab, there is another code editor with the following content:

```

1 <?
2   "msg": {
3     "payload": "FF01"
4   },
5   "metadata": {},
6   "msgType": "POST_TELEMETRY_REQUEST"
7 >

```

A red circle highlights the entire "msg" object. A red arrow points from the text "Mensaje de salida en Hexadecimal" to the line "msgType: 'POST_TELEMETRY_REQUEST'".

Figura 5.17: Prueba de decodificación de mensajes en formato *Base64* a valores hexadecimales.

```

Mensaje
1 ~ {
2   "payload": "FF0001"
3 }

function Transform(msg, metadata, msgType) {
1 // Función para convertir hexadecimal a float usando little-endian
2   function hexToFloat(hex) {
3     if (hex.length !== 8) return null; // Validación para 8
4       caracteres
5
6     var buffer = new ArrayBuffer(4); // Crear un buffer de 4 bytes
7     var view = new DataView(buffer);
8
9     for (var i = 0; i < 4; i++) {
10       view.setUint8(i, parseInt(hex.substr(i * 2, 2), 16));
11
12     return view.getFloat32(0, true); // true para little-endian
13   }
}

Salida
1 ~ {
2   "msg": {
3     "payload": "FF0001",
4     "alarm": "No autorizado"
5   },
6   "metadata": {},
7   "msgType": "POST_TELEMETRY_REQUEST"
8 }


```

Mensaje de entrada en Hexadecimal

Mensaje de salida interpretado

Figura 5.18: Prueba de interpretación de valores hexadecimales según su tamaño.

Para garantizar la precisión de las pruebas, se utilizó *Mosquitto* como herramienta para enviar mensajes simulados al sistema (ver Figura 5.19). Esta herramienta permitió verificar el flujo completo de transmisión.

Todos los resultados confirmaron el correcto funcionamiento del sistema, logrando interpretar los datos en tiempo real de manera eficiente y confiable.

```

mario@DESKTOP-J58FDBR:/mnt/c/Users/Hp$ mosquitto_pub -d -q 1 -h 172.21.240.1 -p 1883 -t v1/devices/me/telemetry -u "fxau...
gwbqfy4sn3kup4i" -m "{"
  "latitude": -25.5153,
  "longitude": -54.61558
}"
Client (null) sending CONNECT
Client (null) received CONNACK (0)
Client (null) sending PUBLISH (d0, q1, r0, m1, 'v1/devices/me/telemetry', ... (48 bytes))
Client (null) received PUBACK (Mid: 1, RC:0)
Client (null) sending DISCONNECT
mario@DESKTOP-J58FDBR:/mnt/c/Users/Hp$ mosquitto_pub -d -q 1 -h 172.21.240.1 -p 1883 -t v1/devices/me/telemetry -u "fxau...
gwbqfy4sn3kup4i" -m "{"
  "latitude": -25.5155,
  "longitude": -54.61574
}"
Client (null) sending CONNECT
Client (null) received CONNACK (0)
Client (null) sending PUBLISH (d0, q1, r0, m1, 'v1/devices/me/telemetry', ... (48 bytes))
Client (null) received PUBACK (Mid: 1, RC:0)
Client (null) sending DISCONNECT

```

Publicación exitosa

Figura 5.19: Uso de *Mosquitto* para simular mensajes durante las pruebas.

Resultados de la integración entre ThingsBoard y Helium

En la Figura 5.20 se observa el resultado del proceso de integración entre ThingsBoard y la red Helium mediante el protocolo *MQTT*. Este proceso consistió en la recepción y procesamiento de cada mensaje enviado desde el dispositivo conectado.

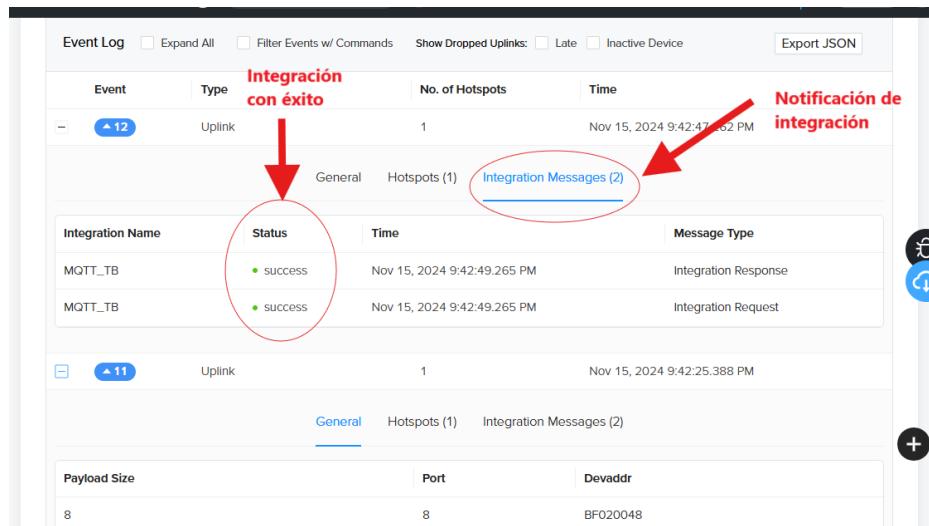


Figura 5.20: Integración Helium-ThingsBoards.

De los más de 120 mensajes transmitidos, un total de 8 mensajes fallaron al momento de integrarse, lo que representa un porcentaje de efectividad del 93.33 % en la integración.

A pesar de estas fallas, los mensajes que sí lograron ser transmitidos exitosamente fueron interpretados y decodificados con un 100 % de efectividad. Esto incluye tanto la transformación de los datos de formato *Base64* a hexadecimales como su interpretación en valores útiles para el sistema, validando el correcto funcionamiento del módulo.

5.1.4. Integración del sistema completo

En esta fase, se realizaron pruebas específicas para validar la integración y el correcto funcionamiento de todos los módulos necesarios para el sistema: RFID, GNSS y LoRaWAN, utilizando las placas de desarrollo ESP-WROOM-32 y Heltec WiFi LoRa 32 V3. El objetivo principal fue asegurar la operación conjunta de estos módulos en el sistema, comprobando la funcionalidad de identificación mediante RFID, la transmisión de datos de ubicación

y el envío de información por la red LoRaWAN. Las pruebas se desarrollaron en tres etapas, detalladas a continuación:

Primera Prueba: Comunicación entre las placas de desarrollo (ESP-WROOM-32 y Heltec WiFi LoRa 32 V3)

Esta prueba inicial consistió en establecer la comunicación entre ambas placas mediante el protocolo UART, con el módulo RFID conectado al ESP-WROOM-32. El sistema transmitió datos de identificación desde el módulo RFID al Heltec WiFi LoRa 32 V3, validando la presencia del UID autorizado. Además, se comprobó que esta transmisión no interrumpiera la conexión con la red LoRaWAN, manteniendo la integridad de los datos y la estabilidad de la comunicación entre los dispositivos.

Segunda Prueba: Integración del Módulo GNSS

En esta etapa, se implementó la versión 2 del código, que integraba el módulo GNSS junto con el módulo RFID. La prueba se orientó a evaluar la transmisión de los datos recolectados, donde:

El UID autorizado fue enviado en formato hexadecimal. Otros datos de ejemplo fueron incluidos en la transmisión para verificar la estructura de los mensajes. Las coordenadas GPS fueron convertidas a un formato adecuado, preparadas como un arreglo de bytes para ser enviadas por LoRaWAN.

Tercera Prueba: Evaluación de las Funciones de Envío de Datos Completos

En la última fase, se probó la versión 3 del código, que incluyó el flujo completo de detección RFID, obtención de coordenadas GNSS y envío de datos a través de LoRaWAN. En esta prueba se evaluaron las siguientes condiciones:

- Detección RFID: El sistema identificó la presencia de un tag RFID autorizado, enviando el UID a través de LoRaWAN en el caso de una identificación válida.
- Envío de Datos por LoRaWAN: Dependiendo del tipo de mensaje seleccionado, el sistema envió el UID del RFID, otros datos de prueba o, en caso de estar disponibles, las coordenadas.
- Ciclo de LoRaWAN: El dispositivo siguió un ciclo de transmisión periódica, entrando en modo de suspensión tras cada envío, optimizando así el consumo energético.

- Validación de Coordenadas: Se verificó que el dispositivo obtuviera coordenadas válidas y las transmitiera en el siguiente ciclo de envío, asegurando que no hubiera pérdida de datos en la transmisión.

Los resultados de las pruebas realizadas para las tres versiones del código se resumen en la Tabla 5.2.

Tabla 5.2: Resultados de las Pruebas por Versión del Código

Versión del Código	Total de Pruebas	Pru
Primera Versión	10	
Segunda Versión	20	
Tercera Versión	25	

A lo largo de las pruebas realizadas, la evolución del sistema mostró mejoras significativas. En la primera versión, se obtuvieron un 80 % de éxito y un 20 % de fallos en 10 pruebas. La segunda versión alcanzó un 75 % de éxito y un 25 % de fallos en 20 pruebas. Finalmente, la tercera versión logró una notable eficiencia con un 96 % de éxito y solo un 4 % de fallos en 25 pruebas.

En la Figura 5.21, se presenta la correcta recepción de los datos de UID, transmitidos como payloads de 4 bytes. En la Figura 5.22, se muestra cómo se recibieron los datos de las coordenadas en los intervalos de tiempo definidos.

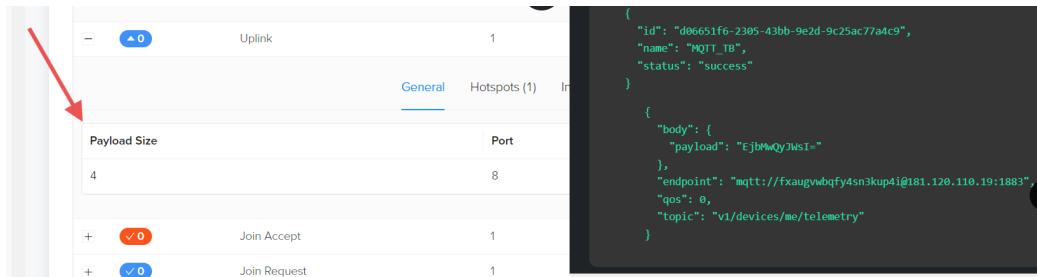


Figura 5.21: Visualización en la consola Helium de los paquetes de datos recibidos del UID.

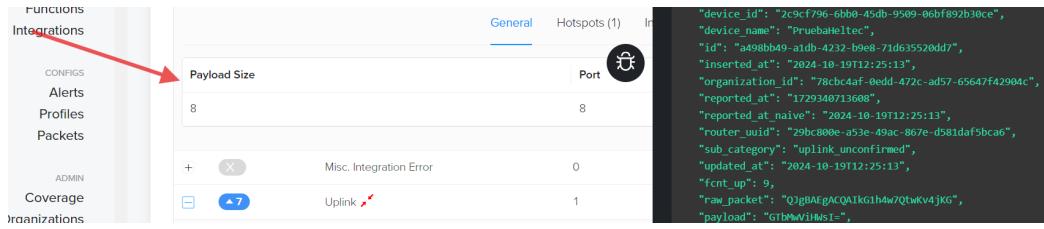


Figura 5.22: Visualización en la consola Helium de los paquetes de datos recibidos de las coordenadas.

5.2. Pruebas de Campo

En esta sección, se documentan las pruebas de campo realizadas con una motocicleta Kenton, modelo GL150, equipada con un sistema de encendido *CDI* (Ignición por Descarga de Condensador) y una batería de 12 voltios. Estas pruebas se llevaron a cabo utilizando la última versión del código ajustado, que integra la funcionalidad de corte de energía y arranque controlado mediante el sistema *RFID* y el módulo relé. Además, se evaluó el rendimiento del sistema en movimiento.

5.2.1. Evaluación Inicial

El propósito de las pruebas iniciales fue verificar la integración del sistema y su capacidad para realizar autenticaciones fluidas y sincronizarse adecuadamente con el módulo de corte de energía. Se comprobó que el sistema respondiera de manera precisa a la presencia de un *tag* autorizado, ejecutando el corte de energía al retirarse o ausentarse dicho *tag*. Además, se llevaron a cabo varias pruebas para medir la respuesta del sistema y garantizar su fiabilidad.

En la Figura 5.23, se presenta el diagrama esquemático de las conexiones realizadas para la integración del sistema desarrollado en la motocicleta. Este esquema representa la interacción entre los componentes eléctricos de la motocicleta y el prototipo diseñado.

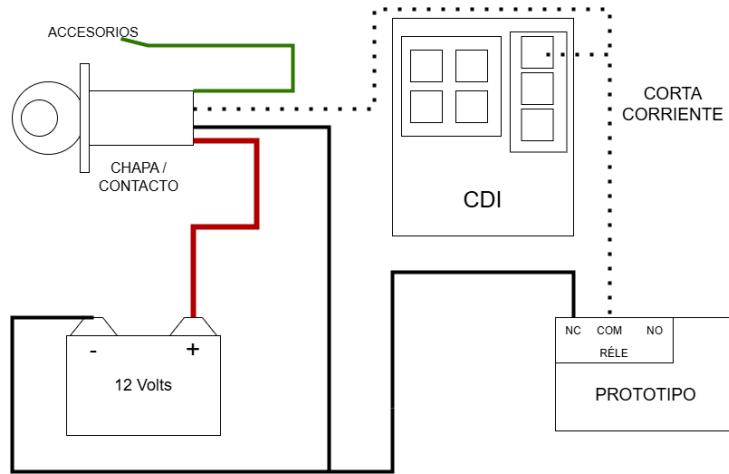


Figura 5.23: Diagrama esquemático de conexiones.

5.2.2. Pruebas en Movimiento

Posteriormente, se realizaron 9 pruebas de campo adicionales para evaluar el comportamiento del sistema en condiciones reales de movimiento. En estas pruebas, se evaluaron las siguientes fases:

- **Conexión a *LoRaWAN* en movimiento:** En tres pruebas específicas, se inició el sistema fuera del área de cobertura de la red *LoRaWAN*, y se comprobó que, al entrar en el rango de la red, el dispositivo lograba realizar el proceso de *join* exitosamente. Todas estas pruebas fueron satisfactorias.
- **Autenticación y arranque del motor:** En cada prueba, se aproximó el *tag RFID* para autenticar al usuario. Aunque en dos pruebas iniciales la lectura del *UID* falló, provocando el apagado inmediato del motor, las pruebas posteriores fueron exitosas, logrando una autenticación fluida y un arranque sin interrupciones.
- **Estabilidad del sistema durante el movimiento:** Durante todas las pruebas, el sistema permaneció operativo, sin desconexiones ni interrupciones, incluso al enfrentarse a cambios de velocidad.
- **Transmisión de datos en tiempo real:** El sistema transmitió correctamente las coordenadas *GPS* y los eventos críticos (autenticación, desconexión, etc.) a la plataforma *ThingsBoard*, donde se visualizaron en tiempo real.

En la Figura 5.24, se ilustra una de las trayectorias recogidas durante las pruebas de movimiento, mostrando cómo el sistema registró y transmitió la ubicación de la motocicleta en tiempo real.

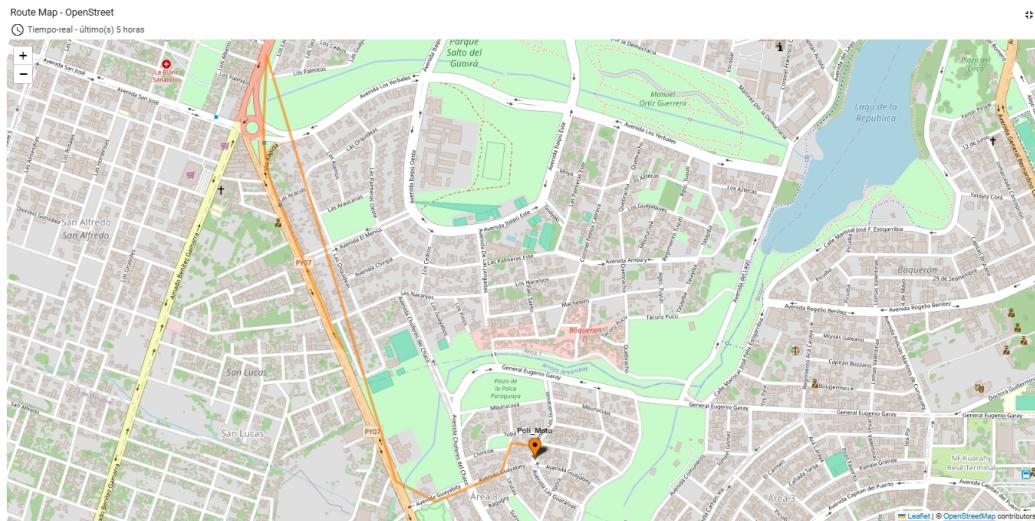


Figura 5.24: Trayectoria registrada durante las pruebas de campo.

5.2.3. Resultados Cuantitativos

En la Tabla 5.3, se resumen los resultados de las pruebas realizadas, incluyendo autenticaciones exitosas, conexiones a *LoRaWAN* y transmisión de datos durante el movimiento.

Tabla 5.3: Resultados de las pruebas de campo

Nº de Prueba	Autenticación exitosa	Conexión a LoRaWAN
1	No	Si
2	No	Si
3	Sí	Si
4	Sí	Si
5	Sí	Si
6	Sí	Si
7	Sí	Si
8	Sí	Si
9	Sí	Si

Porcentajes de Éxito y Fallo

- **Autenticación exitosa:** De las 9 pruebas realizadas, 7 lograron autenticar correctamente al usuario mediante el *tag RFID*, lo que representa un 77.78 % de éxito. Las dos fallas iniciales, que corresponden al 22.22 % de fallo, se debieron a errores humanos durante el proceso de manejo del sistema, como una posición incorrecta del tag RFID o fallos en su aproximación al lector. Estas situaciones fueron identificadas y corregidas en las pruebas posteriores, logrando resultados exitosos en el resto de las evaluaciones.
- **Conexión a *LoRaWAN*:** En el 100 % de las pruebas, el dispositivo logró conectarse exitosamente a la red *LoRaWAN*, incluso en condiciones de movimiento y al iniciar fuera del rango de cobertura. Esto demuestra una fiabilidad completa (100 % de éxito) en la capacidad de conexión del sistema.
- **Transmisión de datos:** En todas las pruebas, los datos del sistema, como coordenadas *GPS* y eventos críticos, fueron transmitidos correctamente a la plataforma *ThingsBoard*. Esto refleja un 100 % de éxito en la transmisión de datos.
- **Sistema estable:** Durante todas las pruebas, el sistema permaneció operativo, sin apagarse ni desconectarse de manera inesperada. Esto indica una estabilidad del 100 %.

- **Corte de energía al alejar la tarjeta:** En el 100% de los casos, el sistema ejecutó correctamente el corte de energía al detectar la ausencia del *tag* autorizado. Este resultado asegura una respuesta fiable del sistema en situaciones de seguridad.

5.2.4. Interpretación de los Resultados

Los resultados generales demuestran un desempeño sólido del sistema en las fases probadas. A pesar de las fallas iniciales en el proceso de autenticación, el prototipo mostró un comportamiento consistente y confiable tras los ajustes realizados.

- **Éxito global:** Considerando todos los criterios evaluados, el sistema presentó un éxito promedio de 95.56 % (calculado como el promedio ponderado de los porcentajes de éxito en cada criterio).
- **Fallo global:** Las fallas observadas se limitaron al proceso de autenticación inicial y representaron un promedio de 4.44 % del total de las pruebas, indicando que los problemas fueron corregidos de manera efectiva.

Este análisis cuantitativo confirma que el prototipo cumple con los requerimientos definidos para garantizar la autenticación de usuarios, la conectividad a la red y la transmisión de datos en tiempo real. Además, su estabilidad y capacidad de respuesta en condiciones reales refuerzan su viabilidad como solución para la seguridad vehicular.

5.3. Prototipo del sistema desarrollado

En este apartado se describen las principales características del prototipo desarrollado, el cual integra *hardware* y diseño estructural. El prototipo se divide en dos partes principales: el *hardware* del sistema y el *software* del sistema.

5.3.1. Hardware del prototipo

El *hardware* del prototipo está compuesto por dos módulos principales. El módulo de autenticación y control identifica un *UID* autorizado mediante un lector *RFID* y realiza una lectura constante para verificar si el *UID* permanece dentro del rango establecido. En caso de que el *UID* autorizado

salga del rango, el sistema corta automáticamente la energía de la motocicleta, impidiendo su funcionamiento. Adicionalmente, el sistema incorpora un módulo de captura y transmisión de datos, diseñado para transmitir información mediante tecnología *LoRaWAN*. Incluye el envío del *UID* autorizado y las coordenadas de ubicación de la motocicleta, además de enviar mensajes de desconexión del usuario.

Modelado y Diseño de Encapsulado 3D

A continuación, en la Figura 5.25, se presenta el diseño de la disposición estructural de los componentes, donde se han colocado los cables de conexión por debajo del *shield*, optimizando así el orden y la accesibilidad de cada módulo en el sistema.

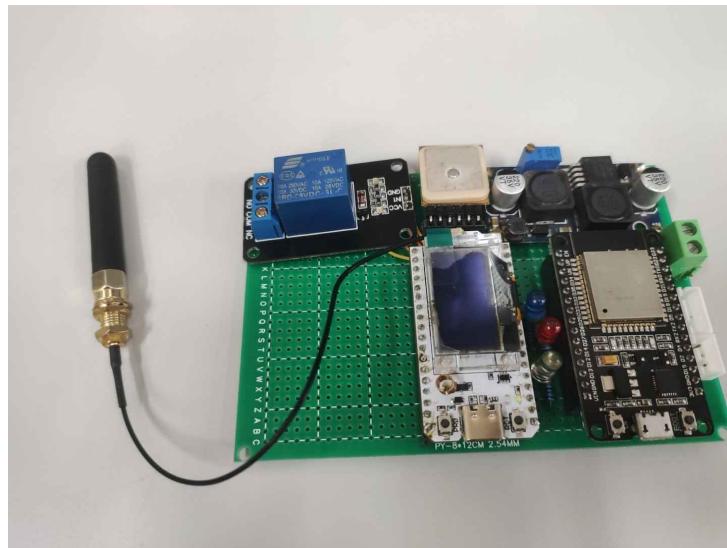


Figura 5.25: Disposición estructural.

En la siguiente Figura 5.26, se presenta el encapsulado en *3D*, elaborado mediante impresión en una impresora *3D*, el cual proporciona protección y organización a los componentes del sistema. En la Figura 5.27, se muestra un encapsulado adicional para el módulo *RFID*.



Figura 5.26: Encapsulado 3D.

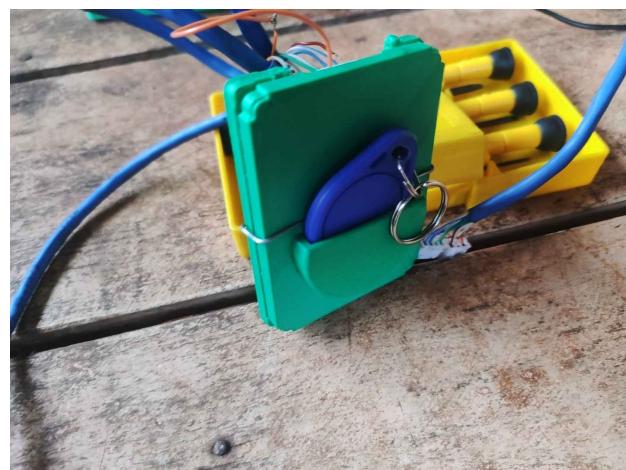


Figura 5.27: Encapsulado 3D para el módulo *RFID*.

Por último, en la Figura 5.28, se presenta el montaje final del sistema, con todos los componentes integrados y dispuestos en la motocicleta. La Figura 5.28 incluye flechas que destacan la ubicación precisa de los prototipos, facilitando su identificación y visualización en el contexto del montaje completo.



Figura 5.28: Prototipo Final.

Software del prototipo

El *software* desarrollado para el prototipo habilita las funciones principales del sistema, asegurando la integración de los módulos de *hardware* con un control eficiente y coordinado. Este *software* gestiona la interacción entre los componentes, facilitando la autenticación de usuarios, el monitoreo continuo y la transmisión de datos mediante *LoRaWAN*.

Entre sus funcionalidades más relevantes, se encuentra la validación de usuarios mediante el lector *RFID*, supervisando de forma constante la presencia del *UID* autorizado. En caso de desconexión, el *software* ejecuta automáticamente acciones críticas, como el corte de energía de la motocicleta, reforzando la seguridad del sistema.

Adicionalmente, el *software* es responsable de la recopilación y transmisión de datos relevantes, como las coordenadas *GPS* y el *UID* autenticado, a través de *LoRaWAN*. Esto asegura que el usuario pueda recibir alertas en tiempo real, incluyendo notificaciones de eventos críticos como intentos de arranque no autorizados.

Una de las características más destacadas es la capacidad del *software* para integrar y mostrar estos datos en una plataforma gráfica basada en

ThingsBoard. En esta interfaz, el usuario puede visualizar la ubicación en tiempo real de la motocicleta sobre un mapa interactivo, además de consultar una tabla de series temporales que detalla las coordenadas transmitidas y otros parámetros relevantes.

En la Figura 5.29 se presenta un ejemplo de la visualización en la plataforma, destacando cómo el *software* permite un monitoreo continuo y detallado. La figura muestra la ubicación de la motocicleta y los datos transmitidos mediante *LoRaWAN*.

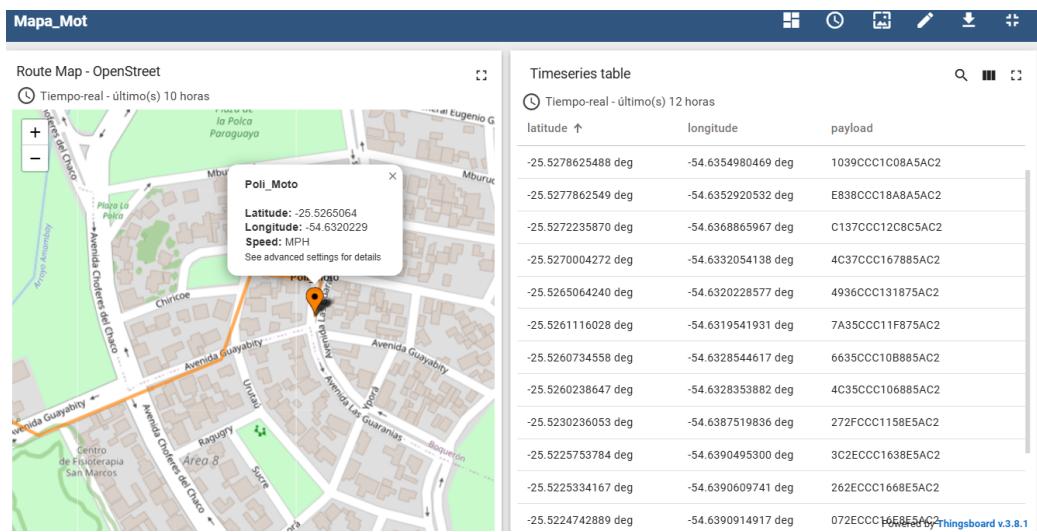


Figura 5.29: Visualización de datos en tiempo real.

Con estas funcionalidades, el *software* habilita una solución tecnológica que combina seguridad, monitoreo y gestión de datos en tiempo real, respondiendo a las necesidades específicas del prototipo desarrollado.

Capítulo 6

Conclusión

En este capítulo se interpretan y sintetizan los resultados expuestos previamente, además de evaluar la validez de la hipótesis, el logro de los objetivos y realizar recomendaciones para futuros estudios.

6.1. Discusión

Se ha desarrollado un sistema de seguridad para motocicletas basado en tecnologías de autenticación *RFID* y transmisión de datos mediante *LoRaWAN*. Este sistema, tras varias etapas de desarrollo e investigación en el ámbito *IoT*, ha sido capaz de autenticar usuarios y rastrear la ubicación de la motocicleta en tiempo real.

Durante el proceso de desarrollo, se lograron integrar exitosamente los módulos de autenticación, comunicación y control, permitiendo una funcionalidad de seguridad que restringe el uso de la motocicleta solo a usuarios autenticados y proporciona un monitoreo continuo en caso de situaciones de riesgo.

Además, se tuvieron en cuenta aspectos como la alimentación del sistema en entornos de prueba y la integración de sensores que permitieran tanto la autenticación mediante *RFID* como la recolección de datos de ubicación a través del módulo *GNSS*. Esto aseguró que el prototipo respondiera adecuadamente en condiciones reales de uso en motocicletas.

Las pruebas realizadas arrojaron resultados favorables, evidenciando la efectividad del sistema en el control de acceso y el rastreo de ubicación. Comparado con métodos de seguridad vehicular que solo emplean bloqueos mecánicos o electrónicos, este sistema proporciona una capa adicional de seguridad, permitiendo la autenticación continua del usuario y la desconexión

automática en caso de que el usuario se aleje del vehículo. Esta solución, probada en entornos urbanos simulados, ha mostrado un grado de efectividad del 95.56 % en las situaciones de prueba.

En comparación con otros estudios y tecnologías de seguridad, como los sistemas de rastreo *GPS* convencionales o las aplicaciones de bloqueo remoto, nuestro sistema destaca por su integración de una red de baja potencia (*LoRaWAN*) que permite un monitoreo constante sin afectar significativamente la autonomía de la batería del vehículo. De este modo, el sistema propuesto ofrece un enfoque novedoso y efectivo en términos de seguridad y rastreo.

6.2. Análisis de la hipótesis

A partir de los resultados obtenidos, la hipótesis planteada inicialmente, que sugiere que la integración de tecnologías *RFID* y *LoRaWAN* para la seguridad y rastreo de motocicletas puede ofrecer una alternativa eficaz a los sistemas actuales, ha sido validada. Los datos recopilados durante las pruebas en entornos controlados y en campo confirman que el sistema desarrollado proporciona una capa adicional de seguridad y monitoreo en tiempo real.

6.3. Principales logros alcanzados

En consecuencia al cumplimiento de los objetivos también presentados en el primer capítulo, han sido alcanzados los siguientes logros:

- Se definió una estructura para el sistema de seguridad, integrando tecnologías de comunicación y control.
- Se desarrolló un *firmware* eficiente para el microcontrolador que permite la interacción fluida entre los dispositivos de seguridad y comunicación.
- Se adaptó y aprovechó el uso de *ThingsBoard* como interfaz de usuario, permitiendo la visualización clara de alertas y datos en tiempo real.
- Se logró ensamblar un prototipo de *hardware* que integra sensores, actuadores y un sistema de alimentación adecuado, garantizando su funcionalidad en el entorno de prueba.
- Se logró diseñar y fabricar un encapsulado *3D* personalizado que asegura la protección eficaz de los componentes del sistema.

- Se llevaron a cabo pruebas de laboratorio y de campo que confirmaron el rendimiento esperado del sistema, validando su efectividad en escenarios reales y obteniendo datos clave para futuros desarrollos.

6.4. Sugerencias para futuras investigaciones

Con base en este trabajo, se sugieren las siguientes áreas de mejora y futuras investigaciones:

- Optimizar el sistema de energía mediante algoritmos de bajo consumo para prolongar la vida útil de la batería de respaldo en períodos de inactividad.
- Desarrollar una aplicación móvil que permita monitorear el estado del vehículo en tiempo real y recibir alertas inmediatas.
- Evaluar la cobertura de la red *Helium* en distintos entornos geográficos y analizar su rendimiento en áreas rurales y urbanas para validar la robustez del sistema.

Anexo

Evidencia de los Procesos Realizados.



Figura 6.1: Validación del Sistema Integrado en la Motocicleta.

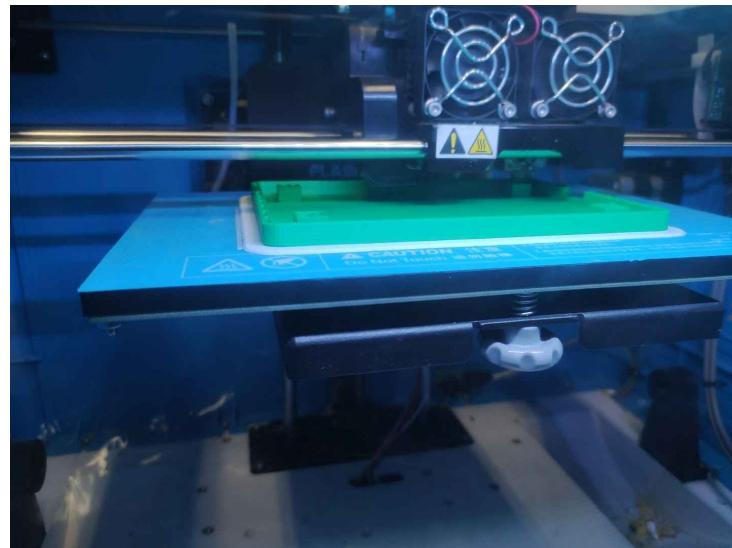


Figura 6.2: Proceso de impresión del encapsulado.

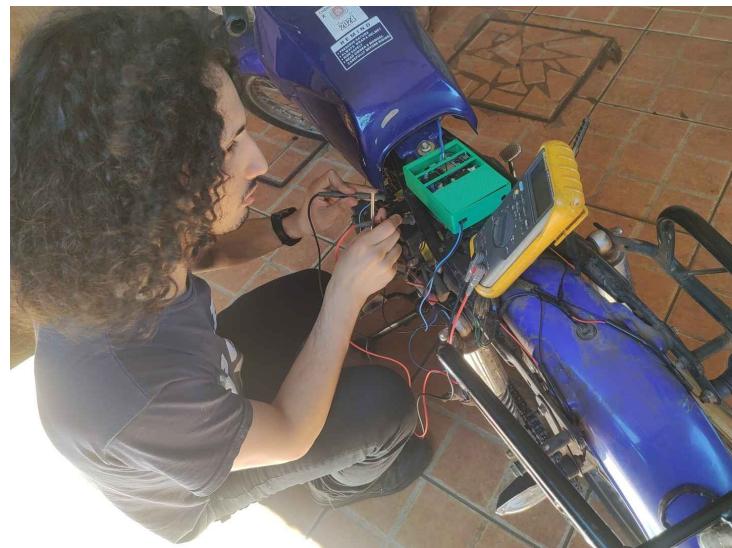


Figura 6.3: Proceso de medición de continuidad en el sistema eléctrico de una motocicleta.

Referencias bibliográficas

- [1] Ikpehai, Augustine, Adebisi, Bamidele, Rabie, K. M., A. Kelvin, Ande, R. E., Hammoudeh, Mohammad, Gacanin, Haris, Mbanaso, y U. M., “Low-power wide area network technologies for internet-of-things: A comparative review,” *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2225–2240, 2019.
- [2] X. Jia, Feng, Quanyuan, Fan, Taihua, Lei, y Quanshui, “Rfid technology and its applications in internet of things (iot),” *2012 2nd International Conference on Consumer Electronics, Communications and Networks, CECNet 2012 - Proceedings*, 04 2012.
- [3] “What is lorawan,” 2020, loRa Alliance. Disponible en: <https://www.thethingsnetwork.org/docs/lorawan/what-is-lorawan>. [Online]. Available: <https://www.thethingsnetwork.org/docs/lorawan/what-is-lorawan/>
- [4] D. de Helium, “Descripción general de la red de helium.” [Online]. Available: <https://docs.helium.com/>
- [5] Dirección del registro de automotores. Estadística de matriculaciones correspondiente al periodo: octubre de 2000 a febrero de 2024. [Online]. Available: <https://www.pj.gov.py/contenido/155-direccion-del-registro-de-automotores/1331>
- [6] Denuncias registradas y aclaradas en las comisarías del país, según tipo de delito. aÑo 2019. Datos.gov.py - Anuario Estadístico 2019. [Online]. Available: <https://www.datos.gov.py/dataset/anuario-estad-stico-2019-estad-sticas-policiales-y-accidentes-de-tr-nsito/resource/c73bf2da>
- [7] A. N. de Tránsito y Seguridad Vial (ANTS). (Año) Informe anual 2021 - observatorio. [Online]. Available: https://www.antsv.gov.py/application/files/8716/6004/9458/INFORME_ANUAL_2021_OBSERVATORIO.pdf

- [8] N. Rana, P. Khatta, y A. D. Mishra, “Smart security system for two-wheelers,” *International Journal of Technical Research & Science (Special Issue)*, 2020. [Online]. Available: <https://doi.org/10.30780/specialissue-ICACCG2020/019>
- [9] M. Sathiyanarayanan, S. Mahendra, y R. B. Vasu, “Smart security system for vehicles using internet of things (iot),” in *2018 Second International Conference on Green Computing and Internet of Things (ICG-CIoT)*, 2018, pp. 430–435.
- [10] A. N. de Tránsito y Seguridad Vial (ANTSV). (Año) Informe anual 2020. [Online]. Available: https://www.antsv.gov.py/application/files/2716/1943/8500/INFORME_ANUAL_2020.pdf
- [11] I. Quesada. (2021) Proyecto control y seguimiento automático de vagones rfid-lorawan. <https://doi.org/10.13140/RG.2.2.36244.01925>.
- [12] A. Cama, E. D. la Hoz, y D. Cama, “Las redes de sensores inalámbricos y el internet de las cosas,” *Revista INGE CUC*, vol. 8, no. 1, pp. 163–172, 2012, artículo que analiza la importancia de las *WSN* en el contexto del IoT, destacando estándares como *IEEE 802.15.4*, *6LoWPAN* y *RPL*. [Online]. Available: <https://www.cuc.edu.co/revistas>
- [13] J. G. R. Berrió y R. A. L. Sánchez, *Internet de las Cosas*. Córdoba, España: Red Educativa Digital Descartes, 2024, libro que ofrece una introducción al IoT y su aplicación en diversos sectores. [Online]. Available: <https://proyectodescartes.org/>
- [14] S. M. González, “El internet de las cosas: Definición y aplicaciones,” *Investiga TEC*, Septiembre 2015, el artículo explora las aplicaciones potenciales del IoT y los desafíos que plantea en términos de privacidad y seguridad.
- [15] J. A. Arévalo, “El internet de las cosas,” *DesiderataLAB*, no. 1, pp. 24–25, 2016, análisis sobre la evolución del IoT y su impacto en la transformación de entornos urbanos e industriales.
- [16] G. Martínez Muñoz, “Red de sensores para localización basada en lora y fwire,” 2019. [Online]. Available: <https://hdl.handle.net/11441/92169>
- [17] D. de semtech, “¿qué son lora® y lorawan®?” [Online]. Available: <https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/>

- [18] Semtech, “Semtech lora technology overview,” 2021, disponible en: <https://semtech.com/lora/lora-technology-overview>.
- [19] P. Bertoletti, R. Paiotti, y R. Leca, *Proyectos con ESP32 y LoRa*. Editora NCB, 2019. [Online]. Available: <https://books.google.es/books?id=Doi0DwAAQBAJ>
- [20] T. T. Network, “Frequency plans — the things network,” 2020, the Things Network. Disponible en: <https://www.thethingsnetwork.org/docs/lorawan/frequency-plans/>.
- [21] L. Alliance, “Rp002-1.0.1 lorawan regional parameters,” 2015, parámetros regionales, especificación RP002, noviembre 2015.
- [22] Semtech, “Lora phy — semtech,” 2021, disponible en: <https://semtech.com/lora/lora-phy>.
- [23] L. Alliance, “About lorawan® - lora alliance®,” 2020, loRa Alliance. Disponible en: <https://lora-alliance.org/about-lorawan/>.
- [24] Semtech. (2021) Lorawan standard — lora — semtech. Semtech. Disponible en: <https://semtech.com/lora/lora-and-lorawan>. [Online]. Available: <https://semtech.com/lora/lora-and-lorawan>
- [25] D. A. Sierra Perera, G. Milián Teijeiro, I. Quesada Hernández, L. Pérez Roche, y P. M. Perera Miniet, “Marco teórico para la implementación de la geolocalización en redes lpwan,” *Revista Telemática*, vol. 22, pp. 45–61, 2023. [Online]. Available: <https://revistatelematica.cujae.edu.cu/index.php/tele/article/view/629/507>
- [26] T. things network, “Arquitectura lorawan,” fecha de acceso, 2024. [Online]. Available: <https://www.thethingsnetwork.org/docs/lorawan/architecture/>
- [27] J. P. M. Álvarez, V. I. R. Abdalá, F. M. R. M. Barboza, y F. R. C. Soria, “Estudio descriptivo de lorawan y aplicaciones específicas,” *difu100cia*, vol. 15, no. 1, pp. 8–17, Apr. 2021.
- [28] “Security — the things network,” <https://www.thethingsnetwork.org/docs/lorawan/security/>, 2024, explica la implementación de claves de seguridad y las medidas contra ataques de repetición en LoRaWAN.
- [29] S. Gemalto, Actility, “Lorawan™ security white paper,” 2017, describe los principios de autenticación, integridad y confidencialidad en la seguridad de LoRaWAN, incluyendo la encriptación de extremo a extremo

- para datos de aplicaciones IoT. [Online]. Available: https://lora-alliance.org/wp-content/uploads/2020/11/lorawan_security_whitepaper.pdf
- [30] D. de teleónica, “¿qué es rfid?” [Online]. Available: <https://telelectronica.com/rfid-faq/>
 - [31] Mohammad, Gouse, Shitharth, S. Syed, Dugyala, Raman, Rao, K.Sreenivasa, Alenezi, Fayadh, Althubiti, Sara, Polat, y Kemal, “Mechanism of internet of things (iot) integrated with radio frequency identification (rfid) technology for healthcare system,” *Mathematical Problems in Engineering*, vol. 2022, pp. 1–8, 03 2022.
 - [32] J. FOMBONA y E. VÁZQUEZ, “Posibilidades de utilización de la geolocalización y realidad aumentada en el ámbito educativo,” 2017, citado el: 18 de Junio de 2021.
 - [33] L. I. Benavides Segura y B. D. Cárdenas Espinoza, “Implementación de un dispositivo iot, basado en tecnología lora para la geolocalización y monitoreo fisiológico de personas en lugares turísticos,” dec 2021, escuela Superior Politécnica de Chimborazo. Riobamba. [Online]. Available: <http://dspace.esepoch.edu.ec/handle/123456789/21248>
 - [34] concepto de servicio, “Gestión de la entrega de servicios.” [Online]. Available: <https://www.nomadia-group.com/es/recursos/blog/la-gestion-de-la-entrega-de-servicios-optimizando-la-satisfaccion-del-cliente/>
 - [35] J. M. V. S. Hole y N. Wetterskog, “Lokalisering av sensorer med lorawan på kalmar länssjukhus,” Ph.D. dissertation, Dissertation, 2021. [Online]. Available: <https://www.diva-portal.org/smash/get/diva2:1612938/FULLTEXT01.pdf>
 - [36] S. Ponis, G. Plakas, E. Aretoulaki, D. Tzanetou, y T. N. Maroutas, “Lorawan for tracking inland routes of plastic waste: Introducing the smart trackplast bottle,” *Cleaner Waste Systems*, vol. 4, p. 100068, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2772912522000689>
 - [37] C. Civelek, “Development of an iot based (lorawan) tractor tracking system,” *Tarım Bilimleri Dergisi*, 08 2021.
 - [38] M. Hamidu, “Use of rfid technology as a reporting mechanism in vehicle tracking system,” *Advances in Wireless Communications and Networks*, vol. Volume 2, pp. Pages: 1–10, 12 2016.

- [39] Heltec, “Wifi lora 32 v3,” 2023. [Online]. Available: <https://heltec.org/project/wifi-lora-32-v3/>
- [40] Semtech, “Sx1262,” 2023. [Online]. Available: <https://www.semtech.com/products/wireless-rf/lora-connect/sx1262>
- [41] A. Cherkaoui, S. Merzouk, A. Marzak, y M. Hain, “Review on embedded systems and the internet of things: Comparative study,” *Proceedings of the 4th International Conference on Networking, Information Systems & Security*, 2021.
- [42] A. Latifov y A. Pradeep, “Design and implementation of a lora-based home monitoring system with heltec esp32 gateway,” *2023 15th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pp. 01–06, 2023.
- [43] N. Jengsriwong y S. Chansareewittaya, “Lorawan gps tracker,” *2023 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON)*, pp. 199–202, 2023.
- [44] N. Semiconductors, “Mfrc522 datasheet,” 2023. [Online]. Available: <https://www.elecrow.com/download/MFRC522%20Datasheet.pdf>
- [45] *GP-02 GPS Development Board Specifications*, Shenzhen Ai-Thinker Technology Co., Ltd, 2021, accessed: Oct. 02, 2024. [Online]. Available: https://cdn.robotshop.com/media/A/AIT/RB-Ait-24/pdf/ai_thinker_gp_02_gps_development_board_gp_02_kit_specifications.pdf
- [46] E. Systems, “Esp32-wroom-32, datasheet,” 2019, [Online]. Available: <https://pdf1.alldatasheet.com/datasheet-pdf/download/1179101/ESPRESSIF/ESP-WROOM-32.html>.
- [47] (2024) Thingsboard - open-source iot platform. ThingsBoard, Inc. Accessed: Oct. 02, 2024. [Online]. Available: <https://thingsboard.io/>
- [48] A. Alquhali, M. Roslee, M. Y. Alias, y K. S. Mohamed, “Iot based real-time vehicle tracking system,” *2019 IEEE Conference on Sustainable Utilization and Development in Engineering and Technologies (CSU-DET)*, pp. 265–270, 2019.
- [49] O. U. Nwankwo, C. I. Nwakanma, D. S. Kim, y J.-M. Lee, “Iot-assisted intelligent vehicle tracking system using cloud computing,” in *2022 13th*

International Conference on Information and Communication Technology Convergence (ICTC), 2022, pp. 1677–1679.

- [50] J. Santa, R. Sanchez-Iborra, P. Rodriguez-Rey, L. Bernal-Escobedo, y A. Gómez-Skarmeta, “Lpwan-based vehicular monitoring platform with a generic ip network interface,” *Sensors (Basel, Switzerland)*, vol. 19, 2019.
- [51] M. Suwaid, M. Habaebi, y S. Khan, “Embedded lorawan for agricultural sensing applications,” in *2019 IEEE 6th International Conference on Engineering Technologies and Applied Sciences (ICETAS)*, 2019, pp. 1–5.
- [52] S. A. Alavi, M. Javadipour, y K. Mehran, “State monitoring for situational awareness in rural microgrids using the iot infrastructure,” *ArXiv*, vol. abs/1906.00437, 2019.
- [53] S. K, J. S, S. B, y G. P, “Iot based smart helmet for workers in mines using lorawan,” *2023 2nd International Conference on Vision Towards Emerging Trends in Communication and Networking Technologies (ViTECoN)*, pp. 1–6, 2023.
- [54] (2024) Mqtt: The standard for iot messaging. MQTT.org. [Online]. Available: <https://mqtt.org/>
- [55] Arduino, “Arduino ide,” [Online]. Disponible: <https://docs.arduino.cc/software/ide/>.
- [56] ——, “Getting started with arduino ide 2,” [Online]. Disponible: <https://docs.arduino.cc/software/ide-v2/tutorials/getting-started-ide-v2/>.
- [57] Fritzing, “Fritzing: Software para la automatización del diseño electrónico (eda),” [Online]. Disponible: <https://fritzing.org/>.
- [58] ——, “Fritzing: Primeros pasos,” [Online]. Disponible: <https://fritzing.org/media/uploads/learning/translations/Fritzing-PrimerosPasos.pdf>.
- [59] (2024) About blender. Blender Foundation. [Online]. Available: <https://www.blender.org/about/>
- [60] (2024) Blender. Blender Foundation. [Online]. Available: <https://www.blender.org/>
- [61] (2024) Helium console. Helium Foundation. [Online]. Available: <https://console.helium.com/>

- [62] D. de Heltec, “Documentación de heltec oficial para el uso de wifi lora 32.” [Online]. Available: https://docs.heltec.org/en/node/esp32/wifi_lora_32/index.html#
- [63] D. de Helium, “Guía de arduino para heltec wifi lora 32 v2.” [Online]. Available: <https://docs.helium.com/network-iot/devices/development/heltec/wifi-lora-32-v2/arduino/>
- [64] (2024) Installing thingsboard using docker (windows). ThingsBoard, Inc. Accessed: Oct. 20, 2024. [Online]. Available: <https://thingsboard.io/docs/user-guide/install/docker-windows/>
- [65] (2024) Installing thingsboard doc). ThingsBoard, Inc. Accessed: Nov. 01, 2024. [Online]. Available: <https://thingsboard.io/docs/guides/>
- [66] (2024) Google maps. Google LLC. [Online]. Available: <https://maps.google.com>
- [67] G. Earth, “Google earth - visualización de coordenadas y medición de distancias,” <https://earth.google.com/web/>, 2024, imágenes utilizadas para validar las coordenadas obtenidas con el módulo GNSS y medir distancias entre puntos.
- [68] O. Calculator, “Calculadora de distancia entre coordenadas (latitud y longitud),” <https://www.omnicalculator.com/es/otros/calculadora-latitud-longitud-distancia>, 2024, herramienta utilizada para calcular distancias precisas entre puntos GPS adquiridos.
- [69] (2024) Helium explorer. Helium Foundation. [Online]. Available: <https://explorer.helium.com/>
- [70] (2024) Base64 to hex converter. Cryptii. [Online]. Available: <https://cryptii.com/pipes/base64-to-hex>
- [71] (2024) Online hex converter. SCADACore. [Online]. Available: <https://www.scadacore.com/tools/programming-calculators/online-hex-converter/>