Memoria

Trabajo de fin de Grado

GRADO EN INGENIERÍA INFORMÁTICA

VNiVERSiDAD
Đ SALAMANCA

Julio de 2025

Autor

Juan Calles Rivas

Tutor/a

Sergio García Sánchez

Diego Manuel Jiménez Bravo

Resumen

Este Trabajo Fin de Grado aborda el diseño y la realización de una solución integral de alertas de emergencia, destinada principalmente a personas mayores o vulnerables en la España rural. El proyecto garantiza su funcionalidad en entornos con dificultades de conectividad, donde la cobertura móvil es baja o prácticamente nula.

La solución propuesta consiste en un sistema tecnológico integral conformada por los siguientes elementos:

- **Dispositivo de usuario**: este dispositivo cuenta con capacidad de detección de caídas mediante algoritmos de inteligencia artificial (IA) y un botón para notificar alertas de forma manual.
- **Infraestructura de radio propia**: para la realización del proyecto, se diseñó una red de telecomunicaciones propia basada en tecnología LoRa (*Long Range*) para dar soporte a los dispositivos.
- **Plataforma *web***: su objetivo es permitir la administración del sistema de forma centralizada.

El núcleo innovador del proyecto reside en gran parte en el desarrollo de un protocolo de comunicación a medida sobre LoRa, denominado *ISHA* (*Inter-System Hybrid Alerting*), que garantiza la seguridad, confidencialidad e integridad de los datos de extremo a extremo (desde la emisión por parte del dispositivo del usuario, hasta el servidor y viceversa).

El resultado global del proyecto es un prototipo completamente funcional que demuestra la viabilidad de la solución propuesta, ofreciendo una respuesta tecnológica robusta y escalable a un problema social creciente en España.

Abstract

This Final Degree Project addresses the design and implementation of a comprehensive emergency alert solution, mainly intended for elderly and/or vulnerable people in rural Spain. It guarantees functionality in challenging environments with limited or virtually no mobile network coverage.

The proposed system integrates three core components:

1. **User device:** a wearable unit featuring AI-driven fall-detection algorithms and a manual button for emergency alerts.
2. **Proprietary radio infrastructure:** a dedicated LoRa (Long Range) network deployed to ensure connectivity in areas lacking conventional mobile coverage.
3. **Web platform:** a centralized management portal for system administration, real-time monitoring, and alert configuration.

The project's innovative core resides in **ISHA** (Inter-System Hybrid Alerting), a bespoke communication protocol operating over LoRa. ISHA provides end-to-end security, confidentiality, and data integrity for all transmissions between user devices and the central server (bidirectionally).

The outcome is a fully functional prototype validating the system's feasibility, delivering a robust, scalable technological response to a growing societal challenge in rural Spain.

Keywords: Emergency Alert System, LoRa, Fall Detection, Artificial Intelligence.

Agradecimientos

Agradecimientos a los tutores del proyecto Sergio García Sánchez y Diego Manuel Jiménez Bravo, por su apoyo y colaboración durante el desarrollo del sistema.

Agradecimientos especiales a José Antonio García García (Técnico de Laboratorio del Departamento de Física Aplicada, Universidad de Salamanca) por su ayuda incondicional en cuanto respecta al proceso de soldadura de componentes hardware.

Agradecimientos a los ayuntamientos de Palacios del Arzobispo y Villamayor de Armuña por el apoyo simbólico del proyecto ante el jurado TCUE.

Agradecimiento especial a (*Beijing Mengxing Electronic Technology Co., Ltd.*) por proporcionar de forma totalmente gratuita los módulos de carga que están integrados en los dos prototipos funcionales.

Palabras clave: Sistema de Alertas de Emergencia, LoRa, Detección de Caídas, Inteligencia Artificial.

Tabla de contenido

Tabla ilustraciones

Índice de tablas

Glosario

A

- **ADL (*Activities of the Daily Living* / Actividades de la Vida Diaria):** conjunto de movimientos y acciones cotidianas (caminar, sentarse, agacharse) que no se consideran una "caída". Se utilizan como datos de la clase "no caída" para entrenar el modelo de inteligencia artificial.
- **API (*Application Programming Interface* / Interfaz de Programación de Aplicaciones):** conjunto de reglas y herramientas que permite que distintas aplicaciones de *software* se comuniquen entre sí sin necesidad de compartir arquitectura *hardware* o *software*, permitiendo tener un "lenguaje común". En este proyecto, el *backend* expone una API para recibir los datos de las antenas. Las misas antenas exponen una API de configuración de las mismas durante su proceso de instalación.
- ***Arduino IDE* Entorno de desarrollo (IDE):** utilizado para programar el *firmware* de los dispositivos. Fue seleccionado por su simplicidad, rapidez para el prototipado y la experiencia previa del autor.
- **ASGI (*Asynchronous Server Gateway Interface*):** interfaz estándar entre servidores *web* y aplicaciones, permite el procesamiento de peticiones de forma asíncrona, mejorando el rendimiento. Es una de las tecnologías en las que se basa *FastAPI.*

B

- ***BaaS* (*Backend-as-a-Service*):** modelo de computación en la nube que externaliza la gestión y el mantenimiento de la infraestructura del lado del servidor. En el proyecto se utiliza Supabase como *BaaS* para la persistencia de datos, autenticación y notificaciones contra el panel *web*.
- ***Backend*:** componente del sistema que se ejecuta en el servidor y es responsable de la lógica de negocio. En este proyecto, gestiona la recepción de datos de las antenas, el descifrado de tramas, la comunicación con la base de datos y el envío de alertas.

C

- ***Chirp Spread Spectrum* (CSS):** técnica de modulación utilizada por la tecnología *LoRa*. Codifica los datos en señales de radiofrecuencia (*chirps*) cuya frecuencia varía en el tiempo, lo que las hace muy resistentes al ruido y a las interferencias.
- **CRC (*Cyclic Redundancy Code* / Código de Redundancia Cíclica):** código de detección de errores utilizado para verificar la integridad de los datos durante la transmisión de las tramas. Es computacionalmente más ligero que una función *hash*, siendo ideal para la validación de tramas en dispositivos con recursos limitados.
- ***CRUD* (*Create, Read, Update, Delete*):** acrónimo que se refiere a las cuatro operaciones básicas en la gestión de la persistencia de datos: Crear, Leer, Actualizar y Borrar.

D

- ***Docker*:** plataforma de código abierto que permite automatizar el despliegue de aplicaciones dentro de contenedores virtualizados. En el proyecto se utiliza para aislar y ejecutar los servicios de *frontend* y *backend* en el servidor de producción en cheetahost.

E

- **ESP32:** microcontrolador de bajo coste y bajo consumo con conectividad *Wi-Fi* y *Bluetooth* integrada. Es el cerebro de los dispositivos de usuario y de las antenas desarrollados en el proyecto.

F

- ***FastAPI*:** *framework* de alto rendimiento para construir *APIs* en Python. Fue la tecnología seleccionada para el desarrollo del *backend* del sistema por su velocidad, su documentación automática y la validación de datos robusta de datos.
- ***Firmware*:** tipo de *software* específico que se programa en la memoria no volátil de un dispositivo *hardware* para controlar su funcionamiento a bajo nivel. En este proyecto, es el código que se ejecuta en los ESP32 de los dispositivos de usuario y las antenas.
- ***Frontend*:** parte de una aplicación de *software* que interactúa directamente con el usuario, es decir, la interfaz gráfica. En este proyecto, es la plataforma *web*.

G

- ***Git*:** sistema de control de versiones distribuido, utilizado para rastrear los cambios en el código fuente durante el desarrollo del proyecto *software*.
- ***GitHub*:** plataforma de alojamiento para repositorios *Git*, facilita el desarrollo y el control de versiones. Se utilizó para gestionar todo el código fuente del proyecto.
- ***GPS* (*Global Positioning System* / Sistema de Posicionamiento Global):** sistema de radionavegación basado en una red de satélites que permite determinar la posición geográfica (latitud, longitud y altitud) de un receptor en cualquier parte de la tierra.

I

- ***ISHA* (*Inter-System Hybrid Alerting*):** protocolo de comunicación, diseñado y desarrollado para este proyecto, que opera sobre *LoRa*. Define la comunicación segura (cifrado AES-256 de extremo a extremo) y la estructura de los mensajes entre el dispositivo de usuario y el servidor (*backend)*, a través de las antenas/*gateway*.

L

- ***LoRa* (*Long Range*):** tecnología de modulación de radiofrecuencia que permite la comunicación inalámbrica a larga distancia con un consumo de energía muy bajo. Es la base de las comunicaciones del sistema, ideal para su despliegue en zonas rurales.
- ***LoRaWAN*:** protocolo de red de área amplia y baja potencia (*LPWAN*) que opera sobre la capa física *LoRa* y define la arquitectura de comunicación. Se consideró, pero se descartó en favor de *ISHA* para tener un mayor control sobre la seguridad y una arquitectura más simple y optimizada para la especificación del sistema.

M

- **MEMS (*Micro-Electro-Mechanical Systems*):** tecnología que integra elementos mecánicos y eléctricos a escala microscópica. Los giroscopios y acelerómetros utilizados en el proyecto son dispositivos MEMS.
- **MPU (*Motion Process Unit* / Unidad de Procesamiento de Movimiento):** componente electrónico que integra un acelerómetro y un giroscopio en un solo integrado/*chip* (como el MPU-6050 usado en el proyecto) para medir el movimiento y la orientación en múltiples ejes.

O

- ***Overfitting* (Sobreajuste):** problema en el aprendizaje automático donde un modelo se adapta excesivamente a los datos de entrenamiento, perdiendo la capacidad de hacer predicciones precisas sobre datos nuevos. El uso de la técnica de *Random Forest* ayuda a mitigar este riesgo.

P

- **Patrón de Diseño *Strategy*:** patrón de comportamiento de *software* que permite definir e intercambiar algoritmos en tiempo de ejecución. Se aplicó en el *firmware* para que el mismo código base pudiera funcionar como "Dispositivo" o como "Antena" solamente cambiando la "estrategia" de operación.
- **PCB (*Printed Circuit Board* / Placa de Circuito Impreso):** soporte físico donde se instalan y conectan eléctricamente los componentes electrónicos. En el proyecto se diseñó y fabricó una PCB a medida para miniaturizar el prototipo.
- **PostgreSQL**: sistema de gestión de bases de datos relacionales de código abierto. Es la base de datos subyacente utilizada por Supabase, donde se almacena toda la información del sistema.
- **Proceso** unificado: Metodología de desarrollo de *software* iterativa e incremental, centrada en la arquitectura y guiada por casos de uso. Fue el marco de trabajo adoptado para la gestión del ciclo de vida de este proyecto.

R

- ***Random Forest* (Bosque Aleatorio):** algoritmo de aprendizaje automático supervisado basado en el ensamblado (*ensemble learning*) de múltiples árboles de

decisión. Fue el modelo considerado para la detección de caídas por su robustez y su capacidad para reducir el sobreajuste.

- ***React*:** biblioteca de JavaScript de código abierto, desarrollada por Meta (Facebook), para construir interfaces de usuario (*frontend*). Fue la tecnología seleccionada para el desarrollo de la plataforma *web* del proyecto.

S

- **SIM (*Subscriber Identity Module* / Módulo de Identidad del Suscriptor):** tarjeta inteligente utilizada en dispositivos móviles para identificar a un suscriptor en la red. Los sistemas de teleasistencia convencionales dependen de ellas y de la cobertura móvil, una limitación que este proyecto resuelve con *LoRa*.
- **SMD (*Surface Mount Device* / Dispositivo de Montaje Superficial**): componente electrónico diseñado para ser soldado directamente sobre la superficie de una PCB. La elección accidental de resistencias SMD en lugar de axiales fue uno de los problemas encontrados durante el ensamblaje del *hardware*.
- **Supabase:** plataforma *open source* que ofrece una alternativa a Firebase, proporcionando un *backend* completo (BaaS) basado en PostgreSQL. Fue la solución elegida para la persistencia de datos por su flexibilidad, transparencia y por evitar el *vendor lock-in*.

T

- **TCUE (Transferencia de Conocimiento Universidad-Empresa):** plan que promueve la colaboración entre la universidad y la industria para transformar la investigación en productos o servicios comerciales. El proyecto fue seleccionado para este plan, lo que implicó la elaboración de un plan de negocio.
- **trilateración:** Método matemático utilizado por los receptores *GPS* para determinar su posición. Se basa en medir la distancia a un mínimo de tres satélites para calcular la intersección de las esferas de señal y obtener las coordenadas 2D (latitud y longitud).

V

- ***Vendor Lock-in*:** situación de dependencia en la que un cliente queda "atrapado" con un proveedor de tecnología debido a los altos costes o la complejidad que supondría migrar a un sistema alternativo. Se eligió Supabase en parte para evitar este riesgo, asociado a plataformas propietarias como Firebase.

1. Introducción

1.1. Contextualización del problema.

Durante las últimas décadas, España ha experimentado un notable descenso de la población activa en sus zonas rurales [1], [2], fenómeno que ha dado lugar al término de la "España vaciada". Como consecuencia de esto, los servicios en estas regiones han ido disminuyendo paulatinamente, dejándolas aún más desamparadas.

La mayoría de los habitantes que residen en estas zonas son personas de edad avanzada. Esto hace que estas regiones sean zonas geográficas con población de riesgo. Las personas mayores son un colectivo especialmente propenso a enfermedades y a sufrir daños considerables en caso de caídas [3]. Estas caídas se pueden agravar considerablemente en el caso de que no sean atendidas a tiempo [4], por la incapacidad de la persona mayor para levantarse de forma autónoma o por no poder notificar a personas de confianza de la caída. De esta problemática surge la idea de este proyecto.

1.2. Presentación de la solución

Este proyecto nace con el objetivo de solventar la problemática presentada en la subsección anterior. La solución es crear un sistema de emergencia real, accesible y capaz de operar en entornos rurales donde la capacidad de operación de los dispositivos convencionales de estas características es prácticamente nula, dada la calidad de las redes móviles SIM (*Subscriber Identity Module*) [5] en estas regiones. Para dar respuesta a esta característica, para la solución presentada se ha realizado una búsqueda de tecnologías de telecomunicación. Esta búsqueda dio como resultado una red de comunicación a través de radiofrecuencia de largo alcance y bajo consumo energético *LoRa* (*Long Range*). Con base en esta tecnología se ha diseñado un protocolo propio que permita la fiabilidad y robustez del sistema. Además de esta contribución, la solución se modela en base a otros dos pilares. A continuación, se detallan los tres pilares en los que se fundamenta el sistema.

1.2.1. Dispositivo de usuario:

Para la consecución de este proyecto se ha diseñado y construido un dispositivo específico. Este dispositivo ha sido concebido para ser lo más portable, fácil de utilizar y mantener, dentro del marco que implica un prototipo.

Este dispositivo está equipado con una unidad *GPS*, *LoRa* y acelerómetro-giroscopio. Además, mediante algoritmos de Inteligencia Artificial (IA), y a partir de la lectura de los datos de los sensores mencionados, el dispositivo es capaz de detectar caídas de forma autónoma. Adicionalmente, el dispositivo incorpora un botón de emergencia para la activación manual de alertas, garantizando así una doble vía de aviso.

1.2.2. Infraestructura de telecomunicación propia

Para superar las limitaciones tecnológicas en las áreas rurales, el sistema se basa en una red de dispositivos, "antenas", que utilizan la tecnología *LoRa* para recibir información de los "dispositivos de usuario" actuando como puente/*bridge,* permitiendo la comunicación dispositivo-servidor. El despliegue de estas antenas se realiza en posiciones estratégicas para dar cobertura al mayor número de usuarios posible. De este modo, se da servicio al sistema sin depender de la red de telefonía móvil, asegurando la transmisión fiable de las alertas desde el dispositivo del usuario hasta el servidor central y viceversa.

1.2.3. Plataforma *software* centralizada:

Consiste en una aplicación *web* que funciona como centro de operaciones del sistema. Esta plataforma ofrece vistas personalizadas según el nivel de autorización de cada usuario:

- **Panel de administración**: ofrece una solución integral para la gestión de usuarios, dispositivos, antenas y otros elementos del sistema.
- **Panel de emergencias**: permite a los responsables geográficos o a los servicios de emergencia gestionar las incidencias en tiempo real, accediendo a la información crítica del suceso.
- **Portal para usuarios y familiares**: los familiares pueden consultar la información médica del portador del dispositivo, su historial de localizaciones, editar su propio perfil, etc. Por su parte, el usuario portador puede revisar su historial de localización y editar sus datos sanitarios.

1.2.4. Declaración de propósito y alcance

El propósito de este Trabajo de Fin de Grado es demostrar a través de un TFG completo y funcional sustentado por (i) el dispositivo *hardware*, (ii) la infraestructura de comunicaciones y (iii) la plataforma *software* la viabilidad de una solución para el problema descrito anteriormente. El proyecto presentado en este documento puede ayudar a resolver o mitigar un problema de real y con gran impacto en la sociedad de nuestra región.

1.3. Estructura del documento

A lo largo de esta Memoria se mostrará la estructura completa a alto nivel del trabajo realizado y los aspectos más importantes del mismo. Estos son:

- **Objetivos**: las metas propuestas que han ido modelando el proyecto a lo largo de su desarrollo.
- **Conceptos teóricos**: los diversos conceptos técnicos/teóricos que conforman la aplicación, haciendo entendible el funcionamiento del sistema por parte del lector y sus funciones en el cómputo global del mismo.
- **Técnicas y herramientas:** herramientas y técnicas utilizadas a lo largo d la realización del proyecto.
- **Aspectos relevantes del proyecto:** aspectos más relevantes del proyecto y cuál ha sido su papel durante el desarrollo del mismo.
- **Trabajos relevantes:** trabajos similares al proyecto, pudiendo servir como referencia para poder mejorar e innovar.
- **Conclusiones:** punto de vista después de la realización del proyecto, generada a partir de las vivencias del alumno durante el proceso de realización de este.
    - **Líneas futuras:** posibles mejoras y/o actualizaciones que se pudieran aplicar al proyecto en un futuro.
- **Bibliografía.**

La memoria del proyecto va complementada con la realización de Anexos específicos, estos entran en mayor detalle dada su sección correspondiente. Los Anexos son los siguientes:

- **Anexo I: Plan del proyecto *software***. En este anexo se realiza la estimación del esfuerzo correspondiente al proyecto y la planificación temporal del mismo.
- **Anexo II: Especificación de requisitos del *software***. En este anexo, se realiza la especificación de requisitos del sistema.
- **Anexo III: Especificación de diseño**. En este anexo se detallará la especificación técnica de diseño del proyecto.
- **Anexo IV: Documentación técnica de programación**. Este anexo entra en mayor detalle sobre los conceptos técnicos, tecnologías y ecosistemas utilizados durante la realización del proyecto.
- **Anexo V: Manual del usuario**. En este anexo, se explica en mayor detalle como lo usuarios finales interactuaran de forma correcta con los diversos componentes del sistema.
- **Anexo VI: Especificación de componentes *hardware***. En este anexo se abordará con mayor detalle el diseño, fabricación y ensamblado de los dispositivos *hardware* del sistema, antena/s y dispositivo/s de usuario.
- **Anexo VII: Especificación de aprendizaje automático para la detección de caídas**. En este anexo, se profundizará en el proceso específico llevado a cabo para la realización del modelo detector de caídas.
- **Anexo VIII: Plan de negocio TCUE**. Este proyecto ha sido seleccionado por el plan de transferencia conocimiento universidad-empresa, por lo que se ha realizado un plan de negocio real del mismo.

2. Objetivos del proyecto

Los objetivos del sistema han sido sintetizados y extraídos de la definición intrínseca del mismo proyecto, derivados del análisis del problema y las necesidades que este implica. Asimismo, se incluyen en esta sección una serie de objetivos personales.

2.1. Objetivos técnicos

El objetivo principal del proyecto es la realización de un sistema de emergencias para la detección automática de caídas en zonas rurales utilizando una infraestructura de telecomunicaciones propia basadas en *Lora*, y tecnología de geoposicionamiento.

Los objetivos técnicos sintetizados que se quieren alcanzar son los siguientes.

- **Gestión de usuarios**: el sistema debe ser capaz de gestionar usuarios de forma totalmente concurrente y transparente. Se muestran diferentes vistas especificas dependiendo del nivel de autoridad de los usuarios en el sistema, pudiendo manejar información personal (dirección, teléfono, vínculos familiares, etc.), clínica (patologías, alergias, observaciones, grupo sanguíneo, etc.) y de geoposicionamiento (historial de localizaciones).
- **Comunicación segura**: el sistema debe proporcionar la certeza de que ningún tipo de datos sensibles ha sido expuesto al medio sin haber sufrido una transformación criptográfica previa que permita la privacidad, integridad y confidencialidad de los datos.
- **Detección de caídas**: el sistema debe ser capaz de detectar de forma autónoma la caída del portador e iniciar la secuencia de notificación correspondiente, permitiendo atender/socorrer al usuario afectado, en el menor tiempo posible.
- **Alerta manual**: el dispositivo debe ser capaz de activar la secuencia de notificación correspondiente.
- **Portabilidad del dispositivo**: el dispositivo debe ser portable dentro del marco del prototipado.
- **Gestión de dispositivos**: el sistema debe garantizar la gestión y coordinación de varios dispositivos.
- **Gestión de notificaciones**: el sistema debe ser capaz de registrar las notificaciones, avisar a los familiares y gestionar su estado.
- **Gestión de dispositivos *hardware***: el sistema debe poder gestionar los diferentes componentes *hardware* que lo conforman.
- **Gestión de autenticación**: el sistema debe llevar a cabo un control tanto de dispositivos (antenas y/o dispositivos de usuario) como de usuarios, garantizando la identidad de cualquier actor que realice operaciones en el sistema (sin contar el usuario anónimo).

2.2. Objetivos personales

El desarrollo autónomo e íntegro de un proyecto de estas características es algo completamente nuevo y supone un reto para el estudiante. Este se puede asemejar más al trabajo real de un ingeniero de *software* multidisciplinario. Dado que el proyecto supone un gran reto para el desarrollador ya que en el mismo se han aplicado prácticamente todos los conocimientos adquiridos durante el grado, además de profundizar en áreas específicas que el alumno ha tenido que aprender de forma autónoma y proactiva. Con esto en mente, se exponen una serie de objetivos personales que el autor del trabajo quiere cumplir una vez el proyecto concluya.

- **Diseño de dispositivos *hardware*:** los conocimientos del *hardware* que se imparten en el grado son muy específicos y a muy bajo nivel, presentando las bases de toda la computación. Estos contenidos son impartidos durante los primeros semestres del grado. No obstante, y gracias a asignaturas optativas, como "periféricos", el alumno ha podido familiarizarse más con los conceptos y técnicas necesarias para la construcción de dispositivos simples, similares al planteado en este TFG. Para la realización del proyecto se plantea el objetivo personal de diseñar y fabricar circuitos integrados (PCB) de forma totalmente proactiva y autónoma.
- **Programación de sistemas embebidos:** este objetivo personal pretende adquirir los conocimientos necesarios para trabajar con sistema embebido y sistemas en tiempo real con el objetivo de poder con las exigencias de desarrollo que un dispositivo de estas características implica. Durante el grado se han cursado asignaturas donde se tratan la programación en tiempo real y los sistemas embebidos, estos conocimientos no son suficientes para cubrir la totalidad de los conceptos necesarios para la realización del proyecto.
- **Aplicar e integrar los conocimientos multidisciplinarios adquiridos durante el grado*:*** este objetivo ha sido el mayor desafío para el alumno, ya que ha requerido asumir el rol de desarrollador *Full Stack*, para poder abordar todos los aspectos del proyecto. Se pusieron en práctica las competencias adquiridas durante el grado, entre las que destacan:
    - **Ingeniería de *software* y Gestión de proyectos**: para la especificación, planificación, diseño y ejecución del proyecto, así como la planificación de costes.
    - **Programación I, II, III, avanzada y EDA I, II:** utilizando paradigmas de programación estructuras de datos siendo las bases del sistema.
    - **Administración de Sistemas, Redes y Seguridad:** conocimientos aplicados para el despliegue del sistema en los servidores, diseño del protocolo de comunicación propio y la aplicación del sistema criptográfico.
    - **Bases de Datos:** para el diseño e implementación de la persistencia dadas las características que exige el proyecto.
    - **Periféricos, Interfaces e IPO:** conocimientos utilizados para la creación del dispositivo físico, la selección de sensores y el diseño de interfaces sencillas e intuitivas para el usuario.
    - **Sistemas inteligentes:** aplicando algoritmos clásicos de inteligencia artificial para la detección de caídas.

3. Conceptos teóricos

En esta sección se entrará en mayor detalle sobre los conceptos teóricos considerados relevantes para la realización del proyecto. Con el objetivo de facilitar la comprensión de este por parte del lector.

3.1. Dispositivo de teleasistencia convencionales

Los dispositivos de teleasistencia convencionales son aparatos pensados principalmente para dar asistencia de forma telemática a un usuario. Tienen como objetivo, dar una capa de seguridad al usuario del dispositivo, normalmente están destinados a personas mayores/dependientes que no viven acompañadas [6].

Normalmente son dispositivos portables, podemos distinguir dos tipos principalmente. Los basados en sistemas de telefonía o los basados en base estática.

- **Los dispositivos basados en telefonía**: estos dispositivos son teléfonos móviles encubiertos, los cuales tienen una tarjeta SIM [5] integrada ya sea física o virtual. Así, estos dispositivos utilizan la red de telefonía móvil convencional [5] como medio de comunicación al exterior. Se presentan a los usuarios como supuestos relojes inteligentes de pulsera o teléfonos móviles simplificados.

Ilustración 1 - Dispositivo móvil simplificado [7].

Ilustración 2 - Dispositivo móvil el formato smartwatch [7].

- **Los dispositivos basados en bases estáticas**: estos sistemas constan de dos elementos fundamentales, una "base" que permanece conectada continuamente a

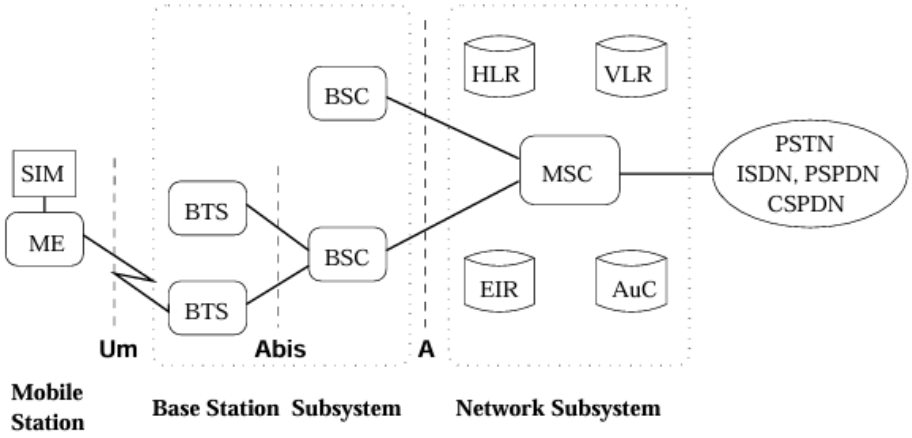
la red eléctrica y un botón de emergencia ya sea en forma de medalla o pulsera. El sistema es parecido al basado en telefonía ya que la base es en realidad un dispositivo móvil encubierto con su SIM [5] integrada, física o virtual. La única funcionalidad extra que presenta es el botón de emergencia el cual no funciona fuera de casa ya que en realidad se trata a un dispositivo similar a un mando a distancia de garaje con el que envía una señal a la base.

Ilustración 3 - Sistema de teleasistencia basado en bases estáticas [8]

3.2. Red de telefonía móvil G y tarjetas SIM

Las tarjetas SIM [5] son un pequeño *chip* que almacena información de un número de teléfono convencional. Se utiliza como elemento de identificación obligatorio para el acceso a las redes de telefonía GSM [5] (*Global System for Mobile communications).*

A través este sistema, los dispositivos anteriores realizan él envió de alertas, lo cual es un inconveniente ya que en las zonas rurales estas tecnologías suelen ser bastante inestables o prácticamente inexistentes. Por ello, las soluciones de teleasistencia basadas en estas tecnologías dejan mucho que desear en zonas rurales [9].



SIM Subscriber Identity Module BSC Base Station Controller MSC Mobile services Switching Center
ME Mobile Equipment HLR Home Location Register EIR Equipment Identity Register
BTS Base Transceiver Station VLR Visitor Location Register AuC Authentication Center

Ilustración 4 - Diagrama de la arquitectura de la red GSP [5].

El diagrama anterior muestra la arquitectura de la red GSM, esta son las bases de la red de comunicación móvil moderna. Esta red se diseñó con el objetivo de realizar llamadas de voz, es muy importante entender cómo los dispositivos de teleasistencia convencionales se comunican a través de internet para enviar alertas.

El proceso de envío de una alerta a través de datos, siguen los siguientes pasos:

- **Identificación y acceso a la red (*mobile station* y BSS):** tal y como pasaba con las igual que pasaba con las llamadas telefónicas, comienza en la Estación Móvil (MS), donde se combina el dispositivo físico (ME) y la tarjeta SIM **(**dispositivos de teleasistencia convencionales). La SIM actúa como el identificador único que da acceso a la red. El dispositivo se conecta por radio (a través de la interfaz Um) a la antena más cercana disponible (BTS), y esta, gestionada por su controlador (BSC), que es el que establece un canal de comunicación. Todo este proceso es totalmente el mismo tanto para una llamada telefónica convencional como para datos. La fiabilidad de este primer enlace por radio es el factor crítico que suele fallar en zonas rurales.
- **La ruta de los datos (*network subsystem*):** en esta sección, es donde se diferencia camino de los datos móviles al del de las llamadas telefónicas. Una vez la solicitud llega desde el BSC al núcleo de la red (NSS), en lugar de ser procesada por el MSC (que es la central para llamadas de voz), se desvía hacia un subsistema paralelo diseñado específicamente para el tráfico de internet, conocido como GPRS (*General Packet Radio Service*).

Aunque no se detallan en la figura anterior (diagrama clásico), este subsistema introduce dos nodos cruciales extendiendo el sistema clásico:

- **Un SGSN** (*Serving GPRS Support Node*), que actúa como un "MSC para datos". Se encarga de la autenticación y gestión de la movilidad del usuario que quiere conectarse a internet.
- **Un GGSN** (*Gateway GPRS Support Node*), que funciona como la puerta de enlace (*gateway*) final entre la red móvil y las redes de paquetes externas, como es Internet.

Conexión a internet: el GGSN es el componente que finalmente conecta la petición de datos del dispositivo de teleasistencia con el servidor de destino en internet. En el diagrama, esta conexión al mundo exterior de datos está representada de forma simplificada por el acceso a la **PSPDN (*Packet-Switched Public Data Network*)**, que es la categoría a la que pertenece Internet.

3.3. MPU: acelerómetro y giroscopio triaxiales

Los acelerómetros y giroscopios son periféricos que normalmente van juntos en lo que se denomina una MPU (*Motion Process Unit* o Unidad de Procesamiento de Movimiento en castellano). El encapsulado del acelerómetro y giroscopio se combina en el mismo encapsulado (*chip*) con la intención de obtener información más precisa sobre el movimiento, orientación de un objeto, etc. [10].

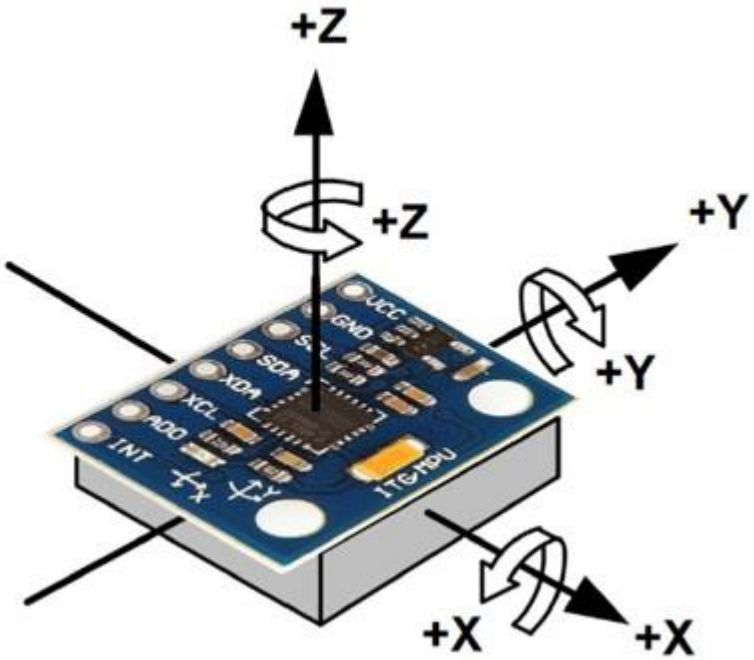


Ilustración 5 - Representación de la MPU 6050 utilizada para el proyecto [11].

Tras esta breve definición, a continuación, se aporta, con más nivel de detalle, información sobre el acelerómetro y el giroscopio.

- **Acelerómetro:** es un instrumento que mide la aceleración, que puede no ser la misma que la aceleración de coordenadas (cambio de velocidad en un espacio dado). Existen diferentes tipos de acelerómetros, siendo los más comunes el

mecánico y el piezoeléctrico. En este proyecto, nos enfocaremos en el acelerómetro piezoeléctrico, que es un tipo de transductor. Un transductor es un dispositivo que convierte una variable física de entrada en una salida eléctrica medible [12]. Los acelerómetros piezoeléctricos funcionan gracias al fenómeno piezoeléctrico, donde, al ser sometidos a tensiones mecánicas, generan una polarización eléctrica en su masa. Utilizando el principio de inercia, que describe la resistencia de un objeto a cambiar su estado de movimiento, podemos medir la aceleración. Los acelerómetros se clasifican según el número de ejes que pueden medir: uniaxiales (un eje), biaxiales (dos ejes) y triaxiales (tres ejes). En el contexto de este proyecto, se ha utilizado un acelerómetro triaxial, que permite medir la aceleración en tres dimensiones [12].

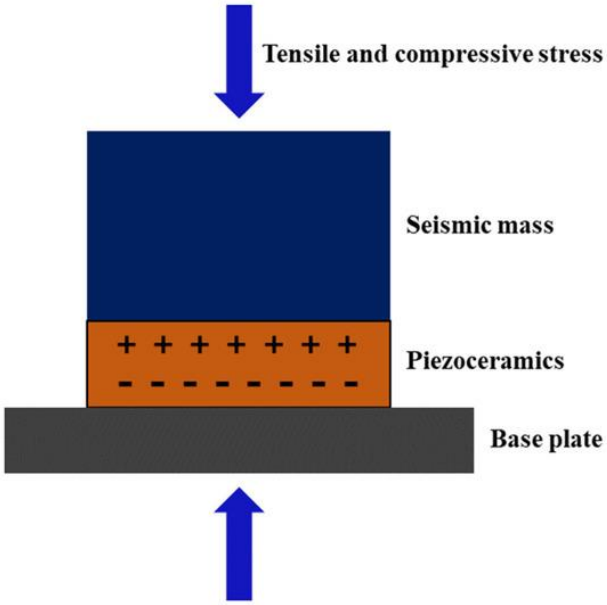


Ilustración 6 - Esquema del principio de funcionamiento del acelerómetro de compresión [12].

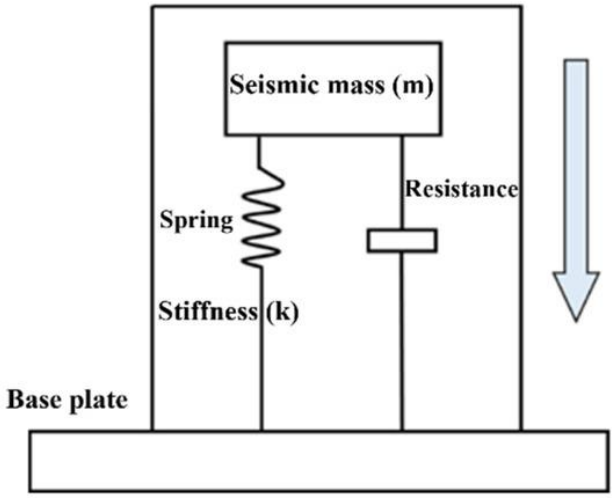


Ilustración 7 - Modelo mecánico simplificado del acelerómetro piezoeléctrico [12].

- **Giroscopio:** es un dispositivo capaz de medir la velocidad angular, es decir, la tasa de rotación que sufre un objeto alrededor de uno o más ejes. A diferencia de los giroscopios mecánicos tradicionales basados en rotores giratorios, en el proyecto se utiliza el giroscopio MEMS. Este funciona basándose en el efecto Coriolis (el efecto Coriolis es una fuerza aparente que actúa sobre un objeto que se mueve dentro de un sistema que está en rotación.) [13]. Dentro del *chip*, una estructura de silicio microscópica es mantenida en un estado de vibración u oscilación constante. Cuando el dispositivo (y por tanto el *chip*) rota, la fuerza de Coriolis actúa sobre esta masa vibrante, induciendo una vibración secundaria en una dirección perpendicular. La magnitud de esta vibración secundaria es directamente proporcional a la velocidad angular de la rotación. Unos sensores capacitivos detectan este desplazamiento y lo convierten en una señal eléctrica que representa la velocidad de giro. Al ser triaxial, mide la rotación en los ejes de balanceo (*roll*), cabeceo (*pitch*) y guiñada (*yaw*) [13], [14].

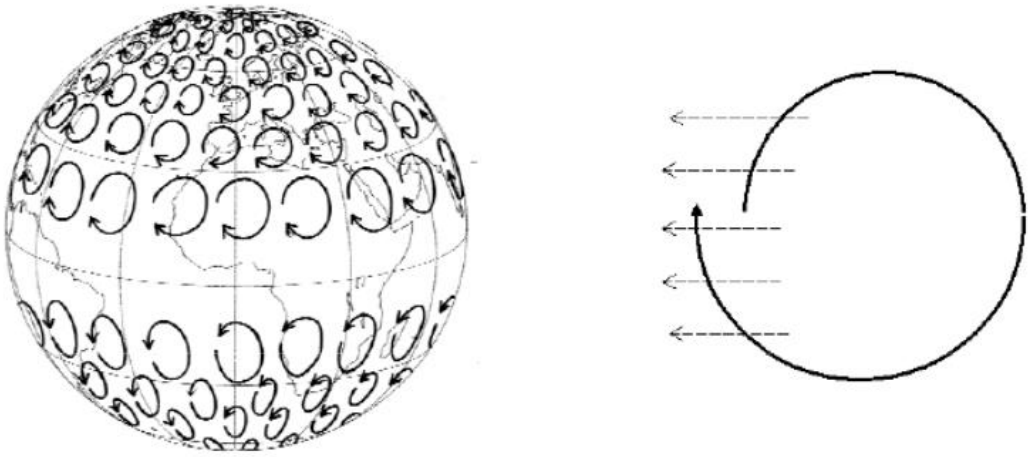


Ilustración 8 - Manifestación a Escala Planetaria del Efecto Coriolis [13].

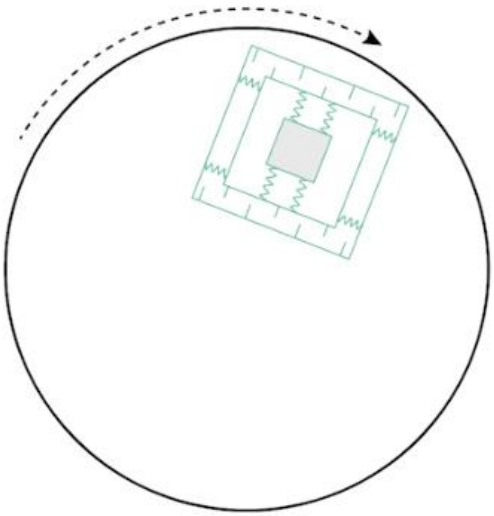


Ilustración 9 - Principio de Funcionamiento de un Giróscopo Vibratorio MEMS [14].

3.4. Detección de caídas a través de algoritmos de inteligencia artificial (IA)

La IA, en concreto el aprendizaje automático supervisado, permite crear modelos capaces de reconocer patrones y predecir comportamientos después de ser entrenados con grandes *datasests* (conjuntos de datos etiquetados) [15]. Dado el proyecto a realizar, el objetivo principal es realizar un modelo de predicción capaz de distinguir entre Actividades de la Vida Diaria (AVD), como podría ser caminar, sentarse, agacharse, etc., y situaciones de pérdida de equilibrio o caída. Para lograr esto, el sistema se entrena utilizando miles de datos previamente recogidos con la misma o distinta (MPU) estos datos ya están etiquetados dependiendo del tipo de secuencia indicando si hay "caída" o "no caída".

Existen diferentes tipos del algoritmo/técnicas de clasificación que se podrían usar para la realización del proyecto. Algunas de las más relevantes son:

- **Regresión logística:** es un algoritmo de clasificación que modela la probabilidad de que una muestra sea perteneciente a una clase determinada. Es simple y rápido en cuanto a rendimiento, pero puede tener dificultades con relaciones no lineales complejas, no siendo el idóneo para el proyecto [16].
- **Máquinas de Soporte Vectorial (SVM):** este algoritmo se busca encontrar un hiperplano permita separar de la mejor forma posible las clases en el espacio de características. Son muy efectivas en espacios de alta dimensión, normalmente son computacionalmente costosas y complejas de implementar, siendo automáticamente descartadas para el proyecto [17].
- **Árboles de Decisión:** Modelo predictivo que utiliza una estructura de árbol para tomar decisiones basadas en las características de los datos. Son muy eficientes e interpretables ("caja blanca"), lo que facilita su implementación en sistemas con recursos limitados como el hardware de este proyecto.
- **Aprendizaje por ensamblado (*Ensemble Learning*):** esta metodología no se basa en un único modelo, sino que contempla las predicciones de múltiples modelos más simples para poder obtener una predicción final robusta y precisa. Dentro de esta familia se encuentran algoritmos como *Random Forest* y *Gradient Boosting*, que son especialmente potentes para problemas de clasificación complejos como el planteado para el proyecto y no muy exigentes en cuanto recursos [18].

Para poder evaluar cuantitativamente el rendimiento de un modelo de clasificación, no es suficiente con el número de aciertos o fallos, sino que es necesario utilizar un conjunto de métricas estandarizadas. Estas métricas se calculan a partir de lo que se denomina matriz de confusión, esta resume el número de aciertos y fallos del modelo. La matriz se compone de cuatro valores: Verdaderos Positivos (VP), Verdaderos Negativos (VN), Falsos Positivos (FP) y Falsos Negativos (FN). A partir de ellos, se definen las siguientes métricas clave:

- **Exactitud:** mide la proporción de predicciones acertadas sobre el total. Es un factor intuitivo, pero puede ser engañoso si las clases están desbalanceadas (hay

una gran desproporción entre "no caída" y "caída" produciendo sesgo en la predicción).

$$Exactitud = \frac{VP + VN}{VP + VN + FP + FN}$$

- **Precisión:** de todas las predicciones que realizó el modelo como "caída", es el número de veces que acertó. Una alta precisión es importante para no generar falsas alarmas en la predicción.

$$Precisión = \frac{VP}{VP + FP}$$

- **Sensibilidad:** de todas las "caídas" reales que se produjeron, es el número de ellas que fue capaz de detectar el modelo. Esta métrica es muy necesaria para el proyecto, ya que un Falso Negativo (no detectar una caída real) tiene consecuencias mucho más graves que una falsa alarma en el proyecto.

$$Sensibilidad = \frac{VP}{VP + FN}$$

- **Puntuación F1:** de trata de la media armónica entre la Precisión y la Sensibilidad, este parámetro ofrece el balance entre ambas. Es muy útil cuando existe un desequilibrio de clases (algo que hay que evitar para no tener sesgos en el modelo).

$$Puntuación\ F1 = 2\ x \frac{Precisión\ x\ Sensibilidad}{Precisión + Sensibilidad}$$

Finalmente, la arquitectura seleccionada se basa en el aprendizaje por ensamblado (*Ensemble Learning*). Aunque se inspira en la robustez de algoritmos como *Random Forest* (que combinan cientos de árboles de decisión), se desarrolló una solución de ensamble ligero a medida para garantizar la eficiencia en el microcontrolador.

Finalmente, la arquitectura seleccionada se basa en un ensamble similar a un *Random Forest* ligero diseñado a medida. La solución consiste en utilizar dos árboles de decisión heterogéneos, entrenados de forma independiente para que se especialicen en detectar diferentes patrones del movimiento. Las predicciones de ambos se combinan en tiempo real en el *firmware* mediante un sistema de votación por acumulación, aprovechando así la diversidad de los modelos para lograr una detección final más robusta y fiable que la que ofrecería un único árbol.

Cada árbol funciona como un sistema de decisiones secuenciales: comienza en un nodo raíz (tronco), se bifurca en nodos internos (ramas) según características específicas de los datos, y culmina en nodos hoja que representan la clasificación final ("caída" o "no caída").

La potencia del *Random Forest* radica en dos mecanismos clave que introducen aleatoriedad controlada durante el entrenamiento. Primero, cada árbol se entrena con un sub-conjunto aleatorio de datos de entrenamiento (técnica conocida como *bagging* o *bootstrapping*). Segundo, en cada nodo de división solo se considera un subconjunto aleatorio de las características disponibles. Para obtener la predicción final, todos los árboles del bosque "votan" individualmente, la clase con más votos gana (estrategia de votación por mayoría) estableciéndose como resultado de la predicción. Este enfoque en conjunto basado en el principio de "divide y vencerás" reduce el sobreajuste (*overfitting*), evitando que árboles individuales memoricen ruido en los datos, además también aumenta la robustez del modelo al combinar múltiples puntos de vista de cada árbol. Estas características lo hacen especialmente adecuado para analizar señales complejas, en este caso de sensores inerciales, donde la diferenciación entre caídas y actividades de la vida cotidiana requieren capturar patrones no lineales y relaciones sutiles entre variables.

3.5. Tecnología de radiofrecuencia *LoRa*, protocolos *LoRaWAN* e *ISHA*

LoRa es una tecnología patentada que permite la comunicación con uno o más dispositivos en rango, a través de comunicación física de radio [19]. *LoRa* opera con un bajo coste energético llegando a tener un largo alcance de acción, de ahí su nombre *LoRa* (*Long Range*). Sus principales características, lo hacen bastante atractivas para el proyecto. Esta tecnología fue diseñada específicamente para comunicar dispositivos IoT (*Internet of Things* o Internet de las Cosas) [20] los cuales demandan transmisión de datos de forma eficiente a distancias mucho más superiores a las que puede ofrecer las tecnologías convencionales como podrían ser el *WiFi* o el *bluetooth. LoRa* funciona a través de modulación inalámbrica de espectro ensanchado (técnica de modulación principalmente utilizada para la transmisión de datos por el medio).

LoRa utiliza una técnica llamada *Chirp Spread Spectrum* (Espectro Ensanchado por “Chirrido”), que codifica la información en "chirridos" (señales que varían su frecuencia en el tiempo). Esta técnica permite que los datos transmitidos sean más resistentes al ruido. Esto permite su correcta recuperación incluso con interferencias. Además, consta de un gran alcance. Siendo este de unos 15-20 kms en zonas rurales y 2-5 kms en entornos urbanos (todo esto bajo condiciones óptimas). Para entender la arquitectura de red, es fundamental distinguir que *LoRa*, solo se refiere a la capa física (a la mera transmisión de información por el medio en este caso utilizando radio) a partir de esto, existen protocolos estándares como sería *LoRaWAN*. En el contexto del proyecto se ha desarrollado un protocolo propio para poder abordar la solución de manera específica al problema planteado, permitiendo tener un mayor control de las transmisiones.

LoRaWAN es el protocolo de red de estándar abierto que opera sobre la capa física de *LoRa*. Este protocolo define la arquitectura completa del sistema y gestiona aspectos complejos como el registro de dispositivos en la red, el direccionamiento, la seguridad de extremo a extremo y la adaptación de la velocidad de transmisión.

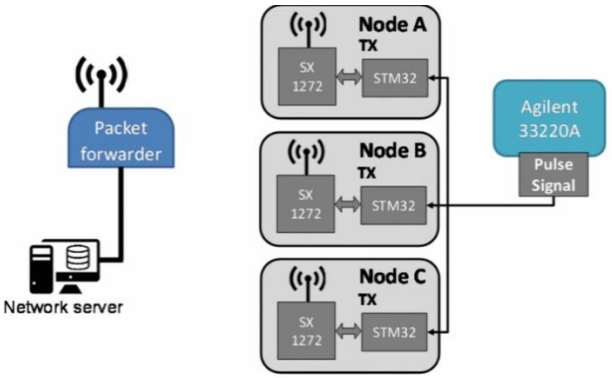


Ilustración 10 - Arquitectura de Pruebas para la Evaluación de una Red LoRaWAN en entornos industriales [21].

A pesar de la existencia del estándar *LoRaWAN* [22], para este proyecto tal y como se comentó anteriormente, se tomó la decisión estratégica de diseñar un protocolo de comunicación a medida, denominado *ISHA* (*Inter-System Hybrid Alerting*). A continuación, se describe su funcionamiento y arquitectura.

El protocolo *ISHA* define un ecosistema híbrido con tres componentes principales que operan en una topología de estrella:

- **Dispositivo de Usuario (DU):** es el dispositivo portátil que lleva el usuario. Equipado con un microcontrolador ESP32, un sensor MPU6050 y un *GPS*, su función es monitorizar la actividad, detectar caídas y comunicarse exclusivamente vía *LoRa* con una Antena/*Gateway*.
- **Antena/*Gateway* (AG):** es un dispositivo fijo que actúa como un puente inteligente. Recibe los mensajes *LoRa* de los DUs y, a diferencia de un *gateway LoRaWAN* pasivo, se autentica contra el servidor y retransmite la información de forma segura a través de una red WiFi mediante peticiones HTTP, desconociendo el contenido de las tramas ya que están cifradas punto a punto (Dispositivo de Usuario – Servidor).
- **Servidor *Backend* (SB):** es el coordinador central del sistema. Implementado con *FastAPI* y conectado a una base de datos, gestiona la autenticación de las AGs, decodifica y procesa los datos de los DUs, y ejecuta la lógica de negocio, como el envío de notificaciones de emergencia.

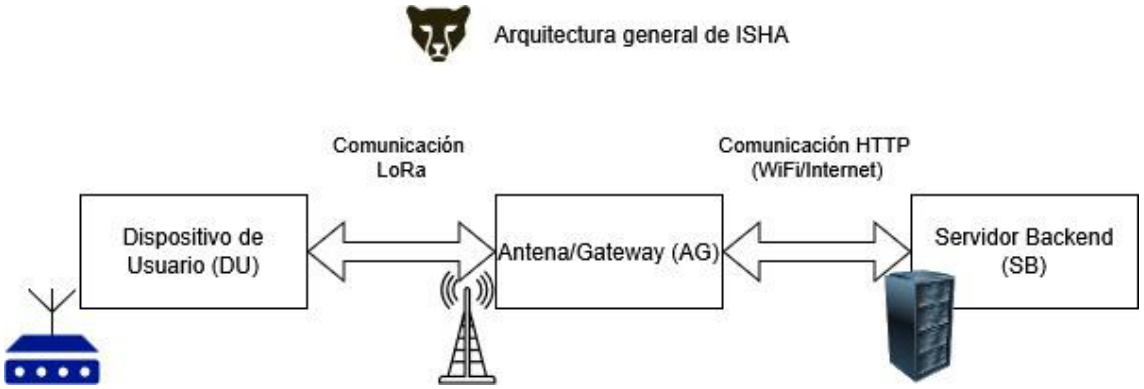


Ilustración 11 - Arquitectura alto nivel del protocolo ISHA.

Esta elección se demuestra las optimizaciones realizadas en el nuevo protocolo para poder abordar de forma óptima los requisitos únicos del sistema de emergencias:

- **Control absoluto de la seguridad**: se buscaba un dominio total sobre el esquema criptográfico, implementando claves simétricas de rotación diaria y vectores de inicialización únicos por trama. Esto permite un sistema de seguridad autocontenida y determinista, sin depender de la infraestructura de servidores externos (*join servers*) que requiere *LoRaWAN* añadiendo complejidad al sistema.
- **Optimización y eficiencia**: el protocolo *ISHA* está diseñado exclusivamente para la transmisión de paquetes de alerta y baliza. Esto elimina la sobrecarga (*overhead*) de los complejos mecanismos de gestión de *LoRaWAN* (clases de dispositivos, ventanas de recepción, etc.), resultando en una comunicación más directa y eficiente para este caso de uso crítico.

Arquitectura simplificada: se optó por una arquitectura donde las "antenas/*gateways*" actúan como pasarelas ligeras y transparentes. Su única función es reenviar los paquetes *LoRa* cifrados a través de TCP/IP, centralizando toda la lógica de negocio (descifrado, cifrando, etc.) en el *backend*. Esto simplifica el *hardware*, facilita el despliegue y reduce el coste de mantenimiento de la infraestructura en campo. Además, garantiza el cifrado punto a punto (dispositivo de usuario – servidor). El objetivo del protocolo *ISHA* no es intentar reemplazar el estándar *LoRaWAN*, sino realizar una solución de ingeniería adaptada y optimizada para maximizar la fiabilidad, seguridad y eficiencia dentro del contexto específico del proyecto.

3.6. Sistema de Posicionamiento global (*GPS*)

El Sistema de Posicionamiento Global (*GPS*) [23], [24] es una red de satélites que están orbitando continuamente alrededor de la Tierra. Estos satélites emiten señales continuas que son captadas por los receptores *GPS*. Las señales recibidas de los satélites contienen información identificativa del mismo, su posición, el instante de tiempo preciso que fue

enviada (pudiéndose sincronizar la fecha y hora ya que los satélites *GPS* constan de relojes atómicos a partir de un proceso de trilateración[1]).

El receptor es capaz de calcular su longitud y latitud (coordenadas que representan su posición física en el sistema terrestre) a partir de la intersección de la señal de al menos 3 satélites, de ahí el termino trilateración. Aunque este proceso solo sirve para posicionar a un dispositivo en un espacio bidimensional. por ello, para poder obtener la altitud necesitaríamos contar al menos con la información de un satélite más.

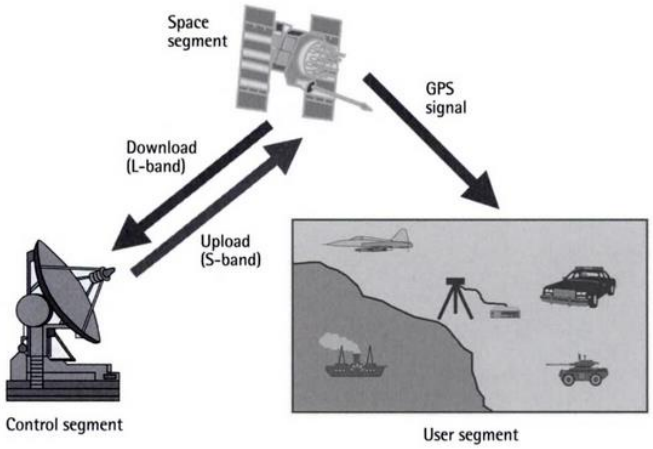


Ilustración 12 - Los tres segmentos del Sistema de Posicionamiento Global (GPS) [24].

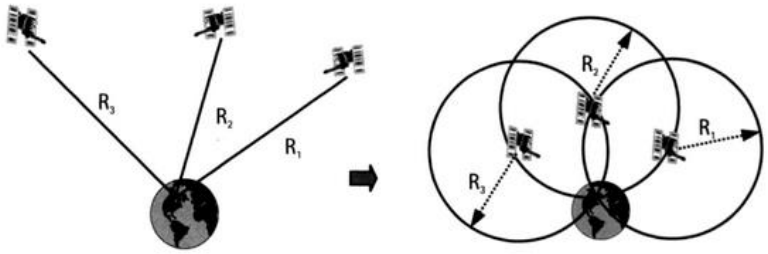


Ilustración 13 - Principio de trilateración en GPS [24].

Dado las características proyecto, la utilización de este sistema de geoposicionamiento es indispensable ya que poder conocer las coordenadas exactas de la persona que ha sufrido una caída marca la diferencia entre una respuesta eficaz y rápida o una tragedia.

3.7. Fundamentos criptográficos

Para garantizar la seguridad del sistema (protocolo *ISHA*), se ha utilizado las siguientes tres primitivas criptográficas.

- **Cifrado simétrico (AES):** es un algoritmo de cifrado simétrico, es decir, utiliza la misma clave secreta durante el proceso de cifrado (mantener la información, pero protegerla haciendo que sea indistinguible), protegiendo así la sección privada de la trama. En otras palabras, es similar a un candado, donde se utiliza la

[1] Principio geométrico para determinar una posición midiendo las distancias a tres o más puntos de referencia. La posición exacta es la intersección de las esferas definidas por los puntos de referencia como centros y las distancias medidas como radios. Es la base del funcionamiento del sistema de posicionamiento global (*GPS*).

misma llave para abrirlo o cerrarlo. Se ha utilizado la modalidad de 256 bits (esto define el tamaño de clave/llave utilizada) cifra en bloques de 16 *bytes* (tamaño justo de la parte privada de la trama). Este algoritmo es un estándar, dada su eficiencia computacional y seguridad [25].

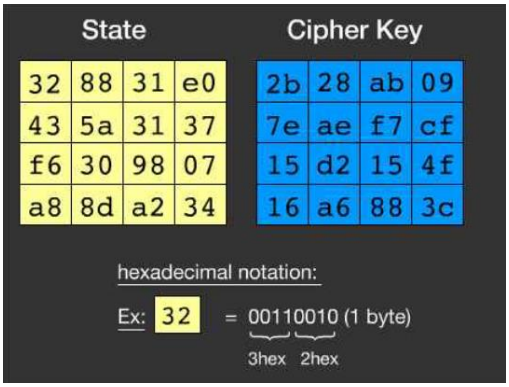


Ilustración 14 - Representación Matricial del Estado y la Clave en el Cifrado AES [25].

- **Funciones *hash* (SHA-256):** las funciones *Hash*/Huella, son funciones matemáticas las cuales permiten crear un "resumen" de longitud fija, a partir de un conjunto de datos. El proceso solo es unidireccional (ya que computacionalmente es inviable realizar el proceso inverso). Este sistema es utilizado dentro del proyecto para crear claves de cifrado diarias de forma determinista. Aunque, convencionalmente, su uso principal es verificar la integridad de los datos contra manipulaciones [25].

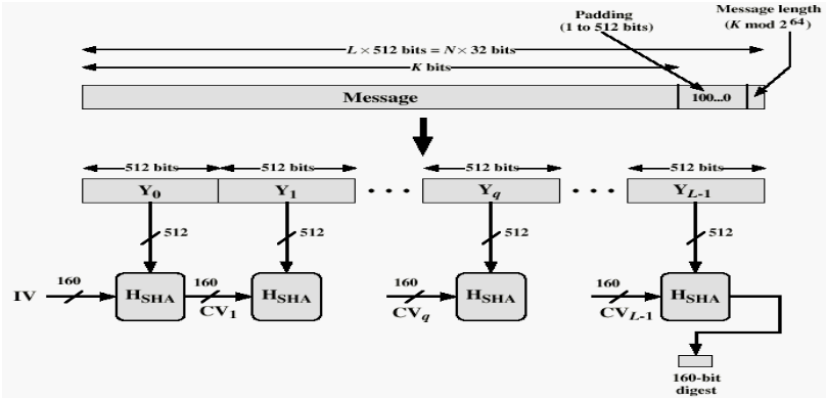


Ilustración 15 - Proceso de Generación de un Hash con el Algoritmo SHA.

- **Código de redundancia Cíclica (CRC):** el CRC es un código de detección de errores. Su objetivo es verificar la integridad de los datos después del proceso de transmisión. Su utilización en el proyecto es debida a que requiere menos recursos que las funciones *hash* pudiendo utilizarlo en la verificación de las tramas, aunque no garantiza el mismo nivel de seguridad [26].

4. Técnicas y herramientas

En esta sección se va a entrar en detalle sobre las técnicas y herramientas utilizadas a lo largo del proyecto.

4.1. Persistencia de datos.

A la hora de elegir una solución tecnológica que permita almacenar los datos del proyecto, se barajaron cinco opciones que en principio se adaptaban a las necesidades del proyecto. Dos de estas soluciones, Firebase y Supabase, disponen de servicios *BaaS* (*Backend-as-a-Service*), lo que permite una mayor funcionalidad aparte de la persistencia permitiendo crear aplicaciones simples, utilizando su sistema de *backend*, sin necesidad de añadir la capa externa por parte del programador. Las otras tres tecnologías de persistencia barajadas fueron MongoDB (base de datos no relacional), PostgreSQL y MariaDB.

A continuación, se realiza una descripción de las opciones mencionadas. Por un lado, compararemos las tecnologías de persistencia más "conservadoras", MongoDB, PostgreSQL y MariaDB, (ya que no ofrecen funcionalidad extra más allá de la mera persistencia de los datos) y las más "progresistas", Firebase y Supabase. Justificando finalmente el porqué de la elección de Supabase como capa de persistencia.

Solución "conservadora" (PostgreSQL, MariaDB, MongoDB)	
Descripción	Estas soluciones son autogestionadas, configuradas e instaladas por el desarrollador. • **PostgreSQL:** es un sistema de gestión de bases de datos objeto-relacional de código abierto, sigue los estándares SQL. Es bastante conocido en la industria [27]. • **MariaDB:** es un sistema gestor de bases de datos relacionales de código abierto derivado del famoso MySQL [28]. • **MongoDB**: es un sistema de gestión de bases de datos no relacionales/NoSQL, principalmente orientado a almacenar colecciones de datos flexibles [29].
Ventajas	• **Control total y flexibilidad máxima:** el desarrollador tiene libertad absoluta sobre la configuración, optimización, seguridad y arquitectura del sistema. • **Soberanía de los datos:** los datos residen en una infraestructura controlada solo por el desarrollador, conociendo en todo momento dónde se almacenan y cómo se gestionan de forma interna. • **Sin costes adicionales:** las versiones de código abierto de estas bases de datos son completamente gratuitas y de libre uso. • **Adaptabilidad del modelo:** permite elegir el modelo de datos más adecuado, ya sea relacional (PostgreSQL, MariaDB) o no relacional (MongoDB), según los requisitos del proyecto.
Desventajas	• **Mayor carga de trabajo para el desarrollador**: el desarrollador es el único responsable de la instalación, configuración, copias de seguridad, escalado y seguridad de todo el sistema.

	• **Desarrollo más lento:** funcionalidades esenciales como la autenticación de usuarios, la gestión de sesiones, las políticas de acceso o el almacenamiento de archivos deben ser implementadas desde cero por el programador. • **Coste de infraestructura:** aunque el *software* sea gratuito, se necesita un servidor operativo 24/7 (VPS, en la nube, etc.), lo cual implica un coste recurrente.

Tabla 1 - Solución Conservadora de Persistencia.

Firebase solución "progresista" propietaria	
Descripción	Es la plataforma *BaaS* más extendida del mercado. Propiedad de *Google*, proporciona múltiples herramientas y servicios facilitando tareas al desarrollador permitiéndole disminuir el tiempo de desarrollo, sin necesidad de tener que alojar una infraestructura propia [30].
Ventajas	• **Ecosistema completo y maduro:** ofrece una solución completa para autenticación, diferentes tipos de bases de datos (*Firestore, Realtime Database*), *hosting*, notificaciones, etc. Además, lleva operando en el mercado durante bastante tiempo. • **Gran escalabilidad:** la infraestructura está gestionada por *Google*, garantizando un alto rendimiento y disponibilidad. • **Buena documentación y comunidad:** al ser tan popular entre programadores, cuenta con infinidad de tutoriales y una comunidad muy activa para resolver dudas sobre sus servicios. • **Plan gratuito generoso:** su plan gratuito es ideal para prototipos y proyectos pequeños.
Desventajas	• **Fuerte dependencia:** utiliza tecnologías meramente propietarias, lo que hace muy complejo y costoso migrar a otra plataforma en el futuro prácticamente obligando al desarrollador a seguir utilizándolas durante todo el ciclo de vida el proyecto. • **Menor control y transparencia:** al ser un servicio de totalmente propietario, se tiene menos visibilidad y control sobre el funcionamiento interno y la disposición en la que se almacena la información. • **Costes impredecibles:** en proyectos con mucho tráfico o datos, los costes pueden escalar muy rápidamente obligando al desarrollador a invertir más dinero para mantener su aplicación a flote.

Tabla 2 - Firebase Solución Progresista propietaria.

Supabase (BaaS) - Solución Elegida

Supabase solución "progresista" libre elegida	
Descripción	Esta fue la solución elegida, ya que proporciona una plataforma desarrollada con tecnologías de código abierto que reducen drásticamente el tiempo de configuración y desarrollo, sin sacrificar el control a largo plazo. Dando la flexibilidad de utilizar sus servicios propios de *hosting* al igual que lo hace Firebase o pudiendo ser "*autohosteada"* [31].
Ventajas	• **Basado en PostgreSQL:** es su principal ventaja. Ofrece toda la potencia y fiabilidad de una base de datos relacional ya madura en el mercado. Permite exportar la base de datos completa y migrarla a un servidor propio en cualquier momento, eliminando el riesgo de *vendor lock-in* que podría existir en Firebase. • **Código abierto y transparencia:** aporta total transparencia y ofrece la posibilidad de autoalojar (*self-hosting*) la pila completa de Supabase en una infraestructura propia permitiendo al programador no depender de servicios externos y manteniendo la flexibilidad que da un BaaS. • **Generación automática de *APIs*:** crea automáticamente una API RESTful segura sobre el esquema de la base de datos, lo que acelera enormemente el desarrollo del *backend*. • **Servicios integrados:** proporciona de serie un sistema de autenticación completo, almacenamiento de archivos y funciones en la nube sin necesidad de mucha intervención por parte del programador. • ***APIs* para Python y JavaScript:** ofrece bibliotecas cliente oficiales para los lenguajes utilizados en el proyecto (Python para el *backend* y JavaScript para el *frontend*), facilitando la integración a su sistema. • **Plan gratuito funcional:** su capa gratuita es más que suficiente para el desarrollo y despliegue de este proyecto sin generar costes adicionales, aunque se ha seleccionado un plan superior para la realización de este proyecto.
Desventajas	• **Ecosistema menos maduro que Firebase:** al ser una plataforma más nueva, su comunidad de desarrolladores y la cantidad de recursos de aprendizaje de momento no son tan extensos como podría ser la de Firebase. • **Límites en la capa gratuita:** aunque funcional, la capa gratuita tiene límites de recursos. Si se superan dicho limites, el proyecto puede sufrir problemas serios de rendimiento hasta que se aumente las especificaciones del plan u se "autohostee".

Tabla 3 - Supabase Solución "Progresista" libre elegida.

El servicio de persistencia seleccionado ha sido Supabase ya que aparte de proporcionar mayor funcionalidad que las alternativas convencionales, es de código abierto pudiendo migrar la infraestructura a servicios propios sin la necesidad de tener que invertir tanto dinero como podría pasar con Firebase. Además, al ser de código abierto y poderse automantener da poder absoluto al programador sobre su persistencia de datos. A su vez, mantiene la comodidad del *BaaS* que será principalmente usado para la realización del

panel *web* con la finalidad de poder realizar operaciones *CRUD* (*Create, Read, Update and Delete*) sobre los datos [32].

Ilustración 16 - Logo de Supabase [33].

4.2. Python y *FastAPI* para la realización del *bakend* de dispositivos.

A la hora de seleccionar tecnologías para la realización del *backend*, no ha existido prácticamente ninguna duda. El principal motivo de elección de esta fue la experiencia previa que el alumno ganó experimentando con *FastAPI*(*framework)* de forma proactiva durante el desarrollo del grado. Es por ello por lo que se utilizó esta tecnología, ya que le transmitía mayor seguridad y tranquilidad al alumno. De esta manera se decidió finalmente realizar el *backend* de la aplicación con esta tecnología sobre la que ya se tenía experiencia previa.

La elección de *FastAPI* llamó la atención ya que es un *framework* bastante, potente, moderno y fácil de aprender. Además, el hecho de que esté basado en el lenguaje de programación Python aseguran una gran portabilidad y potencia. Estos hechos garantizan que *FastAPI* sea la opción ideal y que, además, permite agilizar el desarrollo del sistema al ser una tecnología con las que el alumno ya había sido expuesto. La elección de *FastAPI*, a pesar de ser un *framework* relativamente moderno, no se basó exclusivamente en la familiaridad del alumno con el mismo, si no que se basó también en un conjunto de ventajas técnicas que lo hacen idóneo para los requisitos del proyecto. *FastAPI* está construido sobre Starlette (para la parte *web*) y Pydantic (para la validación de datos). Gracias a su naturaleza asíncrona (ASGI, *Asynchronous Server Gateway Interface*), ofrece un rendimiento extremadamente alto, comparable al de soluciones más tradicionales como podrían ser los *frameworks* de NodeJS o Go. Esta eficiencia es fundamental para poder procesar las peticiones de las antenas reduciendo la latencia. Por otro lado, disminuye el tiempo de desarrollo significativamente gracias diversas características entre las que podemos destacar:

- **Documentación automática:** la generación de documentación de forma automática a partir del código facilita las pruebas y la integración. Para esto utiliza *Suagger [34]* y *Redoc*
- **Uso de estándares:** utiliza estándares de la industria abiertos como *OpenAPI* y JSON.
- **Validación y serialización de datos robustas:** aunque Python es un lenguaje de programación de tipado débil (las variables pueden cambiar de tipo de dato que contiene) la utilización de los *type hints* de Python y *Pydantic* permite realizar una validación de datos potente y clara, reduciendo errores en tiempo de ejecución y ayudando a mantener la legibilidad y mantenibilidad el código.

- **Ecosistema de Python:** desarrollar en Python otorga acceso inmediato a un ecosistema muy extenso y maduro de librerías en el mercado, permitiendo agilizar tareas como la criptografía (PyCryptodome), manipulación de datos y comunicación con la persistencia (cliente de Supabase).

Si bien la familiaridad previa del autor con estas tecnologías agilizó el desarrollo, la elección se fundamenta en que esta combinación de factores tecnológicos que ofrece *FastAPI* permite el equilibrio óptimo entre rendimiento, agilidad de la implementación, seguridad y robustez totalmente necesarios para un sistema crítico como el presentado en este TFG.

Ilustración 17 - Logo de FastAPI [35].

4.3. *GitHub* y *Git* como control de versiones.

Para el control de versiones del proyecto, se ha utilizado el sistema de control de versiones *Git*, junto a la plataforma de *GitHub*.

La elección de *Git* como controlador de versiones fue dada a causa de su robustez, fácil uso, eficiencia y amplia comunidad. *Git* es un estándar, siendo una herramienta muy potente y flexible. Facilita la gestión del historial de versiones del proyecto, permitiendo al desarrollador crear diferentes ramas para experimentar con nuevas versiones, añadiendo funcionalidades de prueba sin tener el temor de comprometer las versiones estables del proyecto. Además, permite comparar versiones y fusionar cambios de manera sencilla, a través de un sistema de selección de cambios.

Para el almacenamiento del repositorio, se optó por la utilización de la plataforma de *GitHub* frente a otras alternativas dados los siguientes factores.

- **Comunidad extensa:** *GitHub* cuenta con una comunidad de desarrolladores muy extensa y activa, lo que facilita enormemente la solución de incidencias que se pudieran dar.
- **Beneficios a estudiantes:** dado el convenio existente entre *GitHub Education* y la Universidad de Salamanca, se hizo uso de las funcionalidades profesionales que se ofrecen totalmente gratuita a los estudiantes de la comunidad universitaria.
- **Experiencia previa:** el alumno ha realizado uso de la plataforma durante la duración del grado para el control de versiones de diferentes trabajos.

Ilustración 18 - Logo de GitHub [36].

Ilustración 19 - Logo de Git [37].

4.4. *Frontend*: *React* y alternativas

Para el desarrollo de la interfaz gráfica de usuario (GUI, *Graphical User Interface*) de la plataforma *web*, se evaluaron las siguientes alternativas: Angular, Vue.js y *React*. Siendo ***React*** la opción seleccionada.

Angular

Es un *framework* completo desarrollado y mantenido por *Google*, basado en el lenguaje de programación TypeScript.

Angular	
Descripción	Plataforma de desarrollo "todo en uno" que proporciona una estructura diseñada para construir aplicaciones a gran escala con gran facilidad [38].
Ventajas	• ***Framework* completo:** ofrece una serie de soluciones para el enrutamiento, gestión de estado, peticiones HTTP y más. • **Estructura consistente:** impone un patrón de diseño (basado en módulos y componentes) que facilita el trabajo en equipos grandes y la mantenibilidad a largo plazo. • **Tipado estricto:** el uso obligatorio de TypeScript ayuda a prevenir errores en tiempo de desarrollo, ayudando a la legibilidad del código.
Desventajas	• **Curva de aprendizaje elevada:** es considerado el más complejo de los tres, con una gran cantidad de conceptos que aprender. • **Menor flexibilidad:** al ser tan estructurado, deja menos libertad para elegir herramientas o arquitecturas alternativas que puedan adaptarse a soluciones específicas. • **Verbosidad:** generalmente requiere escribir más código para poder lograr las mismas funcionalidades que en *React* o Vue.js.

Tabla 4 – Angular.

Vue.js

Es un *framework* de JavaScript progresivo y altamente conocido por su facilidad de uso y excelente documentación.

Vue.js	
Descripción	Un *framework* bastante versátil pudiendo ser adoptado de manera incremental. A menudo se le considera como un punto intermedio entre *React* (por su gran flexibilidad) y Angular (por su estructura) [39].
Ventajas	• **Curva de aprendizaje suave:** este *framework* es ampliamente considerado el más fácil de aprender, siendo ideal para principiantes en el desarrollo de *frontends* modernos. • **Amplia documentación:** su documentación oficial es muy extensa. • **Alto rendimiento:** ofrece un rendimiento muy competitivo.
Desventajas	• **Ecosistema más pequeño:** aunque es muy completo, su conjunto de librerías y herramientas no es tan extenso como las de *React*.

	• **Menor adopción en la industria:** en comparación con *React*, su presencia en grandes corporaciones es menor, lo que puede influir en la disponibilidad de recursos a largo plazo.

Tabla 5 - Vue.js.

4.4.3. *React* (solución elegida)

React es una librería de JavaScript, desarrollada y mantenida por Facebook (la nueva META), enfocada exclusivamente en la creación de interfaces de usuario interactivas a través de un modelo de componentes reutilizables.

React	
Descripción	*React* es una librería diseñada para la realización de interfaces graficas *web*. Permite construir interfaces complejas a partir de pequeñas piezas reutilizables (componentes). No es un *framework*/marco de trabajo completo, lo que le proporciona una gran flexibilidad [40]
Ventajas	• **Modelo basado en componentes:** facilita la creación de interfaces complejas, promueve la reutilización de código y mejora la mantenibilidad del proyecto. • **Ecosistema inmenso:** es la tecnología con la mayor comunidad activa de las comparadas y con el mayor número de librerías disponibles para cualquier necesidad en cuanto a funcionalidad y presentación. • **Flexibilidad:** al ser una librería y no un *framework*, da una libertad total al desarrollador permitiéndole seleccionar la arquitectura y las herramientas que mejor se adapten a cada proyecto en específico. • **Popularidad:** es la librería de *frontend* más demandada en el mercado laboral, lo que asegura una gran cantidad de recursos y soporte por parte de la comunidad a través de los foros.
Desventajas	• **No es una solución "todo en uno":** requiere integrar librerías de terceros para añadir funcionalidades que un *framework* consolidado podría proporcionar (como el enrutamiento o la gestión de estado), lo que puede suponer una mayor complejidad durante la configuración inicial. • **Barrera de entrada:** su sintaxis JSX, que mezcla HTML y JavaScript, puede ser un poco confusa al principio, pudiendo requerir un periodo de adaptación para nuevos desarrolladores.

Tabla 6 - React Solución Elegida.

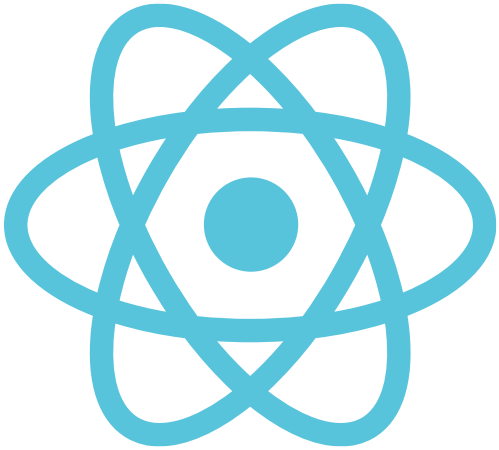


Ilustración 20 - Logo de React [41].

4.5. Entorno de desarrollo para *firmware*: *Arduino IDE* y *PlatformIO*

Para el desarrollo del *firmware* de dispositivos, se valoraron los dos principales ecosistemas disponibles para la realización de programa para el microcontrolador ESP32. Para ello se estuvo barajeando las siguientes opciones: PlataformIO IDE y *Arduino IDE*.

La solución finalmente seleccionada fue *Arduino IDE*. No obstante, a continuación, se realiza una descripción de las opciones planteadas y la justificación de su elección de *Arduino IDE*.

PlatformIO IDE

PlatformIO es una extensión para el editor de código abierto *Visual Studio Code* (*VS Code*), permite añadir un ecosistema orientado a sistemas embebidos al editor *VS Code.*

	PlatformIO
Descripción	Entorno de desarrollo orientado a la industrial, dependiente a la placa de desarrollo o al *framework.* Proporciona herramientas avanzadas para la creación de proyectos de IoT industriales.
Ventajas	• **Integración con VS Code:** permite utilizar como base el editor VS Code, pudiendo utilizar sus herramientas como el autocompletado avanzado (*IntelliSense*). • **Gestor de dependencias:** ponsta de un archivo de configuración que permite el control preciso de las versiones de las librerías utilizadas, garantizando la replicación del proyecto en otros equipos. • **Soporte multi-placa:** facilita el trabajo de desarrollo, siendo compatible con diferentes arquitecturas de microcontroladores, pudiendo cambiar de arquitectura fácilmente • **Herramientas profesionales:** incluye de serie funcionalidades para la realización de pruebas y análisis de código.
Desventajas	• **Curva de aprendizaje pronunciada:** requiere un tiempo de adaptación para comprender el funcionamiento y configuración del ecosistema. • **Complejidad innecesaria para prototipos:** para la realización de proyectos prototipo rápidos, su configuración puede resultar muy excesiva y compleja.

Tabla 7 – PlatformIO.

***Arduino IDE* (entorno elegido)**

El *Arduino IDE* es el entorno de desarrollo diseñado para el proyecto Arduino, es accesible y fácil de usar, especialmente para principiantes y para el prototipado rápido.

	Arduino IDE
Descripción	Es un entorno de desarrollo especializado y simplificado, que incluye un editor de código, herramientas de compilación (gcc), herramientas de carga del *firmware*, gestor de librerías, etc.

Ventajas	• **Fácil de usar:** su facilidad de uso es unos de sus puntos fuertes, siendo la curva de aprendizaje prácticamente nula para programadores con conocimientos básicos en C/C++, permitiendo empezar a programar en cuestión de minutos. • **Gran comunidad y recursos:** tiene una gran cantidad de tutoriales, ejemplos y librerías disponibles para casi cualquier sensor o módulo siendo compatibles entre placas. • **Ideal para prototipado:** su facilidad de uso permite validar ideas y desarrollar prototipos funcionales de manera muy ágil. • **Experiencia previa:** la familiaridad del autor con este entorno, adquirida en asignatura de "Periféricos" durante el desarrollo del grado, fue un factor clave para poder acelerar el desarrollo del *firmware*.
Desventajas	• **Editor limitado:** carece de las funcionalidades avanzadas de un IDE moderno (*IntelliSense dinamico*). • **Gestión de dependencias básica:** en proyectos con múltiples librerías, puede resultar difícil gestionar las versiones de estas y la resolución de conflictos. • **Orientado a la educación y al prototipado:** no está diseñado para proyectos de *software* industriales, más bien al sector académico.

Tabla 8 - Arduino IDE.

Aunque *PlatformIO* ofrece un entorno más potente y profesional, no es conveniente para la realización de un prototipo como el realizado en este TFG. La simplicidad y la rapidez que ofrece el *Arduino IDE*, sumadas a la experiencia previa del autor, resultaron ser los factores decisivos para su selección. El entorno proporcionó todas las capacidades necesarias para desarrollar el *firmware* de manera eficiente sin introducir una capa de complejidad adicional en la gestión del proyecto.

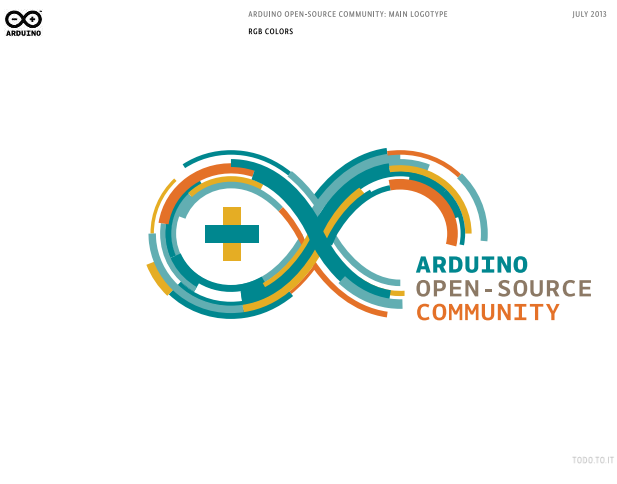


Ilustración 21 - Logo de Arduino [42].

4.6. Herramientas de ofimática, planificación y diseño

Para la realización del proyecto, además de las herramientas técnicas involucradas en el desarrollo de *software* descritas anteriormente, se han utilizado otra serie de herramientas para la creación de circuitos, modelos 3D, la redacción de este mismo documento, etc. A continuación, se mencionarán estas mismas y el porqué del uso de estas.

4.6.1. Suite de ofimática y documentación

Microsoft Word 365

La herramienta seleccionada para la redacción, formateo y maquetación de la documentación técnica del proyecto (de esta memoria y los anexos técnicos específicos) fue Microsoft Word 365. El motivo de su elección fue su madurez en el mercado y el libre uso del mismo por parte de la comunidad universitaria, gracias a los acuerdos que tiene la Universidad de Salamanca con Microsoft. Por este motivo se utilizó esta potentísima herramienta, que además se trata de un estándar en la industria.

4.6.2. *Software* de gestión y planificación de proyectos

Microsoft Project

Esta herramienta fue seleccionada, al igual que Microsoft Word 365, ya que al formar parte de la comunidad universitaria se permite el acceso y uso de la misma de forma totalmente gratuita. Además, el alumno ya tenía experiencia previa en la utilización de la misma gracias a las prácticas realizadas en la asignatura de gestión de proyectos. Esta herramienta, potentísima, permite realizar gestiones del trabajo y organizarlo de mejor forma, pudiendo aumentar el rendimiento del equipo. Esto se puede realizar visualizando los denominados diagramas de Gantt, cronogramas, etc., entre otras funcionalidades que ofrece. Se hizo uso de este *software* para la realización del Anexo I.

EZEstimate

Herramienta también proporcionada por la universidad y conocida por el estudiante tras la realización de prácticas con la misma durante el desarrollo de la asignatura de gestión de proyectos. Esta herramienta permite simplificar los cálculos matemáticos necesarios para poder alimentar el modelo de la estimación del esfuerzo (para mayor detalle, consúltese el Anexo I).

4.6.3. *Software* de modelado y diseño

Software de modelado UML (PlantUML)

Este *software* fue nuevo para el alumno, ya que nunca antes había trabajado con este estándar de UML en texto plano. Esto fue a causa de la caducidad de la licencia proporcionada al alumno del programa que utilizó durante el desarrollo del grado (Visual Paradigm), no pudiendo acceder a dicho *software* de forma lícita, por lo que se descartó directamente. Esta alternativa a Visual Paradigm fue propuesta por uno de los tutores de este mismo proyecto. La herramienta permite realizar diagramas UML a partir de texto, siendo más cómodo a la hora de realizar tareas de modificación. Simplemente instalando una extensión en el editor Visual Studio, el diagrama en texto se interpreta en tiempo real. Fue por ello que la mayoría de los diagramas UML presentes en el proyecto están realizados con esta herramienta.

Software de diseño de circuitos (Fritzing):

Este *software* de código abierto llamó bastante la atención, ya que permitía la realización esquemática de prototipos de una manera muy simple e intuitiva. Para poder descargar la versión del programa, aunque se trate de *software* libre, piden un donativo de 8 o 25 dólares, que se realizó para la obtención del mismo y apoyar al proyecto. Este *software* fue utilizado para la creación del diagrama a alto nivel de los componentes hardware que conformarían el dispositivo y el posterior diseño de la PCB realizada para el proyecto.

Software de edición y modelado 3D (TinkerCad)

La carcasa del dispositivo, finalmente, después de dos intentos, se realizó a través del diseño 3D. La selección de este *software* se debió a su coste gratuito y a su no muy alta curva de aprendizaje. Es una herramienta de CAD (*Computer-Aided Design en* español Diseño Asistido por Computadora*)* simplificada con objetivos pedagógicos. Es utilizada para introducir a las personas de a pie a este tipo de herramientas de ingeniería. A causa de que el modelado 3D no permitía opciones muy potentes, solo a través de la unión y sustracción, se complicó un poco el proceso de realización de la carcasa. Para mayor detalle, consúltese el Anexo VI.

Entornos de desarrollo integrado (IDE)

Además de los entornos mencionados anteriormente para la realización del *firmware*, se seleccionó el entorno de desarrollo de Visual Studio Code, siendo el editor principal del proyecto. Esto se debe a su coste gratuito, es de código abierto, y su gran extensibilidad y adaptabilidad a través de *plugins*, lo convirtieron en la herramienta idónea para la realización del proyecto.

5. Aspectos relevantes del desarrollo

En la siguiente sección, se centrará en comentar los aspectos más relevantes que el autor ha considerado durante todo el proceso de desarrollo del sistema de una forma simplificada y superficial. Para mayor detalle se puede consultar los anexos específicos correspondientes.

5.1. Desarrollo *hardware*

Para el desarrollo de los dispositivos *hardware* (tanto antenas como dispositivos de usuario comparten *hardware*) se han definido a partir de las necesidades y especificaciones del proyecto, para la selección de componentes se tuvo en cuenta la funcionalidad deseada, lo establecido que está el componente en el mercado (garantizando su existencia a futuro y resolución de incidencias) y su coste.

Siendo los componentes seleccionados los siguientes:

- **ESP32 - 30 PIN:** microcontrolador del dispositivo.
- **MPU 6050:** unidad de Procesamiento de Movimiento.
- **NEO-M8N-010:** receptor *GPS*.
- **LED RGB:** indicador visual de estados.
- **Resistencias de 220 Ohmios:** como protección para el LED RGB.
- **Pulsador:** entrada física al aparato.
- **Buzzer 3.3V:** indicador sonoro.
- ***LoRa* SX1276:** módulo de telecomunicaciones por radiofrecuencia.
- **Condensador de 220 µF a 10 V:** instalado en el módulo *lora* para su correcto funcionamiento)
- **MCP73871:** controlador de carga y alimentación del dispositivo.
- **Baterías INR 18650**: baterías seleccionadas para la realización del proyecto.
- **Conector magnético de carga macho y hembra:** utilizado para simplificar el uso del dispositivo.
- **USB-C hembra:** utilizado como entrada de alimentación en la base de carga.
- **DC-BOOST:** utilizado para regular el voltaje en el circuito de alimentación del dispositivo.

Dado que el prototipo montado en las placas de prototipado sin soldadura tenía tamaños excesivos, llegando a ocupar 1 placa de prototipado grande y una mediana, se decidió realizar una PCB (Circuito impreso) con el objetivo de miniaturizar el dispositivo. Para ello se utilizó el programa Fritzing [43], dando como resultado el siguiente circuito impreso.

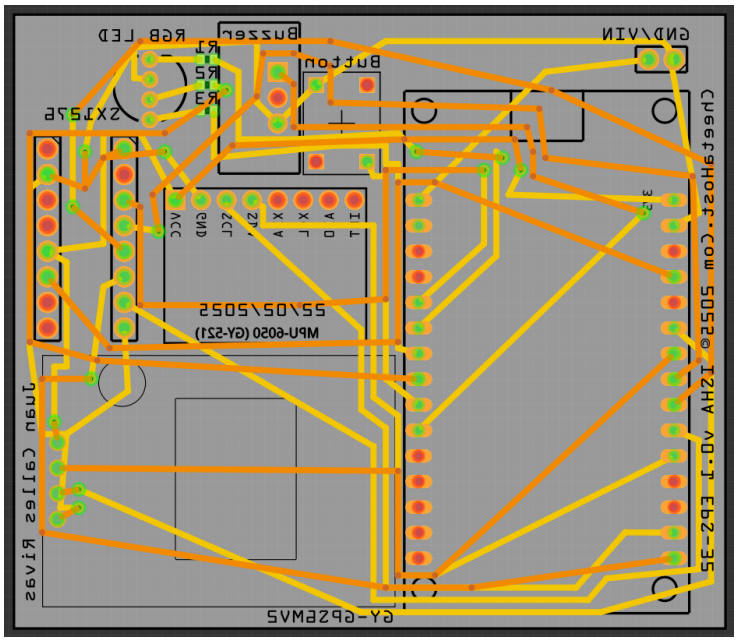


Ilustración 22 - Circuito impreso desarrollado para el proyecto.

Este circuito impreso, se mandó a fabricar a jlcpcb [44]. Dando como resultado el siguiente circuito impreso:

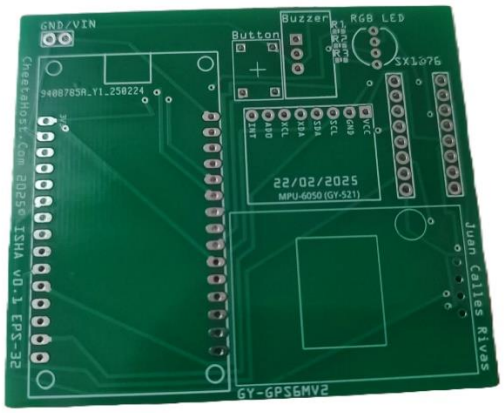


Ilustración 23 - Circuito impreso ya fabricado sin montar.

Finalmente, para el montaje se decidió, utilizar pines de prototipado hembra, esto con el objetivo de poder intercambiar componentes de forma modular, aunque aumenta el tamaño del dispositivo (si se tratara de un producto final, se realizaría la soldadura de los componentes directamente a la PCB)

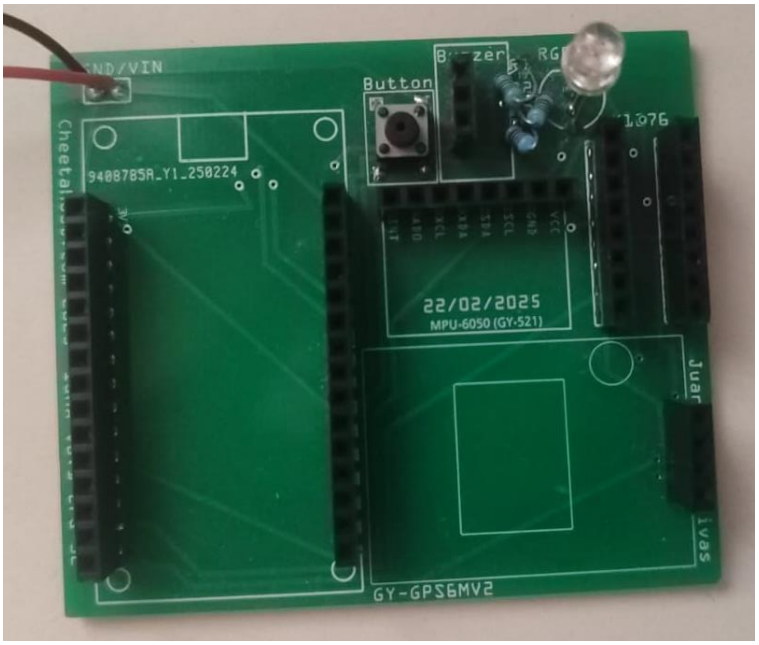


Ilustración 24 - Circuito impreso ensamblado con pines de prototipado.

5.2. Creación de carcasas para dispositivos

Para la realización de la carcasa y estructura de la base de carga y dispositivo, se hicieron uso del FabLAB USAL [45]. Se realizaron 3 prototipos utilizando técnicas diferentes, estos prototipos no se realizaron simultáneamente si no que fue más secuencialmente (uno después de otro) distanciados en el tiempo que se muestran a continuación:

- **Corte por Fresado**: se pensó originalmente realizar la carcasa con metacrilato acrílico, pudiendo ser visibles los componentes desde el exterior, haciendo más visual el dispositivo. La carcasa contaba de dos espacios contenedores, uno para los componentes *hardware* y otro para el sistema de alimentación (esta idea se mantuvo en el planteamiento de las otras dos carcasas). Finalmente fue totalmente descartada por si fragilidad, alto coste de producción. El prototipo que se genero fue el siguiente:

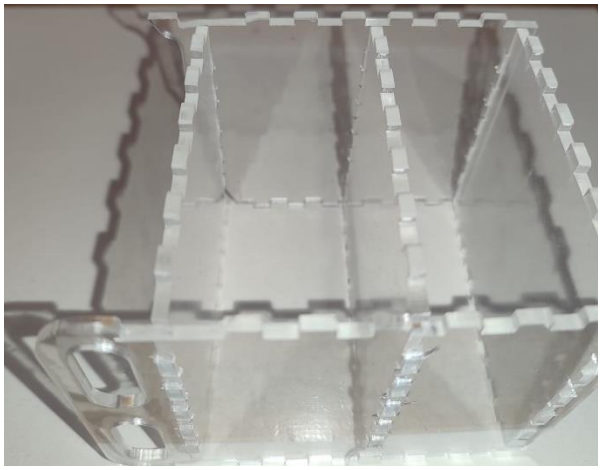


Ilustración 25 - Prototipo de carcasa Nº1 metacrilato.

- **Corte laser:** para la realización de este prototipo, se pensó en aumentar la durabilidad y disminuir los costes de producción. Se mantuvo gran parte del diseño anterior, uniendo las piezas mecánicamente. Esta vez se utilizó aglomerado y la cortadora láser. Además, se incorporaron cierres de maleta para poder acceder de una forma más sencilla a los componentes internos, sin necesidad de tener que desmontar la carcasa por completo. Este prototipo se muestra a continuación:

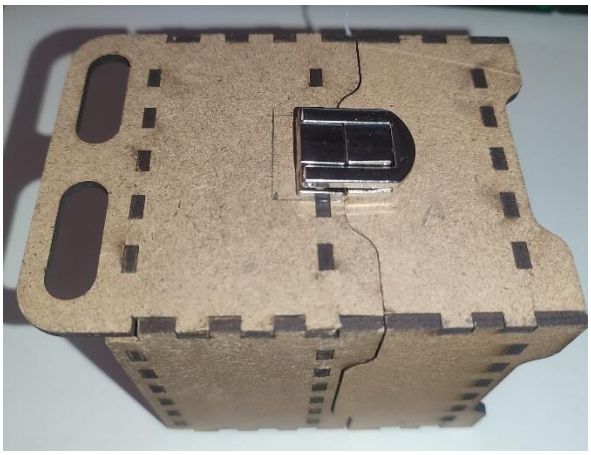


Ilustración 26 - Prototipo de carcasa Nº2 aglomerado.

- **Impresión 3D**: para la realización de este prototipo, se descartó todo lo anterior a excepción de las dos cavidades separadas (*hardware*-sistema de alimentación), para ello se tomó las medidas de los componentes del sistema realizando un mecanismo por presión permitiendo tener los componentes ajustados internamente sin necesidad de utilizar tornillos. El diseño se pensó para poder acceder a los componentes de forma rápida sin necesidad de desmontar toda la carcasa. Para la realización del diseño tanto del aparato como el de la base de carga se utilizó Tinkercar [46]. A continuación, se mostrarán algunas imágenes del modelo 3D, impresión de este y ensamblado final.

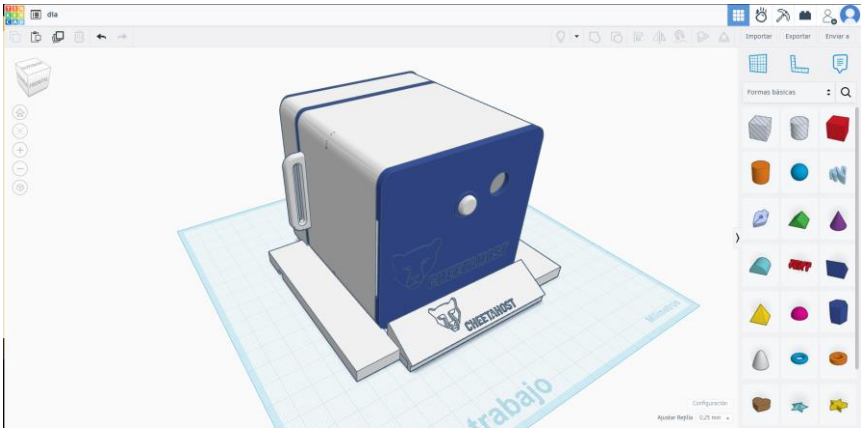


Ilustración 27 - Diseño 3D de la versión final del dispositivo y base de carga.

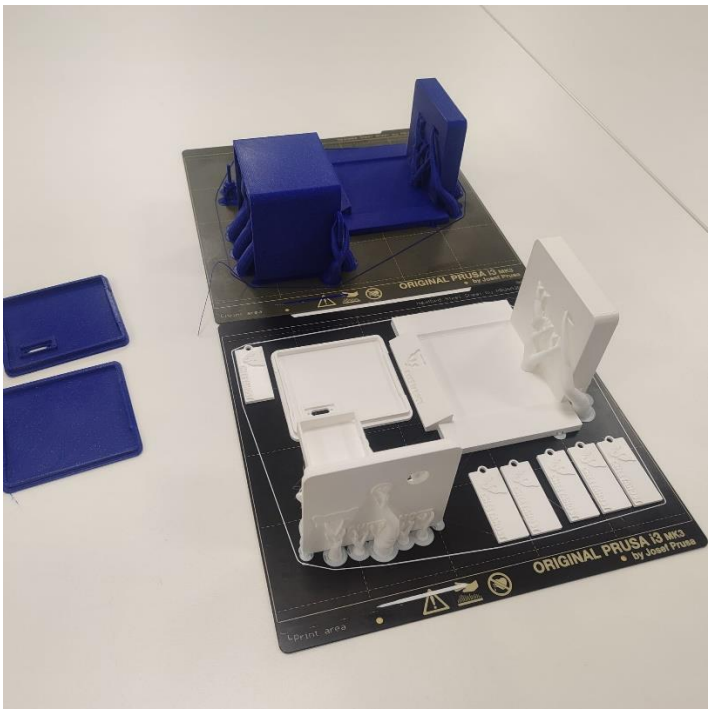


Ilustración 28 - Prototipo de carcasa Nº3 recién salido de la cama de impresión.

Ilustración 29 - Carcasa Nº3 ensamblada.

5.3. Creación del modelo detector de caídas.

Para la realización del modelo de detección de caídas, se realizó un desarrollo iterativo que abarcó desde la adquisición y tratamiento de datos hasta el entrenamiento y la selección final de un modelo considerado como el mejor para el proyecto.

5.3.1. Adquisición y preparación de datos: un enfoque híbrido

Como primer paso, fue la búsqueda y construcción de un conjunto de datos etiquetados compatibles con las características del proyecto. Tras realizar una búsqueda exhaustiva,

se encontró el *dataset* SisFall [15], el cual contiene un gran número de actividades de la vida diaria (ADL, actividades de la vida cotidiana) y caídas, específicamente, lo que llamó la atención fue que contenía un conjunto de datos de personas mayores de 65 años.

Este *dataset* es bastante completo, pero presentaba dos desafíos principales para el proyecto.

Como primer desafío que se presentó, los datos fueron grabados utilizando un sistema de 9 ejes (dos acelerómetros y un giroscopio). Dado que la MPU-6050 utilizada en el proyecto consta de 6 ejes (un acelerómetro de 3 ejes y un giroscopio de 3 ejes), fue necesario realizar una adaptación en el *dataset*. Esta adaptación se realizó mediante *scripts* en Python que seleccionaban únicamente las columnas correspondientes a los 6 ejes compatibles, descartando los datos del segundo acelerómetro para poder asegurar la mayor compatibilidad con el *hardware* del proyecto.

Como segundo desafío, para prevenir que el modelo de detección de caídas no fuera del todo compatible con el *hardware* seleccionado para el proyecto (MPU-6050), se decidió realizar grabaciones de secuencias propias. Se creó un *script* con el objetivo de capturar en tiempo real los datos del sensor a través del microcontrolador utilizando el puerto serie del ordenador, permitiendo registrar y etiquetar manualmente secuencias de "caídas" y "no caídas". Estas secuencias propias conformaron un *dataset* a medida, que resultó ser fundamental para el entrenamiento de un modelo compatible y preciso, como se detallará más adelante.

5.3.2. Extracción de características

Los datos crudos de los sensores, que son secuencias temporales, los cuales no son directamente utilizables para realizar el entrenamiento del modelo de clasificación. A causa de esto, se aplicó una técnica de ventanas deslizantes para poder transformar los datos. El proceso consistió en segmentar la señal continua en ventanas de 1 segundo con un 50% de solapamiento. Para cada una de estas ventanas temporales, se calcularon un conjunto de 15 características estadísticas que resumen la información contenida en el intervalo dado:

Estadísticas básicas del acelerómetro: se calcularon la media, desviación estándar, valor máximo y el mínimo para cada uno de los tres ejes de aceleración (ax, ay, az).

Magnitud de los vectores: se calculó la media de la magnitud del vector de aceleración ($\sqrt{a_x^2 + a_y^2 + a_z^2}$) y del vector de velocidad angular ($\sqrt{g_x^2 + g_y^2 + g_z^2}$). Estas métricas son independientes de la orientación que tenga el dispositivo, capturan la intensidad general del movimiento.

Cambio de inclinación (Δ_{tilt}): se calculó la variación en el ángulo de inclinación del dispositivo, esto es un indicador clave para detectar cambios bruscos en las posturas del usuario y poder detectar las que implican una caída:

$$\Delta_{tilt} = arcos\left(\frac{a_{z,final}}{||a_{zfinal}||}\right) - arcos\left(\frac{a_{z,inicial}}{||a_{zinicial}||}\right)$$

Este proceso de extracción de características se aplicó de manera sistemática a las secuencias del *dataset* propio, generando así el conjunto de datos final para el entrenamiento de los modelos.

5.3.3. Evolución del entrenamiento

Una vez teniendo los datos procesados y listos, se exploraron varias opciones. Al igual que en la realización de las carcasas, se desarrollaron diferentes modelos con tecnologías distintas hasta poder llegar al finalmente utilizado en el proyecto. La primera de estas tecnologías/técnicas fue *TinyML*, con el objetivo de poder portar el modelo entrenado en Python al microcontrolador (código C). Esta opción fue descartada debido a su poca compatibilidad con el sistema que se estaba usando para desarrollar el *firmware* de los dispositivos, por lo que se buscaron otras alternativas, donde destacó emlearn [47].

Se utilizó emlearn para poder entrenar un árbol de decisión, permitiendo exportarlo en un fichero de cabecera .h integrable directamente con el *firmware* del dispositivo. Sin embargo, esta opción fue rápidamente descartada, ya que generaba un gran número de falsos positivos.

El análisis del problema que la causa del problema se trataba de un fuerte desequilibrio de clases en el conjunto de datos, dado que las muestras de actividades "cotidianas" superaban y por mucho a las de "caídas". Para solucionar esto y evitar sesgos en el modelo, se realizó un proceso de balanceo de datos, permitiendo tener un equilibrio entre el número de elementos de clases.

Con los datos ya balanceados, se diseñó la arquitectura del modelo final. Dado que un algoritmo convencional como el *Random Forest* podía ser demasiado pesado para el microcontrolador, mientras que uso de un único árbol de decisión ya había demostrado ser demasiado simple. Por ello, se optó por una solución especifica:

Realizar un sistema de ensamblado ligero formado por dos árboles de decisión heterogéneos uno balanceado y el otro no, funcionan en conjunto a través un mecanismo de votación por acumulación de evidencia.

Este sistema funciona de la siguiente manera:

1. **Dos árboles de decisión especializados:** se entrenaron dos modelos de árbol de decisión de forma independiente. La clave para su eficacia es que cada árbol se entrenó utilizando un conjunto de características distinto, lo que los hace expertos en detectar diferentes matices del movimiento. Esta diversidad permite lograr una predicción conjunta más robusta.

2. **Sistema de votación con acumulador temporal:** en lugar de una votación simple, el *firmware* implementa una lógica de "integración de evidencia". A través de las predicciones de ventanas temporales sucesivas, un contador va acumulando "puntos de evidencia". Si ambos modelos coinciden en que se trata de una “caída”, el contador aumenta más rápidamente. Una alerta de caída solo se confirma cuando este contador supera un umbral predefinido, lo que garantiza que la detección se base en un patrón sostenido en el tiempo y no en un movimiento brusco y aislado.

Esta solución final logró el equilibrio deseado: la robustez de un sistema de ensamblado para reducir los falsos positivos y la eficiencia computacional necesaria para poder operar de forma fiable en el *hardware* limitado del dispositivo.

5.4. Ciclo de vida

Para la realización de la estructura y desarrollo del proyecto, se ha optado por el uso del proceso unificado. Esta elección, se debe a las características proyecto y a la familiaridad del desarrollador con esta metodología.

El Proceso Unificado contiene varias características fundamentales que guían el desarrollo de sistemas:

- **Orientado a casos de uso:** la implementación de la metodología se dirige y verifica a través de los casos de uso, lo que garantiza la implementación y el correcto funcionamiento de todas las características esperadas por el sistema durante su fase de diseño.
- **Iteractividad y aumento gradual:** se adhiere al principio de divide y vencerás, dividiendo el proyecto en subsecciones pequeños y manejables. Este enfoque facilita la gestión y el control del progreso.
- **Prioridad arquitectónica:** la arquitectura del sistema es diseñada al principio y va evoluciona progresivamente a lo largo del desarrollo. Se realiza utilizando múltiples perspectivas que se van especializando continuamente.

El proceso se desarrolla en fases denominadas ciclos que se van repitiendo a lo lardo del desarrollo del proyecto. Cada uno de estos ciclos iterativos constan de cuatro fases:

- **Fase inicial:** la fase inicial es donde se definen los objetivos y se delimita el alcance del proyecto. En el caso del TFG, se definieron las funcionalidades que debería implementar el sistema y los objetivos a alcanzar.
- **Fase de elaboración:** en esta fase, se lleva a cabo la planificación de una forma más detallada. Definiendo la funcionalidad de forma precisa y la arquitectura técnica a usar. En esta sección, se realizaron las especificaciones técnicas de la funcionalidad que deberían cumplir el proyecto, exponiendo tecnologías, etc.
- **Fase de construcción:** es la etapa central de desarrollo, donde se realiza la implementación del código y de los componentes. Durante esta fase se procedió a la implementación del *firmware*, *bakend* y *frontend.*
- **Fase de transición:** durante esta fase, es donde construcciones realizadas en la fase anterior, comienzan a integrarse y probarse. Durante esta fase, se identifica e intenta subsanar cualquier defecto o realizar mejoras que surjan durante la puesta en funcionamiento del sistema. El sistema es finalmente desplegado en los servicios de cheetahost [48] pudiendo realizar pruebas en producción.

La elección del Proceso Unificado frente a otras metodologías se debe a dos razones principales:

- **Ventaja en la experiencia del desarrollador:** el desarrollador está familiarizado con el proceso Unificado de forma práctica, gracias a la realización de numerosas

prácticas durante el desarrollo del grado, permitiendo ser elegida antes que otras metodologías ágiles como podría ser *Scrum*.
- **Consistencia con el inicio del proyecto:** desde las fases tempranas e incluso conceptuales del proyecto, se han ido estableciendo los objetivos y requisitos funcionales del sistema, así como la propia lógica de negocio, en base a esta metodología.

Para poder entrar en mayor detalle sobre esta sección, se ruega al lector consultar el Anexo I.

5.5. Especificación de requisitos y análisis del sistema.

En esta sección, se va a explicar los elementos que han sido más relevantes durante el desarrollo de las tareas de análisis y especificación de requisitos. Dado que estas operaciones se realizaron con mayor detalle a lo largo del Anexo II, a lo largo de esta sección solo se especificarán los aspectos más relevantes dados durante el desarrollo del citado anexo.

5.5.1. Subsistemas del proyecto

A continuación, se van a explicar el papel que ocupan los subsistemas del proyecto que aparecen en la ilustración 30:

- **Gestión de emergencias:** este subsistema se encargará de procesar las alertas y notificar a los familiares/servicios de emergencia correspondientes.
- **Consulta de datos:** acceso a la persistencia del sistema para los diversos fines correspondientes.
- **Autenticación y cuentas:** subsistema encargado de administrar el acceso al sistema por parte de los usuarios.
- **Gestión de antenas:** subsistema encargado de la administración de las antenas desplegadas en el sistema, permitiendo realizar operaciones en estas ya sean consulta de información, datos estadísticos, etc.
- **Gestión de dispositivos:** subsistema encargado de manejar los dispositivos que están dados de alta en el sistema, permitiendo realizar asignaciones de estos a usuario, consulta de estadísticas, etc.
- **Gestión de usuarios y roles.**

Estos subsistemas conforman a grandes rasgos tanto la interfaz gráfica del sistema como la lógica de negocio tal y como se muestra a continuación.

Interfaz gráfica (*frontend*): el panel *web*, permite realizar las operaciones principales relacionadas con la gestión de la persistencia. Es el punto de entrada del usuario a los demás subsistemas. Los subsistemas implicados son:

- Gestión de Emergencias.
- Consulta de Datos.
- Autenticación y Cuentas.
- Gestión de Antenas.

- Gestión de Dispositivos.
- Gestión de Usuarios y Roles.

Lógica de negocio (*Backend*): realiza el procesamiento de la información que fluye por el sistema y las operaciones ordinarias del mismo.

- Gestión de Alertas.
- Gestión de Antenas.
- Gestión de Dispositivos.
- Consulta de Datos.

Desde el planteamiento del proyecto, se ha intentado definir de forma clara y concisa la funcionalidad den los correspondientes subsistemas. Esto con el objetivo de facilitar el desarrollo del proyecto y su futura mantenibilidad.

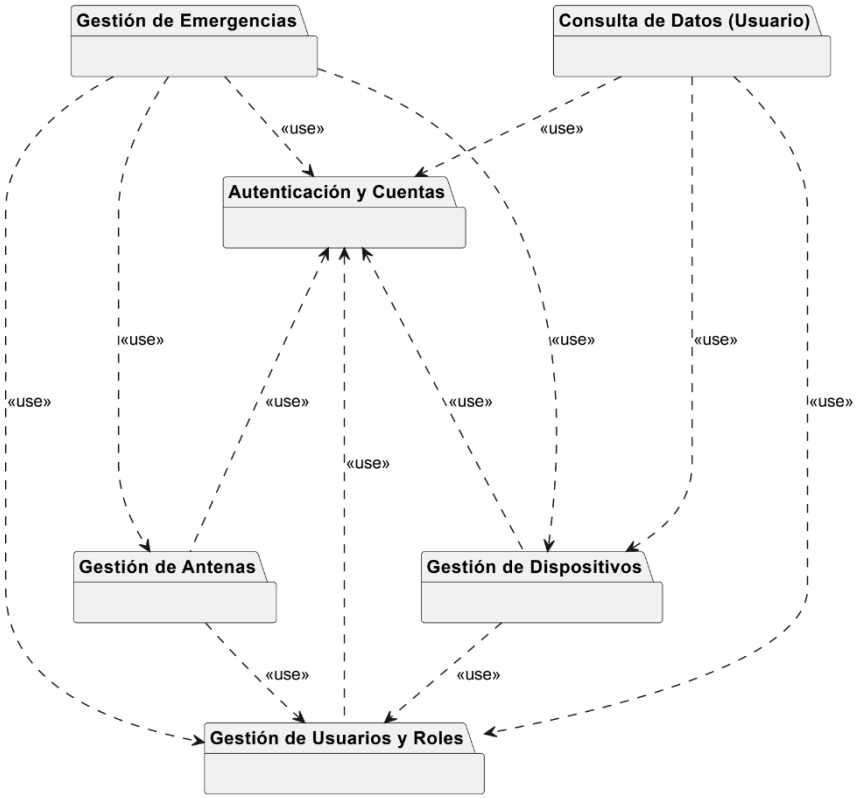


Ilustración 30 - Diagrama de subsistemas.

5.5.2. Diagramas de casos de uso

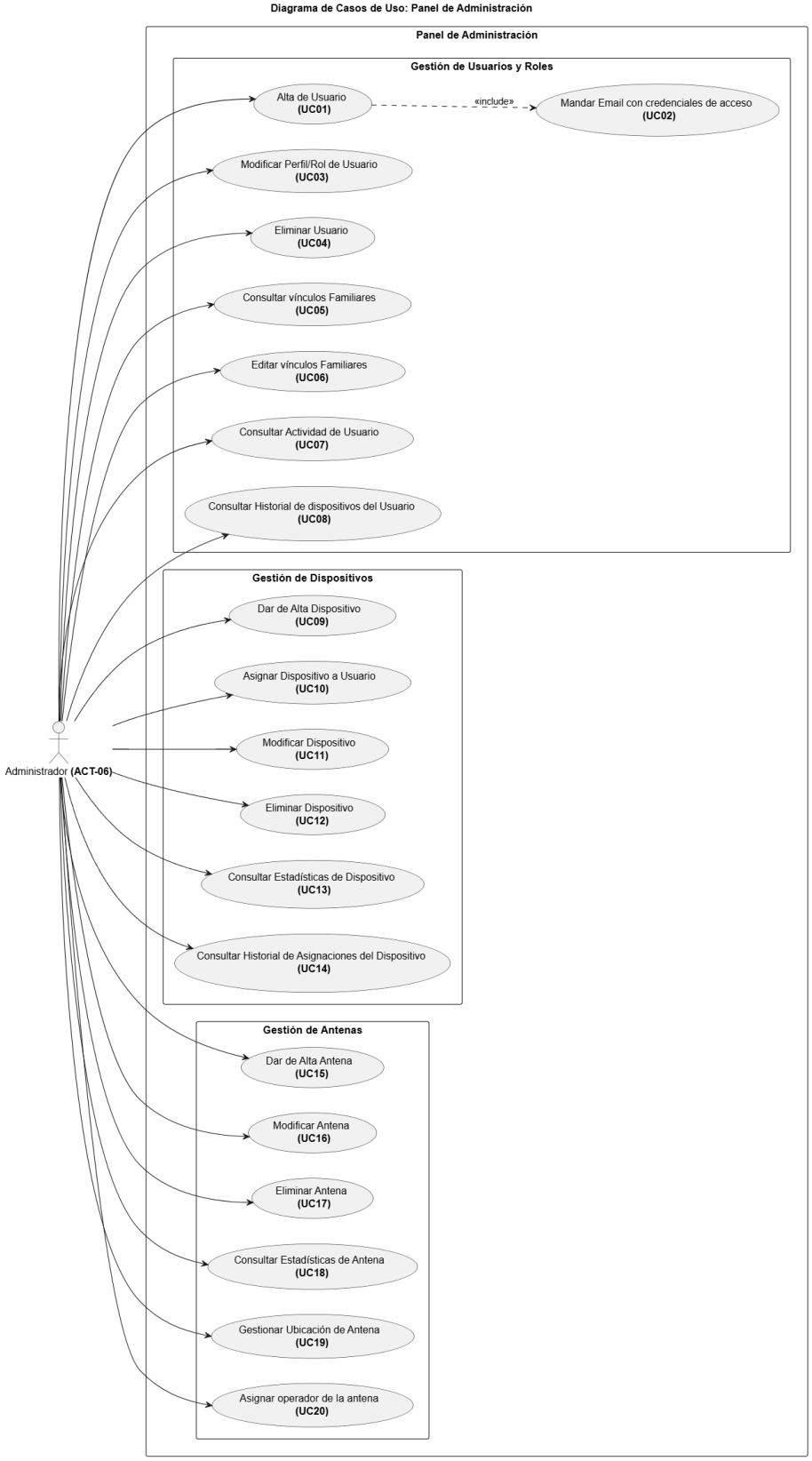


Ilustración 31 - Diagrama de casos de uso: Panel de Administración.

Los casos de uso que se pueden observar en la ilustración 31, son los que puede llevar a cabo el usuario administrador del sistema durante su tarea de administración. Para mayor detalle sobre estos, consultar el Anexo II

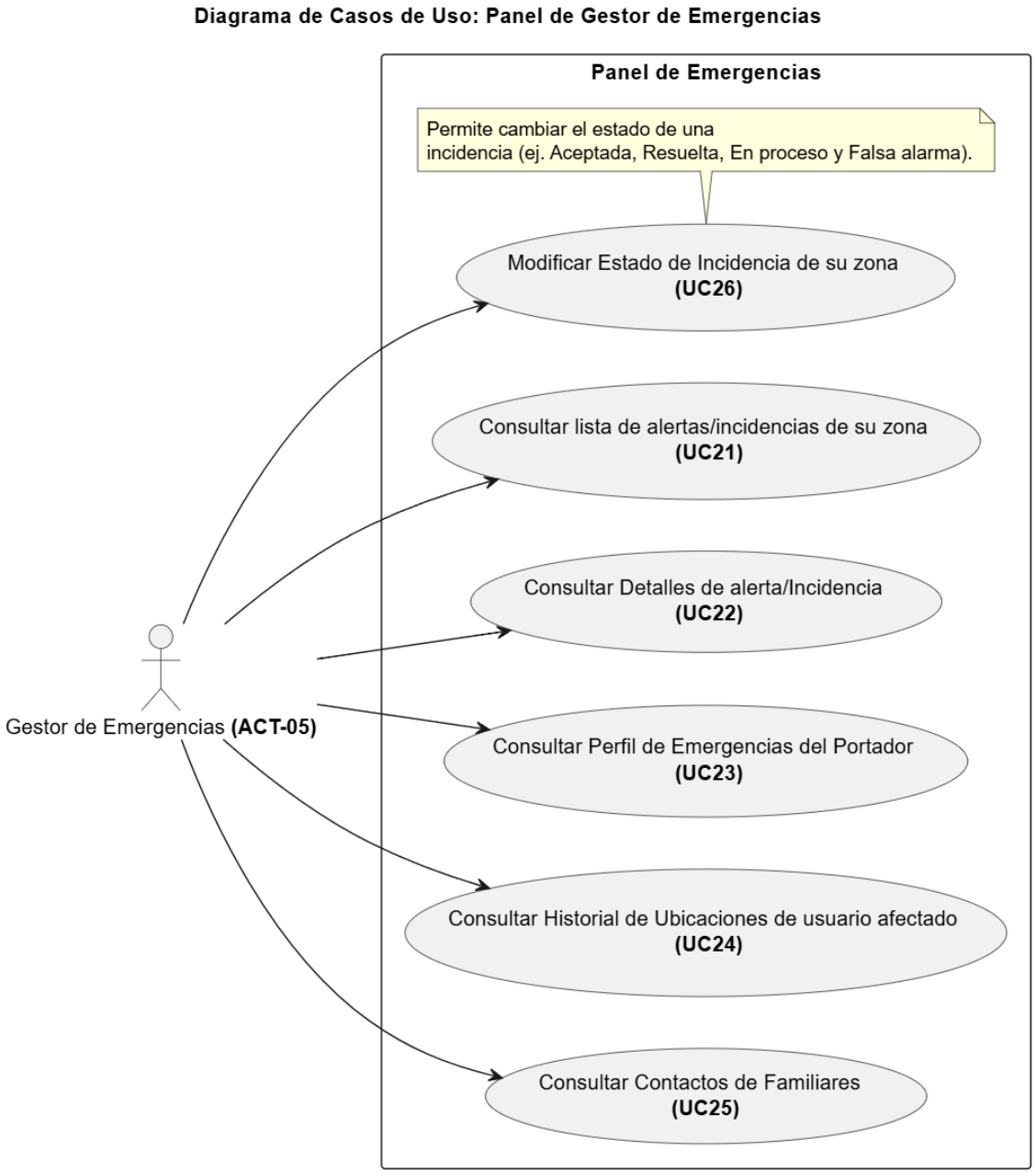


Ilustración 32 - Diagrama de casos de uso: Panel de Gestor de Emergencias.

Los casos de uso presentes en la ilustración 32, son los que puede realizar el usuario gestor de emergencias. Definiendo el proceso correspondiente a la gestión de emergencias en el sistema. Para mayor detalle, consulte el Anexo II

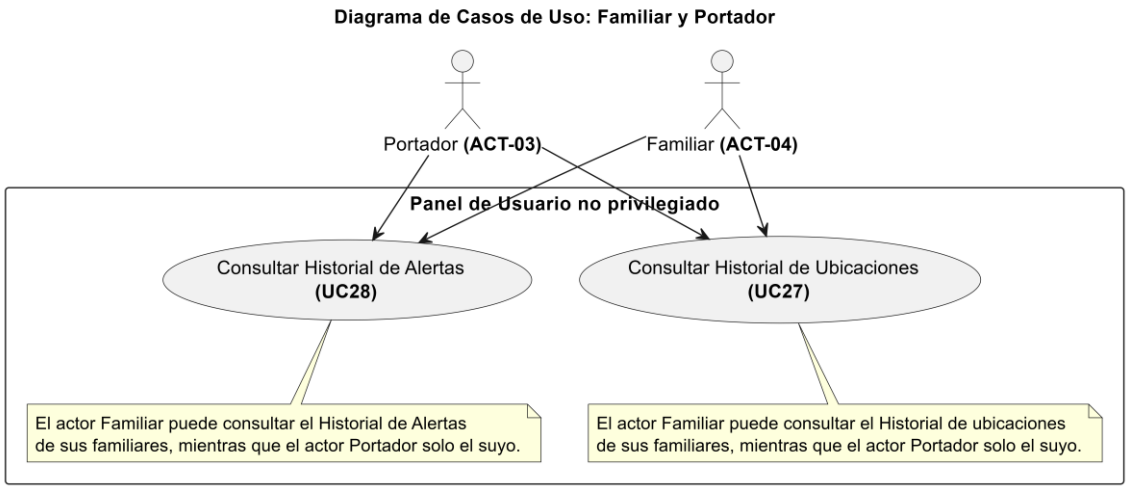


Ilustración 33 - Diagrama de casos de uso: Familiar y Portador.

En la ilustración 33, representan las acciones que pueden realizar los usuarios familiares y portadores en el sistema. Para mayor detalle, consulte el Anexo II

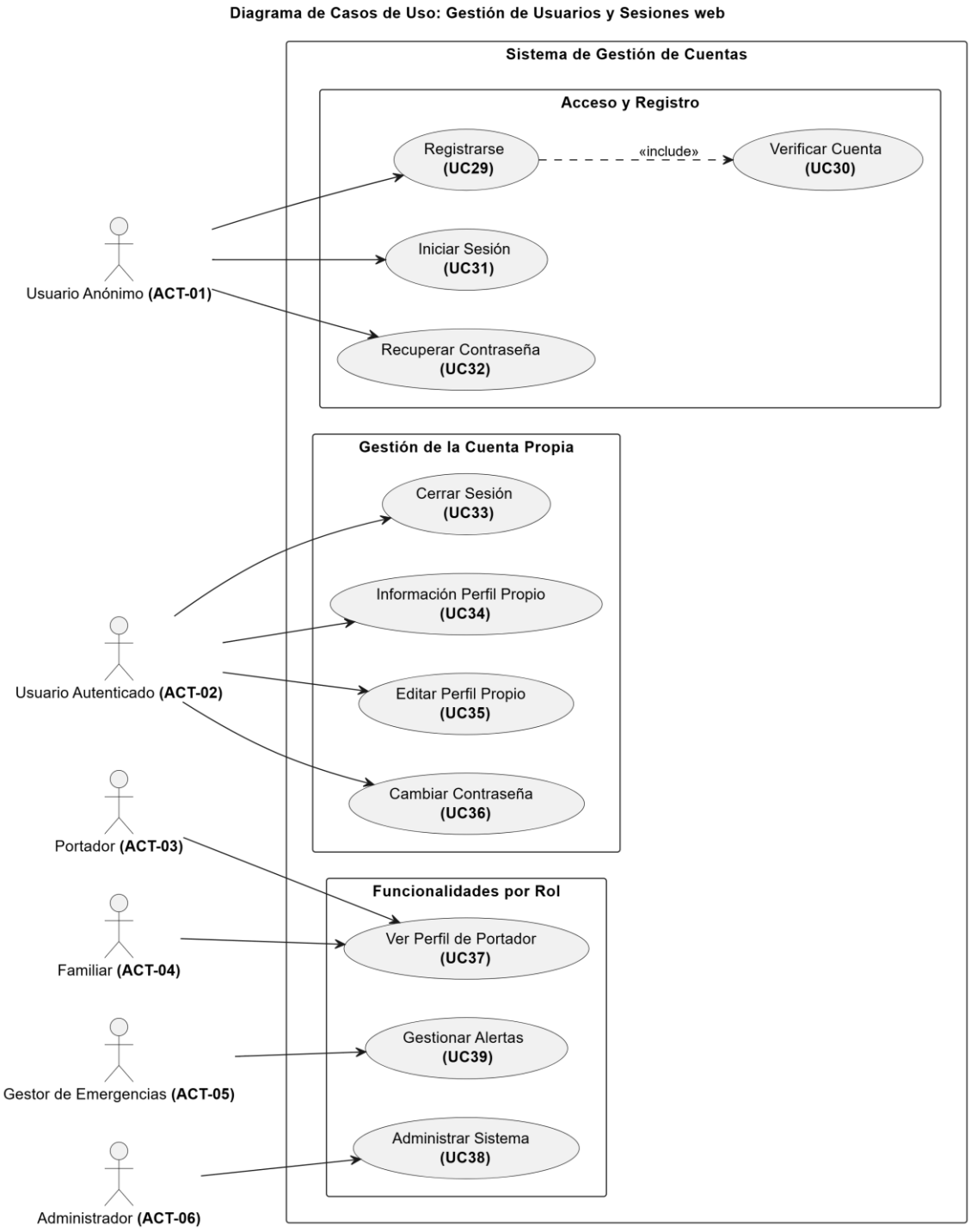


Ilustración 34 - Diagrama de casos de uso: Gestión de Usuarios y Sesiones web.

En la ilustración 34 se representa a alto nivel las acciones que puede realizar los usuarios humanos del sistema. Para mayor detalle consulte el Anexo II.

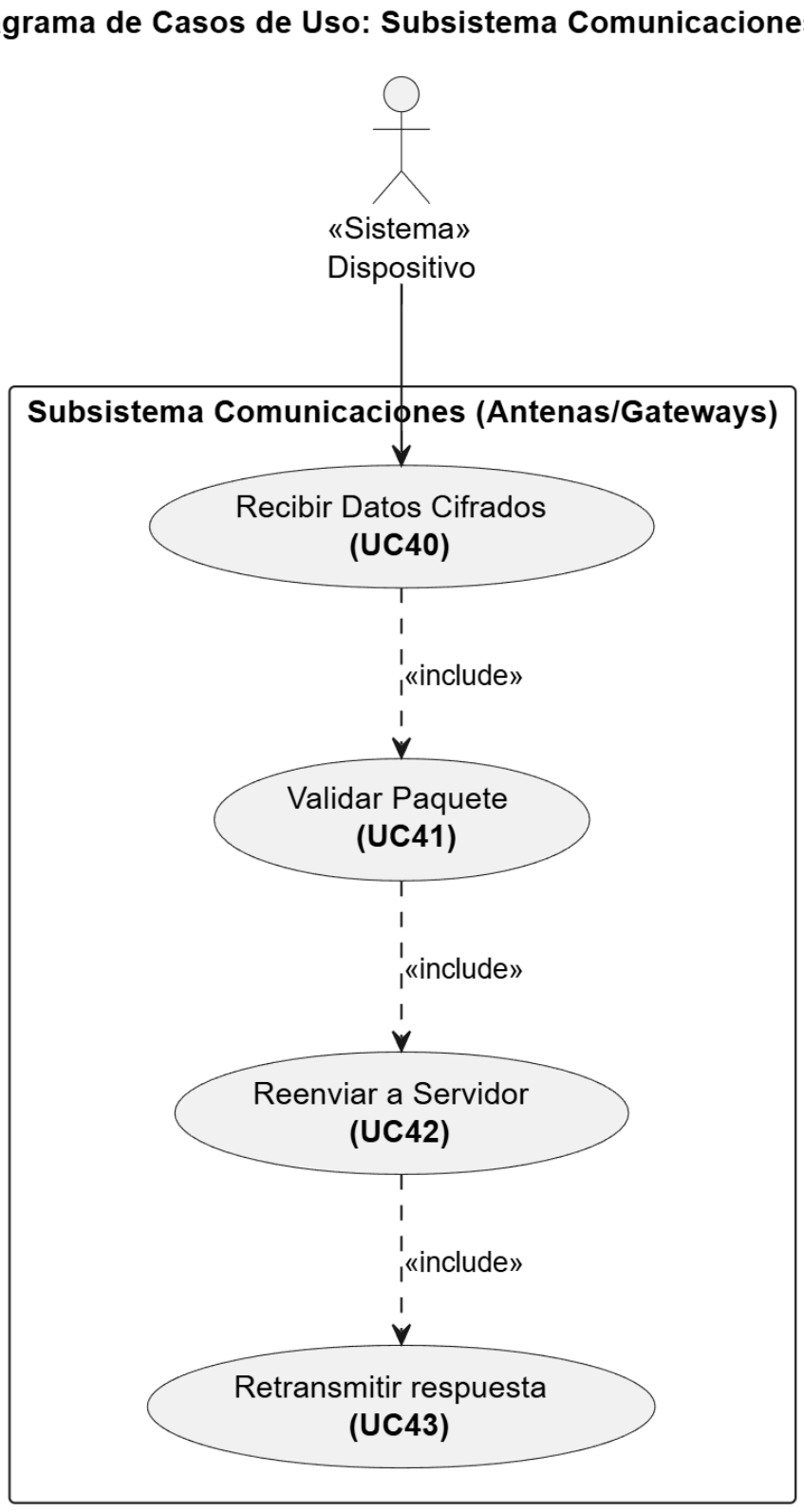


Ilustración 35 - Diagrama de casos de uso: Subsistema Comunicaciones.

En la ilustración 35 se representa el proceso de avance de mensajes que realizan los operadores no humanos dispositivos y antenas en el sistema. Para mayor detalle, consulte el Anexo II.

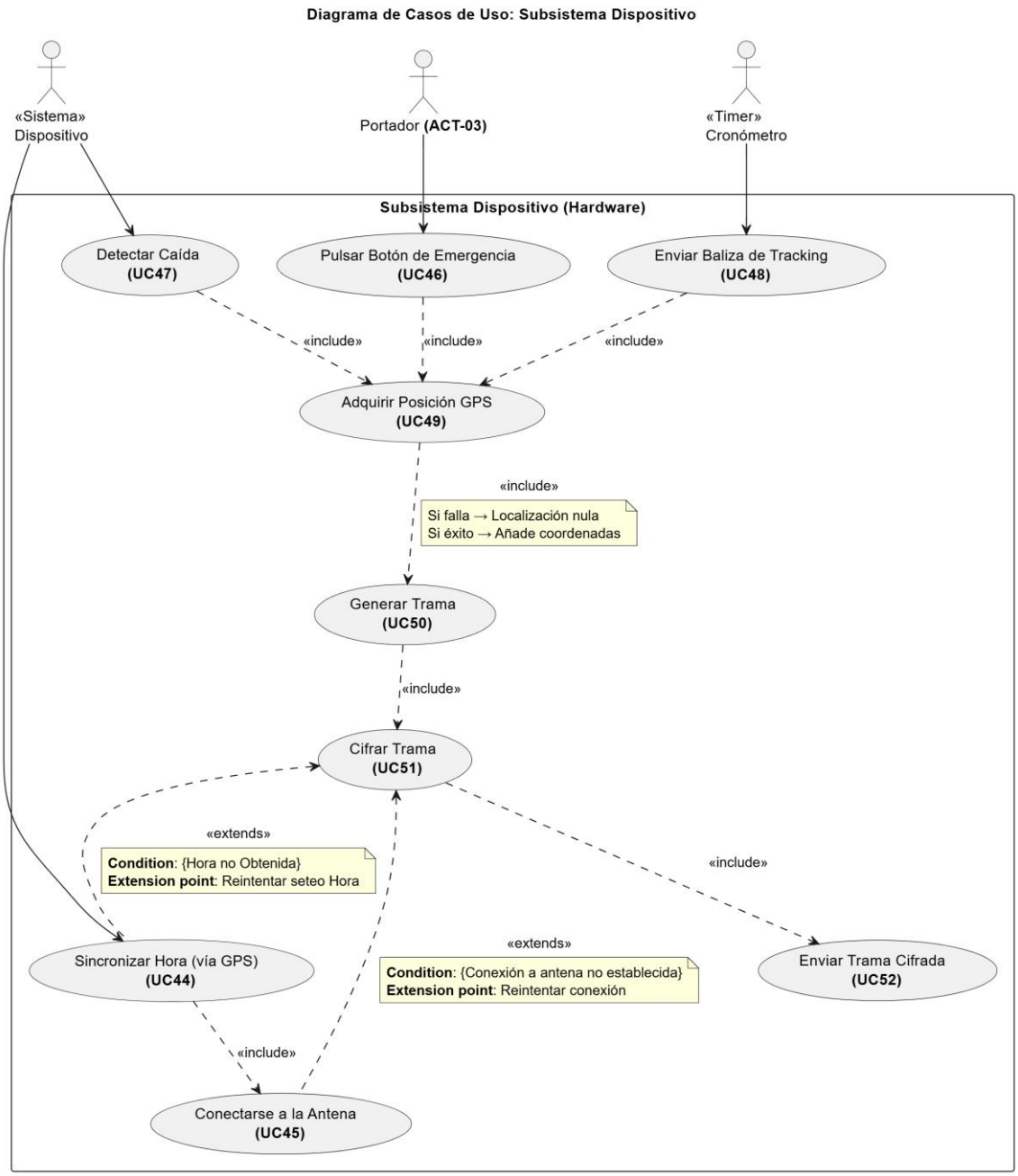


Ilustración 36 - Diagrama de casos de uso: Subsistema Dispositivo.

En la ilustración 36, se representa el funcionamiento interno del dispositivo de usuario. Para mayor detalle, se pide consultar el Anexo II.

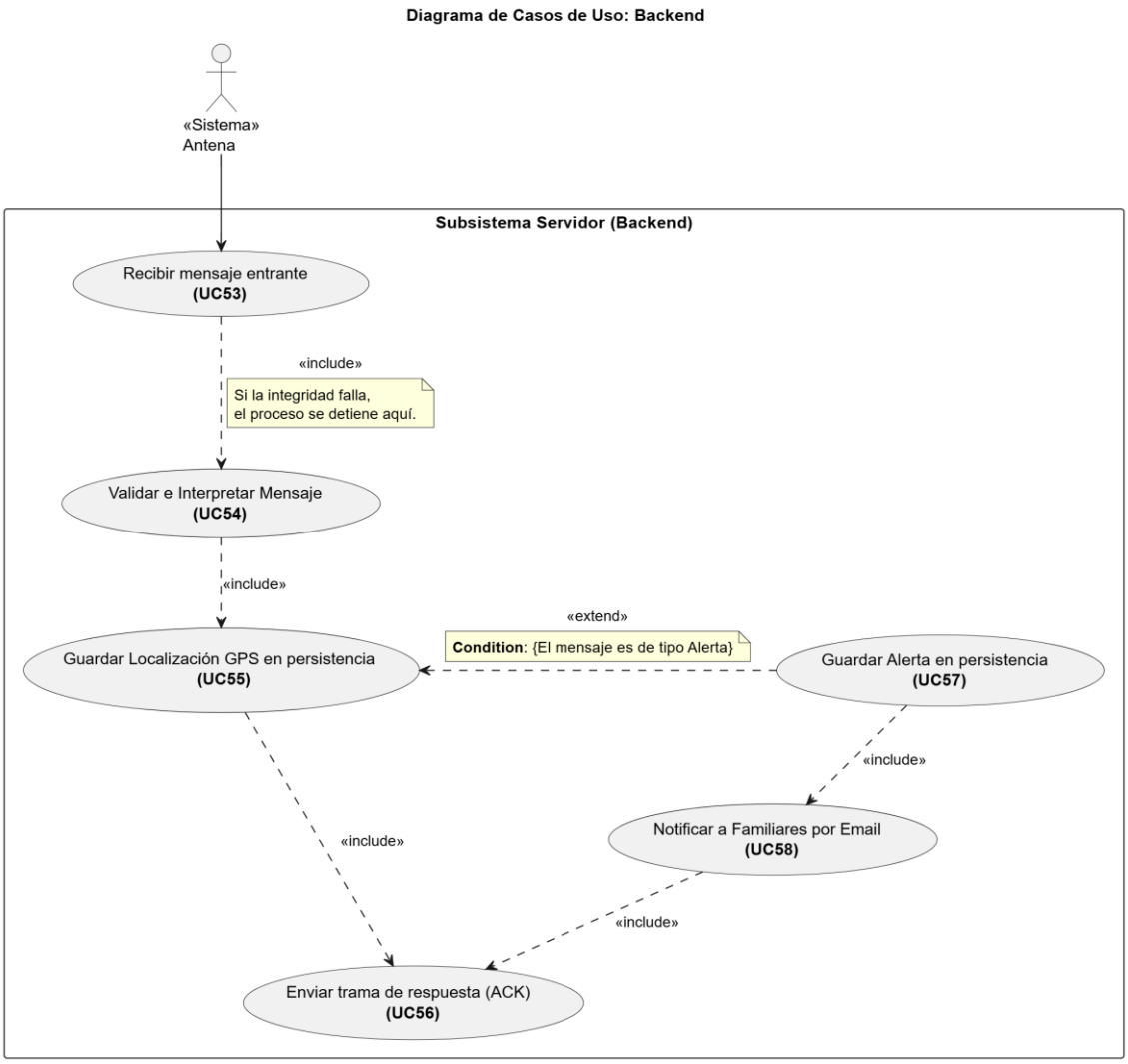


Ilustración 37 - Diagrama de casos de uso: Backend.

En el diagrama 37 se representa el funcionamiento interno del servidor. Para mayor detalle, consúltese el Anexo II.

En los diagramas de casos de uso mostrados, se modela la estructura global del proyecto, mostrando las dos formas principales de interactuar los usuarios con el mismo. Dadas las especificaciones del proyecto los usuarios podrán interactuar a través del *frontend* o a través del propio dispositivo de usuario.

5.5.3. Modelo de dominio

Lo más importante realizado durante la fase de análisis, fue el desarrollo del modelo de dominio donde además de modelar los requisitos de información, se tiene en cuenta las relaciones entre ellos.

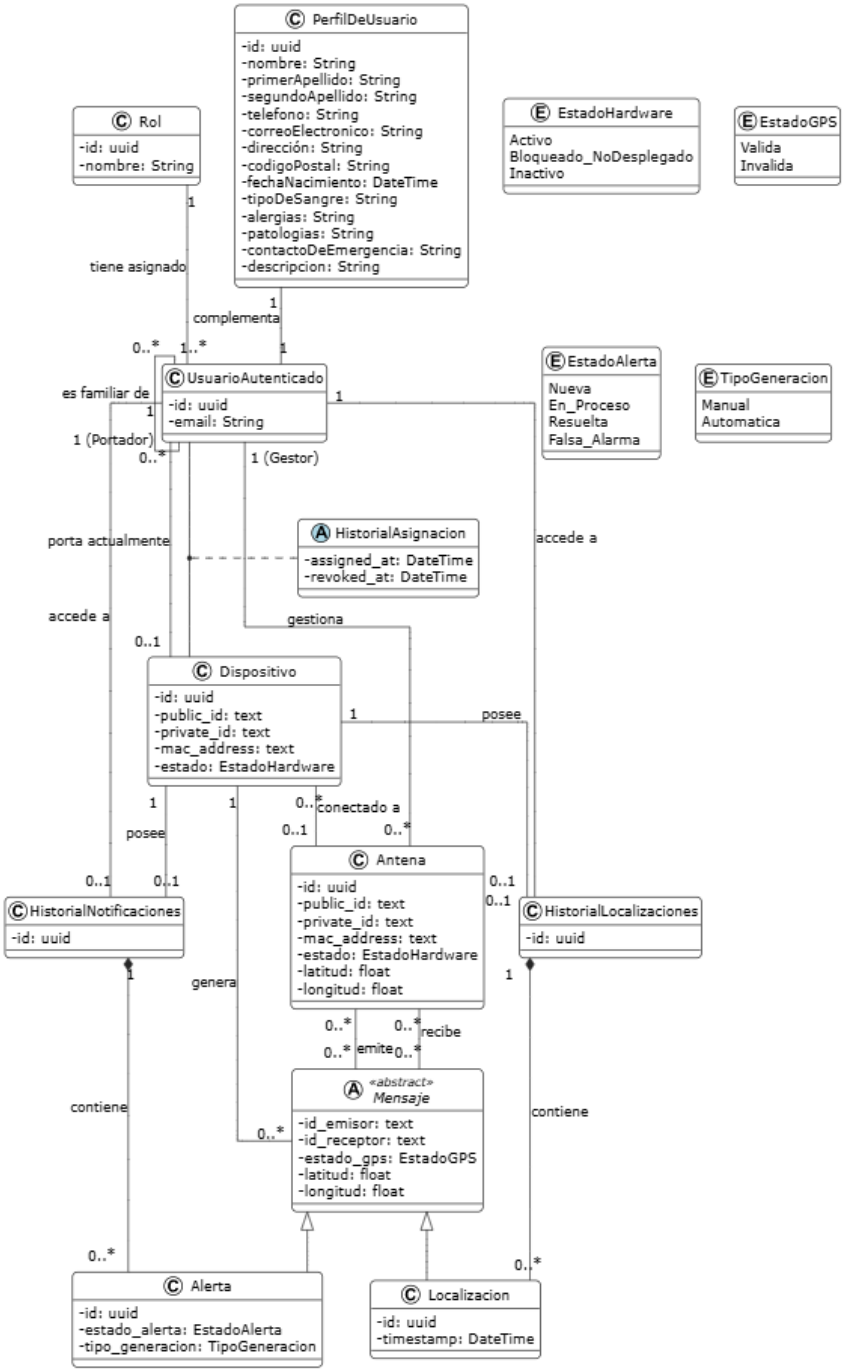


Ilustración 38 - Modelo de Dominio.

En este diagrama, se ha realizado el modelado de las clases que conforman el dominio del sistema, mostrándose como se relacionan los diferentes componentes y los datos que tienen.

A continuación, se realizará una breve descripción de los elementos que componen el modelo de dominio que se puede observar en la ilustración 38, para mayor detalle se pide consultar el Anexo III.

- **UsuarioAutenticado:** representa a cualquier persona que utiliza el sistema, gestionando su identidad, rol y las relaciones principales con otros componentes como dispositivos o familiares.
- **Dispositivo:** modela el dispositivo de usuario que registra y envía datos de telemetría, como alertas y localizaciones, siendo la fuente principal de eventos del sistema.
- **Antena:** actúa como un puente de comunicación (*gateway*) que recibe las señales de los dispositivos (*LoRa*) y las retransmite a la red de internet para que lleguen al servidor.
- **Mensaje (abstracta):** es la plantilla base para todas las comunicaciones enviadas por los dispositivos, definiendo la estructura común de datos como la geolocalización y los identificadores.
- **Alerta:** representa un tipo de mensaje específico para eventos de emergencia, ya sean activados manualmente por el usuario o automáticamente por el dispositivo.
- **Localizacion:** es un tipo de mensaje periódico (*heartbeat/*baliza) que el dispositivo envía para reportar su ubicación de forma rutinaria, permitiendo el seguimiento continuo.
- **HistorialAsignacion:** registra qué dispositivo ha estado asignado a qué usuario a lo largo del tiempo, detallando el inicio y el fin de cada periodo de asignación.
- **HistorialNotificaciones y HistorialLocalizaciones:** son colecciones que agrupan todos los mensajes de alerta y de localización generados por un único dispositivo a lo largo de su vida útil.
- **PerfilDeUsuario:** almacena la información personal y médica detallada del usuario, como datos de contacto, alergias o patologías, complementando a la entidad UsuarioAutenticado.
- **Rol:** define el nivel de permisos y las capacidades que un usuario tiene dentro del sistema, como por ejemplo "Portador", "Familiar" o "Gestor".

5.6. Especificación de diseño

Lo más importante de la especificación es el modelado del diagrama de entidad-relación, la forma en la que se han integrado los datos requeridos y normalizado la capa de persistencia. A continuación, se muestra el diagrama entidad relación del proyecto.

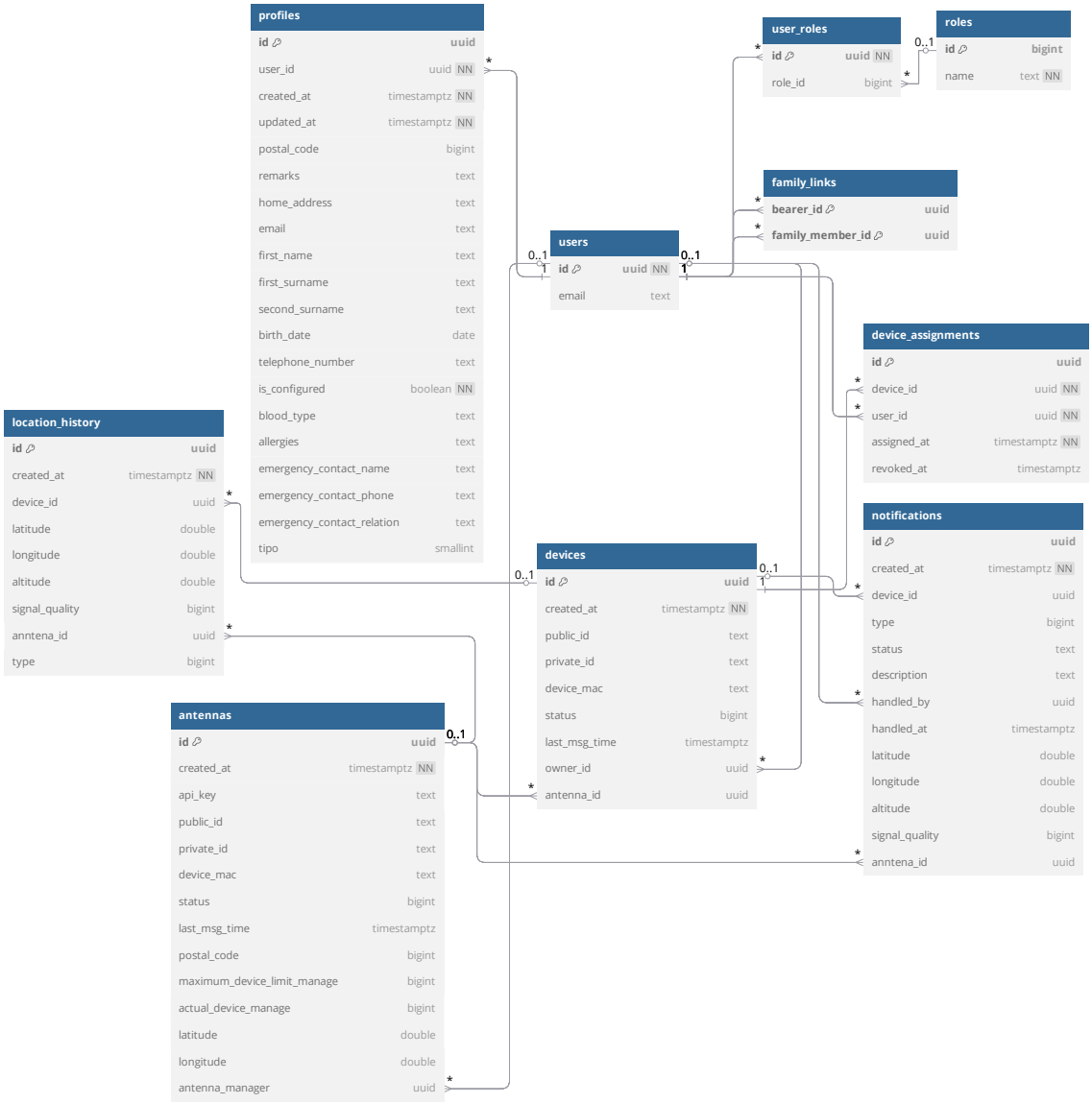


Ilustración 39 - Diagrama Entidad Relación del sistema.

Continuación se realizará una breve descripción de las tablas que conforman la persistencia mostrada en la ilustración 39. Para mayor detalle, consulte el Anexo III

- **profiles:** almacena toda la información personal, de contacto y médica de los usuarios, como su nombre, dirección o tipo de sangre, etc.
- **roles:** Contiene el catálogo de los diferentes tipos de usuario que existen en el sistema, como "Administrador", "Familiar" o "Portador".
- **user_roles:** es una tabla intermedia que conecta a cada usuario con su rol específico, definiendo así sus permisos y nivel de acceso en el sistema.

- **devices:** registra cada dispositivo físico del sistema, incluyendo sus identificadores únicos y el usuario que lo tiene asignado en ese momento, etc.
- **antennas:** guarda la información de cada antena de la red, como su identificador, su ubicación geográfica y el gestor responsable de ella, etc.
- **notifications:** almacena un registro detallado de cada alerta de emergencia generada, incluyendo quién la generó, dónde ocurrió y cómo se gestionó.
- **location_history:** mantiene un historial completo de todas las coordenadas geográficas enviadas por los dispositivos, permitiendo el seguimiento de su ubicación.
- **family_links:** establece y gestiona las relaciones entre los usuarios portadores y sus familiares o contactos designados dentro de la plataforma.
- **device_assignments:** lleva un registro histórico de todas las asignaciones de dispositivos a usuarios, detallando quién usó qué dispositivo y durante qué período.
- **users:** es la tabla central de autenticación que contiene la información de inicio de sesión (*email*, contraseña) para cada persona registrada en el sistema.

5.6.1. Clases Antenas y Dispositivos como parte de la abstracción de la arquitectura.

Las clases Antenas y Dispositivos se han modelado a partir de la abstracción del sistema, representando los componentes *hardware* diseñados. Ambos componentes *hardware* comparten el mismo *firmware* en común, su funcionalidad cambia dependiendo de su rol (Dispositivo o Antena). Para gestionar estos cambios de comportamiento manera flexible y extensible, se decidió hacer uso del patrón de diseño *Strategy*.

El Patrón Strategy permite definir una familia de algoritmos (en este caso serían los diferentes modos de operación del *firmware*), encapsular cada uno de ellos y hacerlos intercambiables. Esto significa que el algoritmo puede variar independientemente de los clientes que lo utilicen.

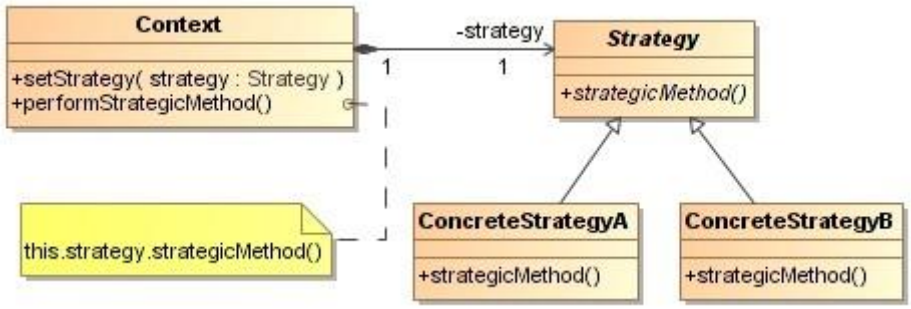


Ilustración 40 - Patrón de Diseño Strategy.

La elección del patrón Strategy para la realización del *firmware* se motiva por los siguientes motivos:

- **Reusabilidad del *firmware*:** dado que las Antenas y Dispositivos utilizan el mismo *firmware* base, el patrón Strategy permite mantener una única implementación del *firmware* y solo cambiar la "estrategia" o el "modo de operación" que ejecuta el *firmware* en función de si es una Antena o un Dispositivo. Esto agiliza considerablemente el desarrollo asentando unas bases comunes para todos los modos de operación sin tener que realizar *firmwares* de forma totalmente separada para cada uno de los componentes.
- **Flexibilidad y extensibilidad:** si en un futuro fuera necesario añadir nuevos modos de operación al *firmware*, simplemente se crearía una nueva estrategia sin modificar las clases existentes Antena o Dispositivo ni el *firmware* base existente.
- **Claridad del diseño:** separa los modos de operación del comportamiento del *firmware* subyacente, lo que mejora la legibilidad y el mantenimiento del código, permitiendo utilizar las mismas piezas para fines distintos.
- **Evita condicionales complejas:** en lugar de utilizar extensas sentencias *if-else* o *switch* para poder determinar el comportamiento en función del tipo de *hardware* en todo momento, el patrón Strategy permite delegar la responsabilidad de la estrategia a secciones específicas.

5.7. Despliegue del sistema

Para la realización del despliegue, se han utilizado las tecnologías comentadas en el apartado 4 de este mismo documento.

A continuación, se muestra el diagrama de despliegue del sistema realizado:

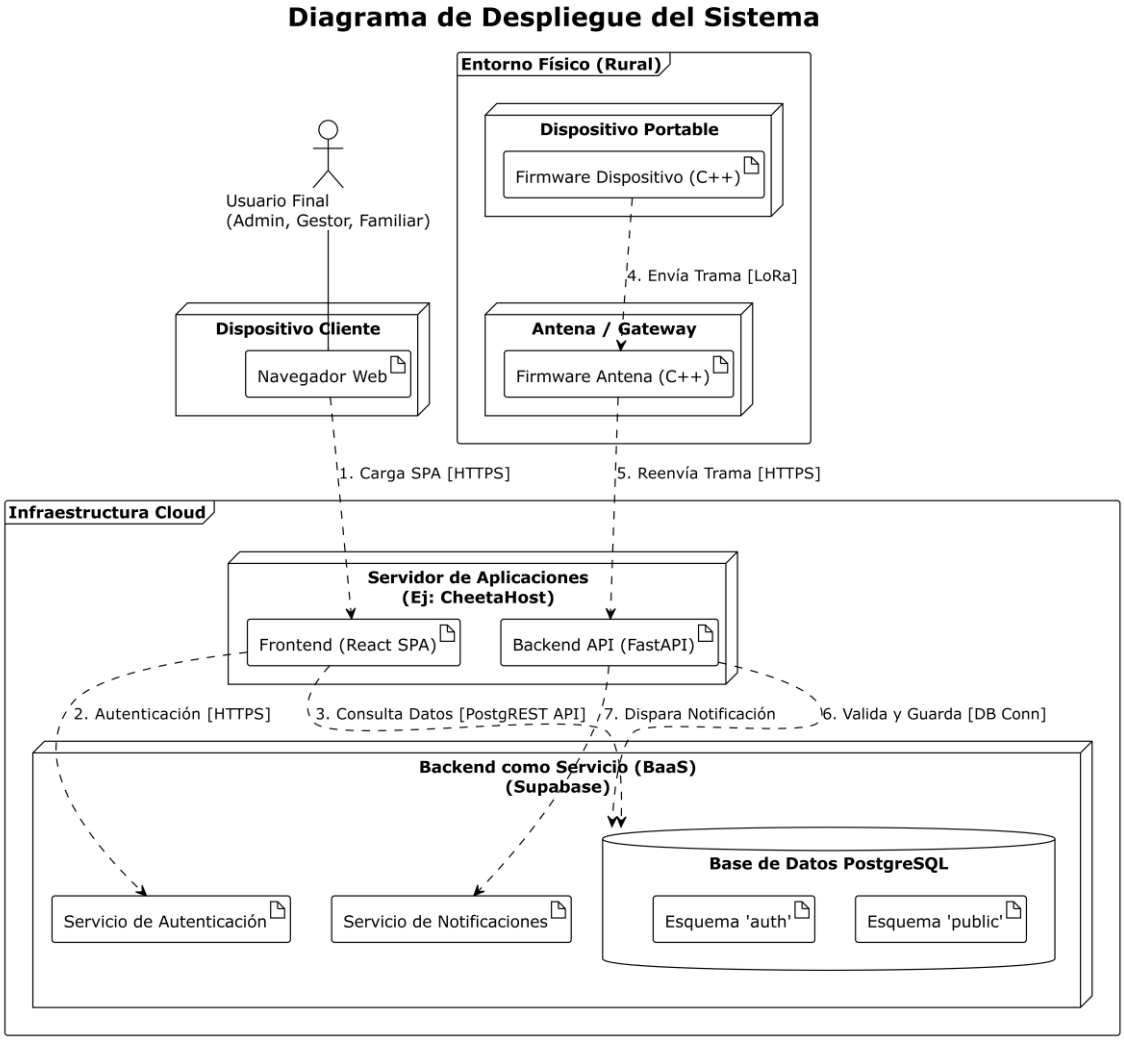


Ilustración 41 - Diagrama de Despliegue.

En el diagrama se puede apreciar la infraestructura de Supabase [31] *BaaS* utilizada para la autenticación de usuarios y él envió de notificaciones.

Tanto el *backend* de dispositivos como el servidor de *frotend* están alojados en CheetHost.com [48]. El *backend* de dispositivos y el servidor *web* funcionan insolados en contenedores *Docker* a través de virtualización. Ambos servicios se han desplegado en los servidores de CheetaHost.com a través del protocolo SFTP. Dando una solución integra sin necesidad de usar servicios extras.

El entorno físico corresponde a los dispositivos *hardware* desarrollados para el proyecto (antenas y dispositivos de usuario). Representa cómo se comunican a través de *LoRa* y la forma en la que la antena actúa como (*gateway*/puente) entre los dispositivos de usuario y el resto del sistema.

5.7.1. Funcionalidad de la aplicación

A continuación, se mostrarán algunas capturas de la aplicación *web* final. Para mayor detalle, consulte el Anexo V.

Sección pública.

En esta parte, el usuario anónimo podrá realizar diversas tareas, como consultar la página de inicio del proyecto, identificarse, darse de alta en el sistema, restablecer sus credenciales, etc.

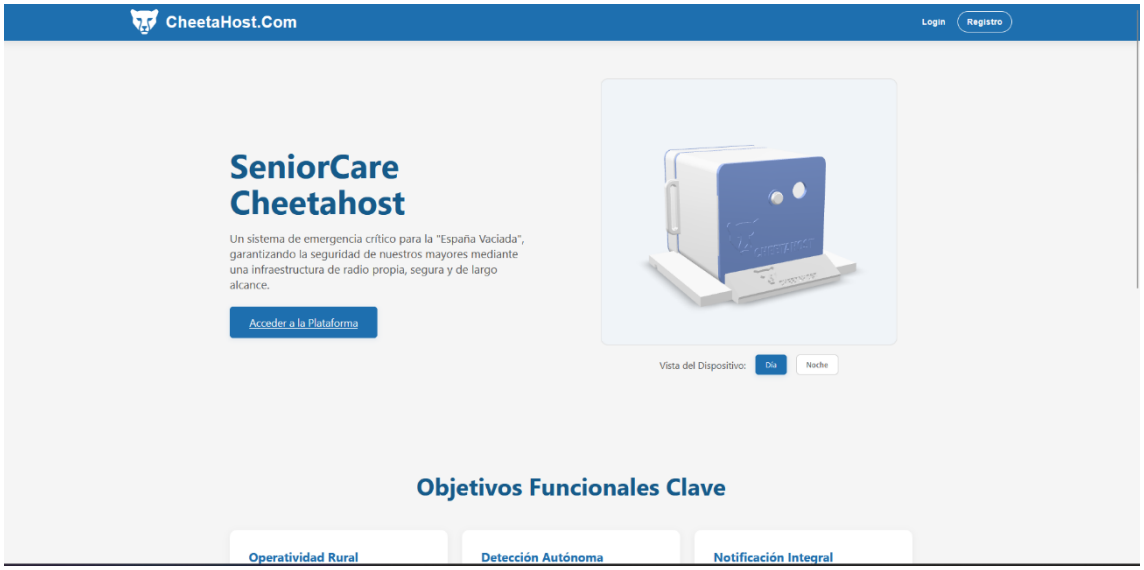


Ilustración 42 - Página de Inicio.

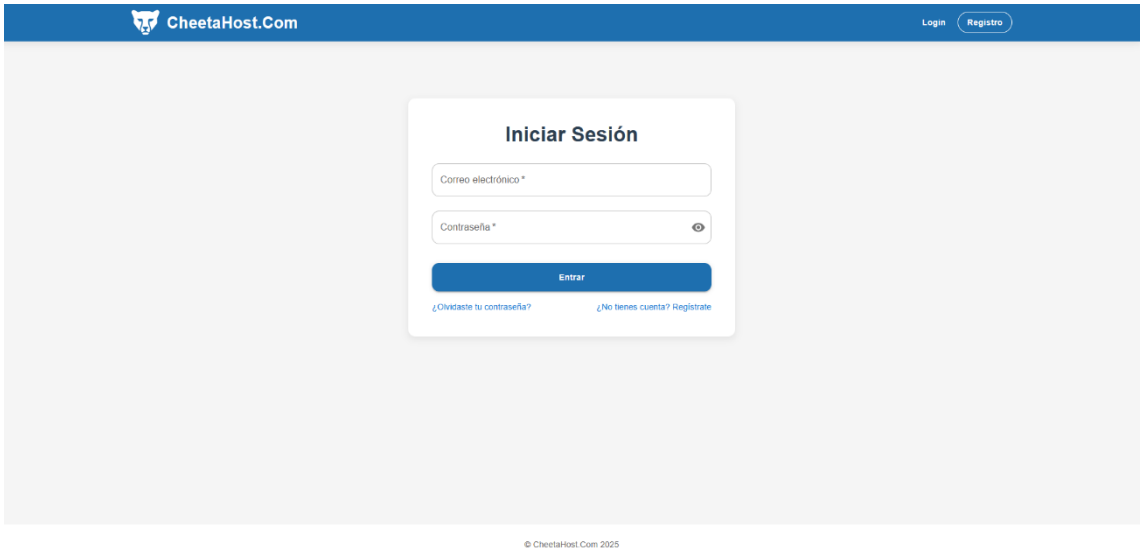


Ilustración 43 - Inicio de sesión.

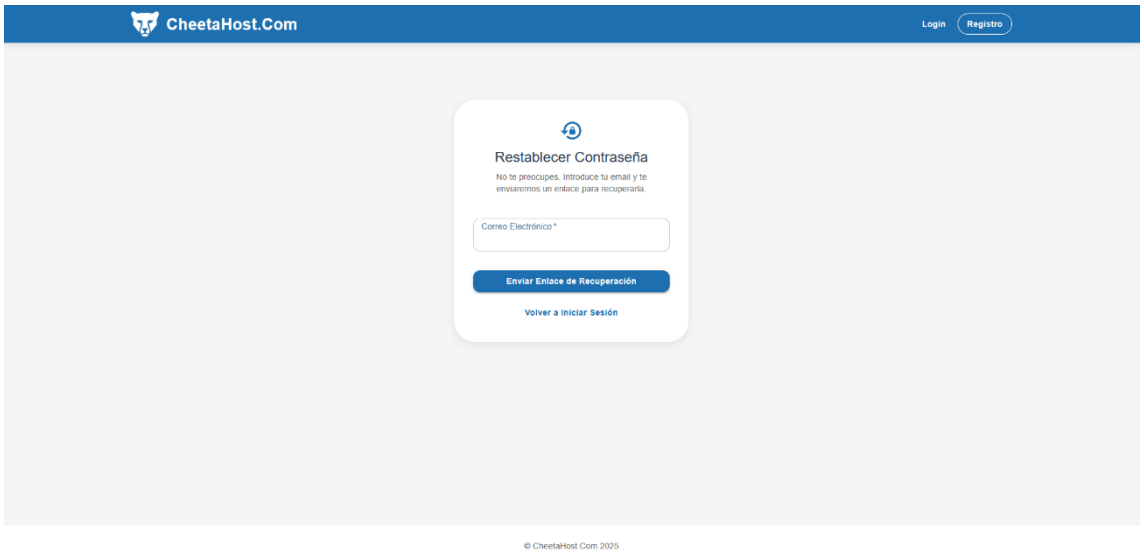


Ilustración 44 - Restablecer Contraseña.

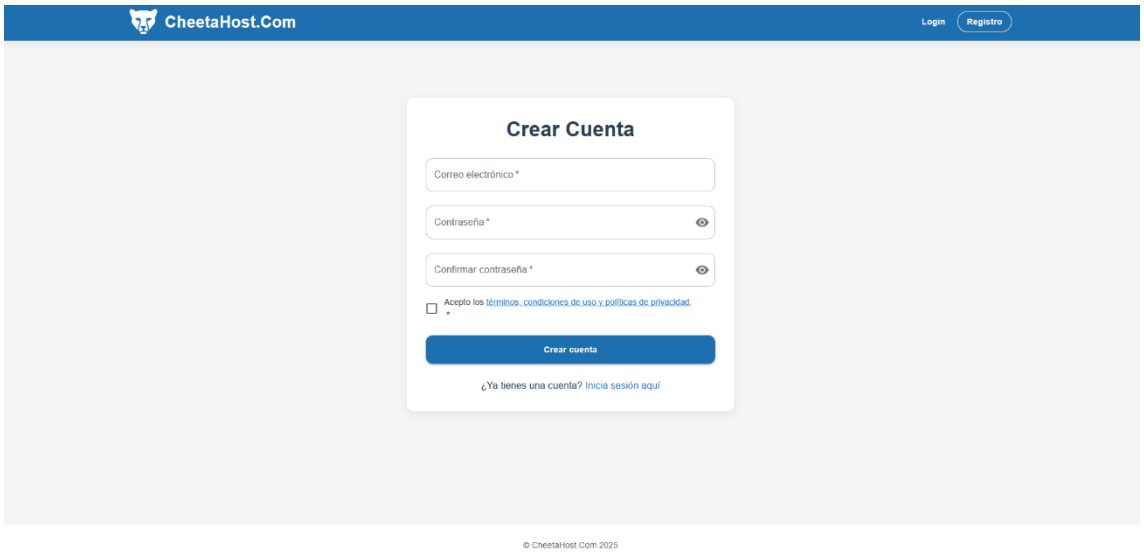


Ilustración 45 - Crear Cuenta.

Panel del Portador

Es el panel que el usuario portador de un dispositivo podrá consultar. Este le permitirá visualizar su información personal, editarla y revisar su histórico de localizaciones a continuación se mostrarán algunas capturas de dicha funcionalidad:

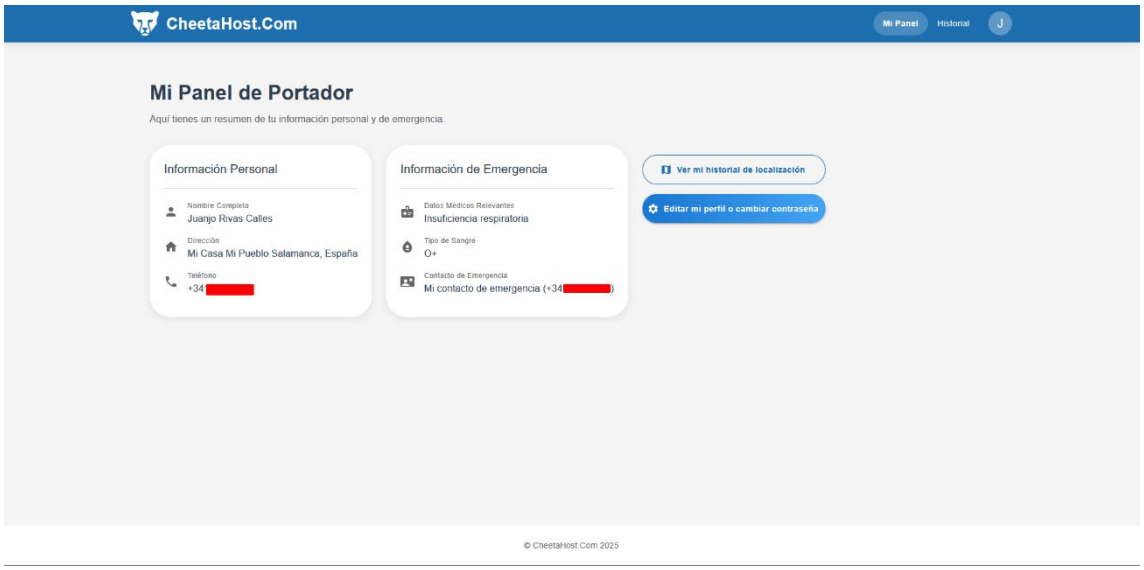


Ilustración 46 - Vista general datos portador.

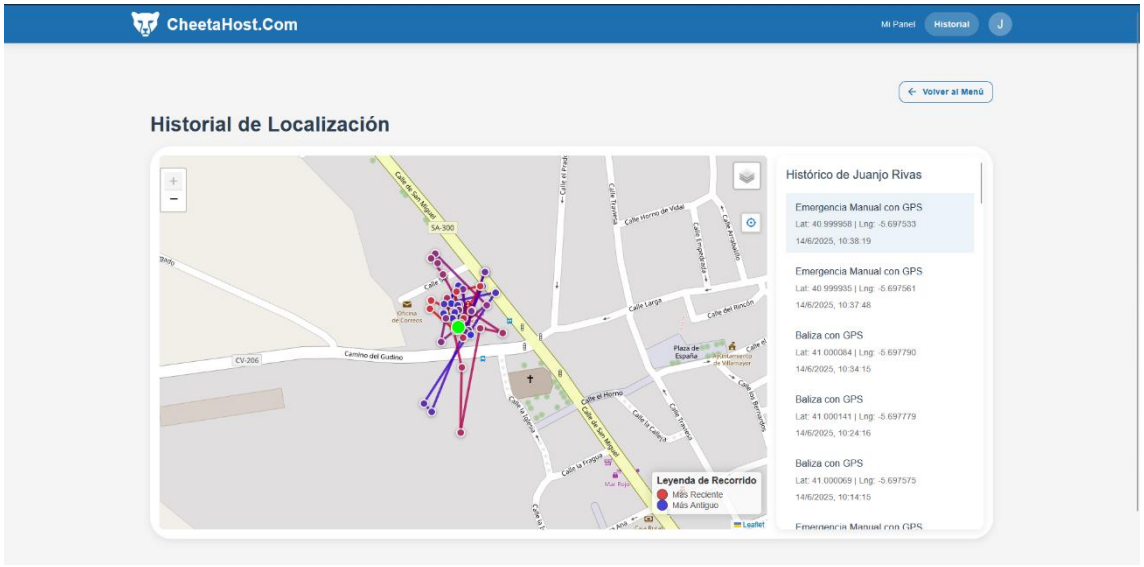


Ilustración 47 - Vista historial de sus localizaciones portador.

Panel de Familiar

En esta sección, los usuarios familiares podrán consultar los datos de sus familiares, pudiendo observar su historial de localizaciones de forma simple o detallada.

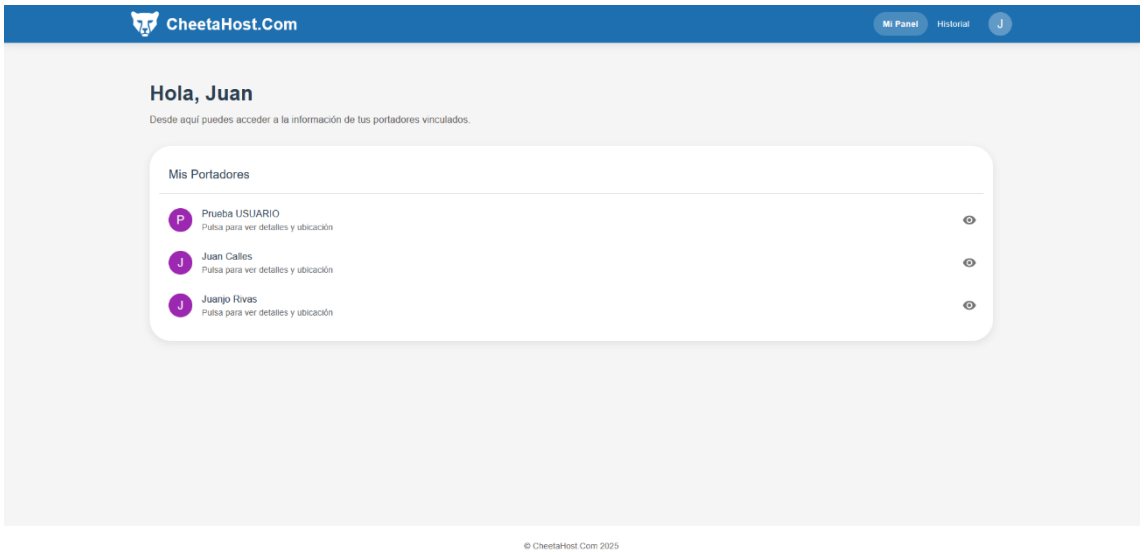


Ilustración 48 - Listado de familiares portadores disponibles.

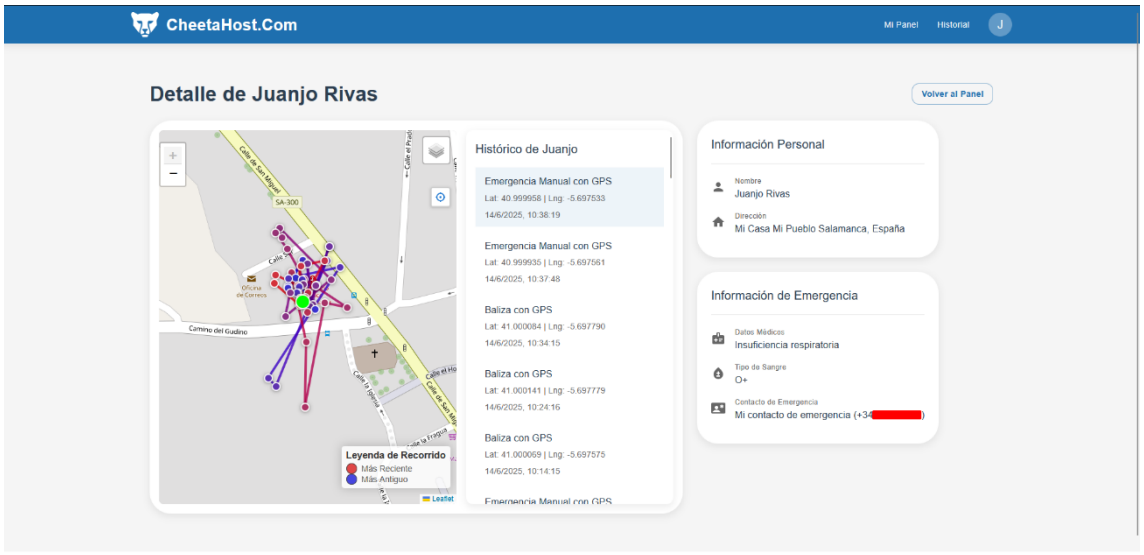


Ilustración 49 - Vista general de familiar.

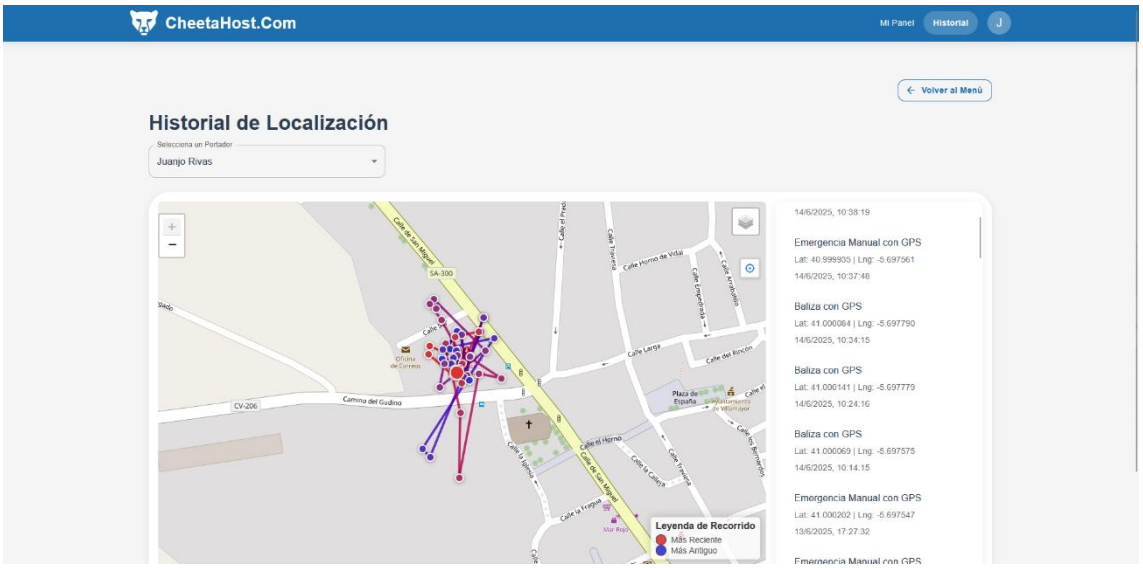


Ilustración 50 - Vista detallada localización familiar.

Panel de Emergencias

Este panel está destinado a los servicios sociales locales/emergencias. Su función principal es la gestión de las alertas e incidencias surgidas a portadores por parte del gestor de incidencias. Algunas capturas que muestran su funcionalidad son las siguientes:

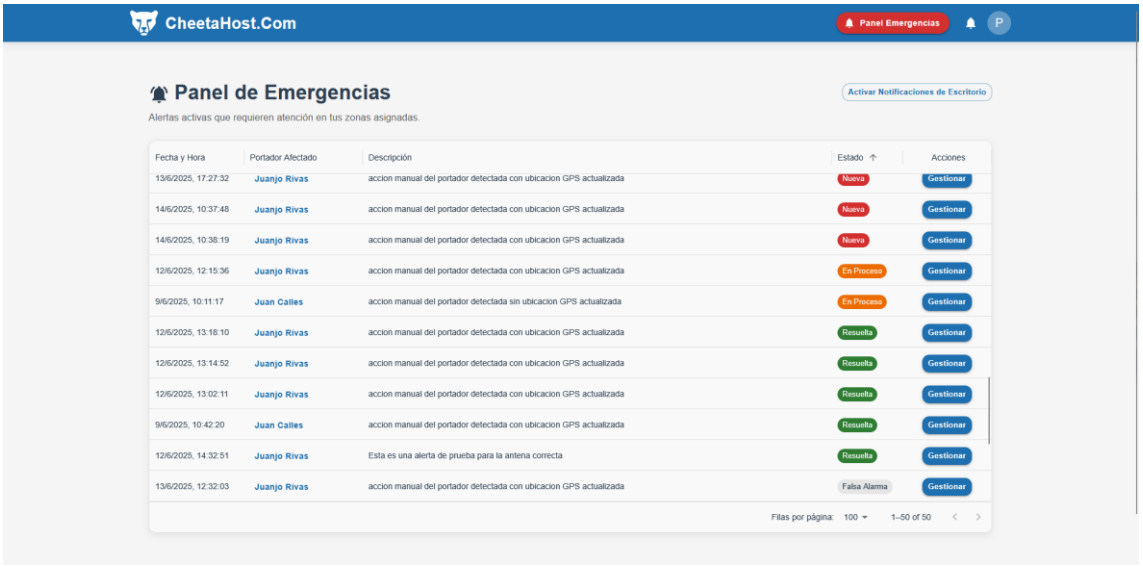


Ilustración 51 - Vista general panel emergencias.

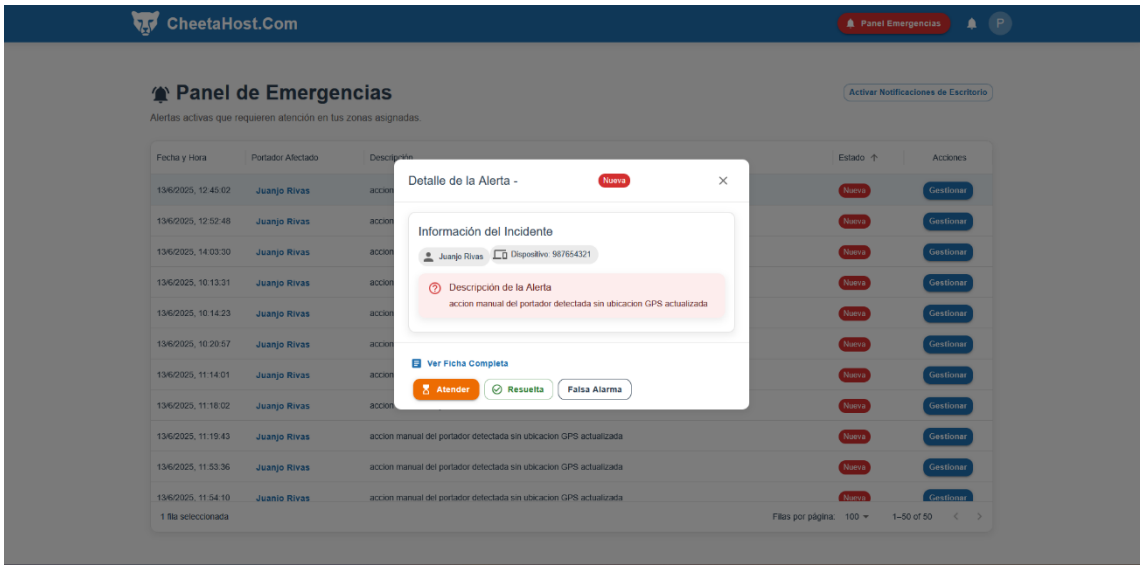


Ilustración 52 - Vista información alerta.

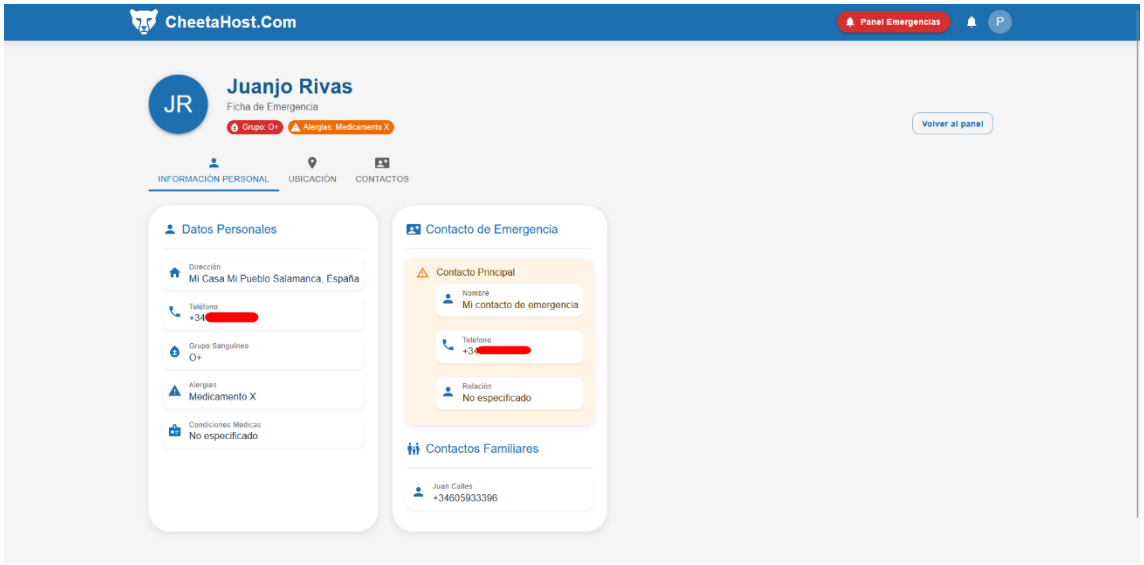


Ilustración 53 - Vista perfil emergencias portador.

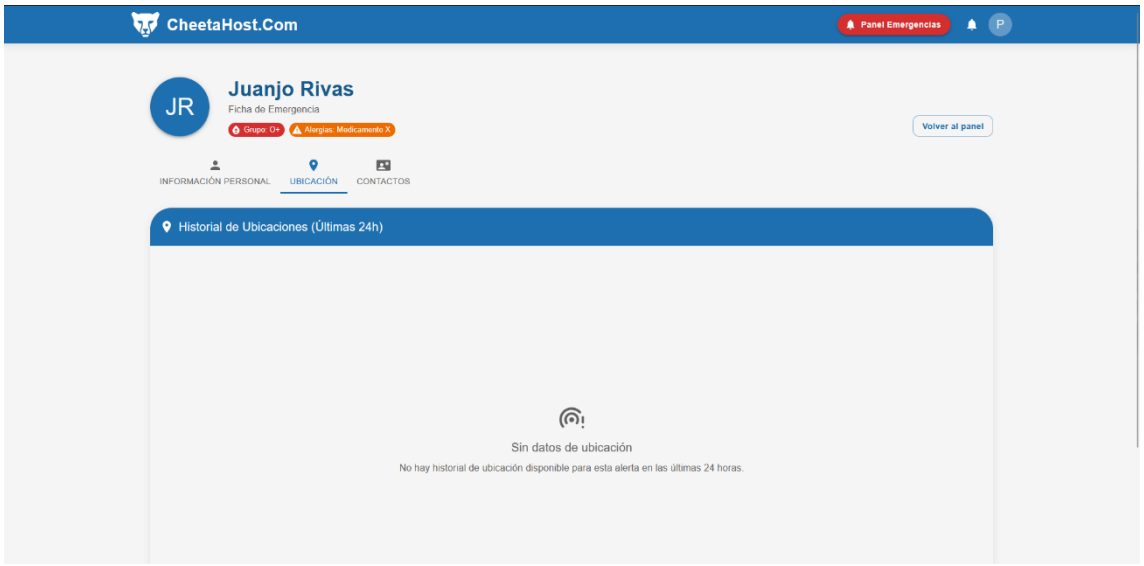


Ilustración 54 - Vista historial de ubicaciones usuario implicado en emergencia.

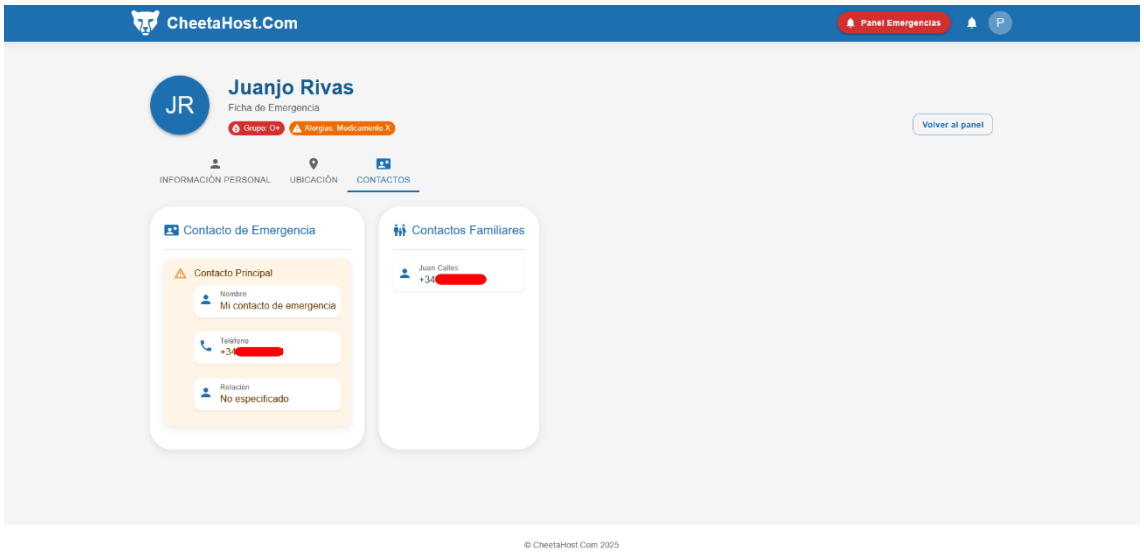


Ilustración 55 - Vista contactos usuario portador con alerta.

Panel del Administrador

En este panel se realizarán las tareas de administración del sistema entre las que destacan la gestión de dispositivos, gestión de usuarios y gestión de antenas. A continuación, se mostrarán algunas capturas de funcionalidades implementadas en el sistema.

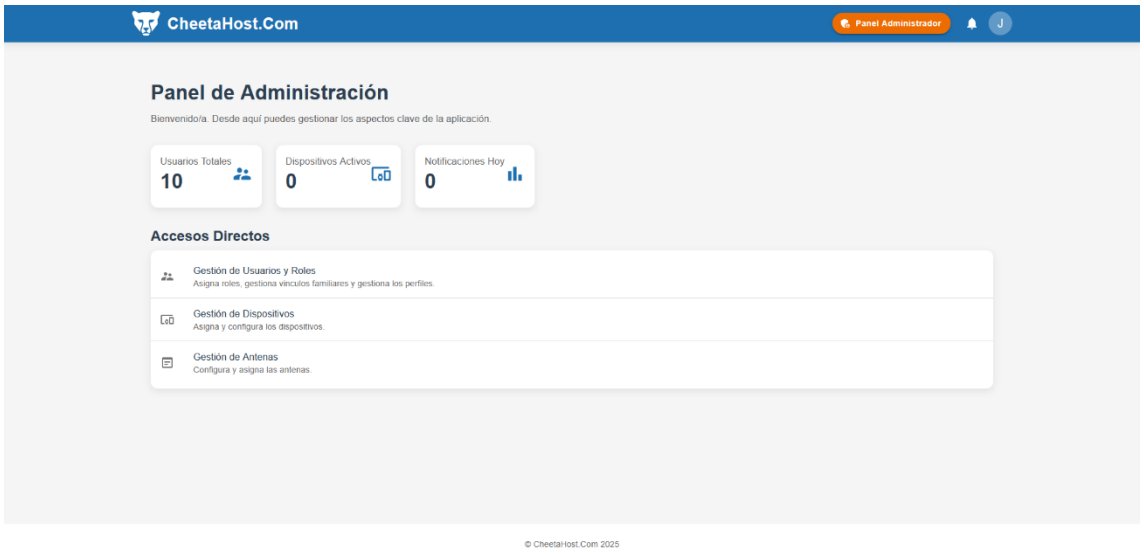


Ilustración 56 - Vista general panel administrador.

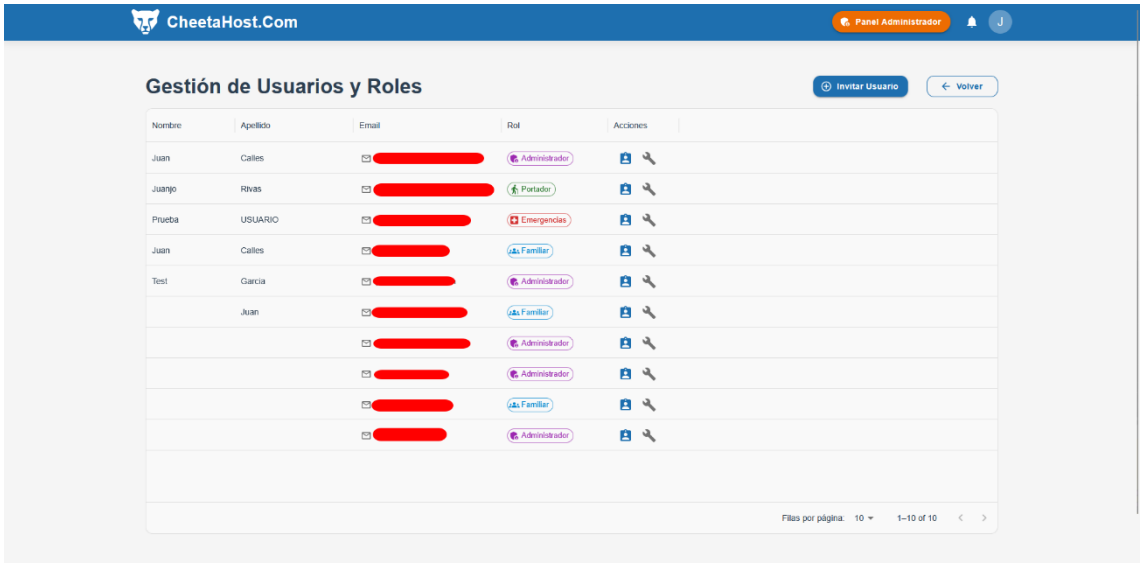


Ilustración 57 - Vista general gestión de usuarios.

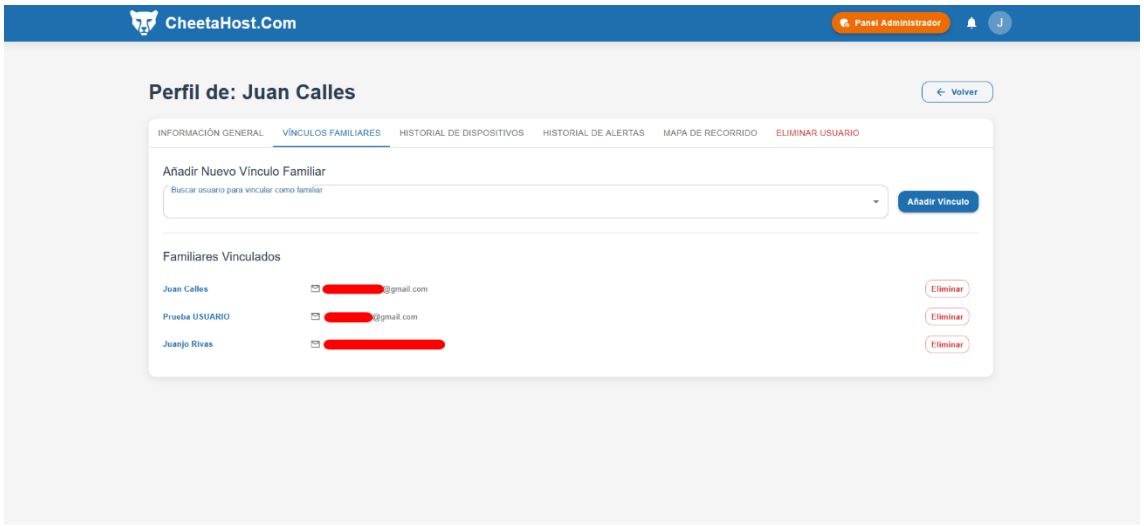


Ilustración 58 - Vista vínculos familiares usuario.

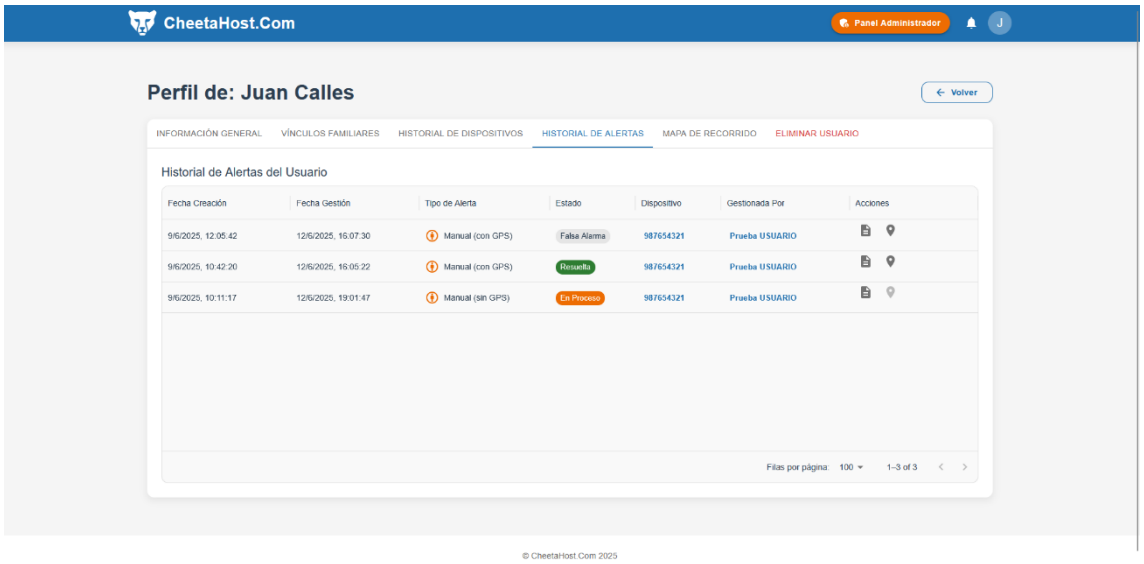


Ilustración 59 - Historial de alertas.

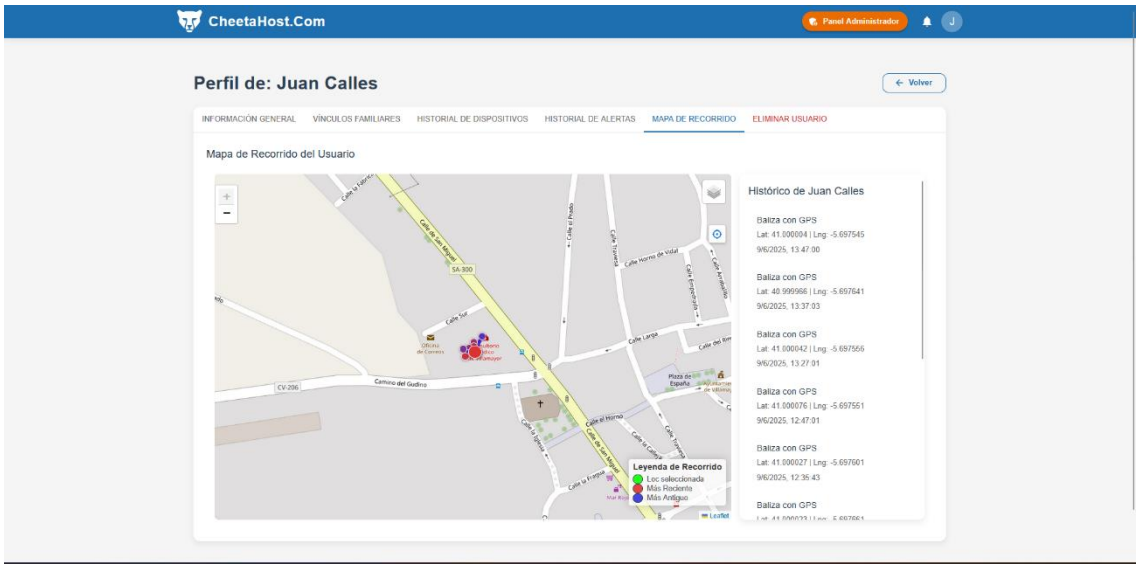


Ilustración 60 - Historial de localizaciones usuario.

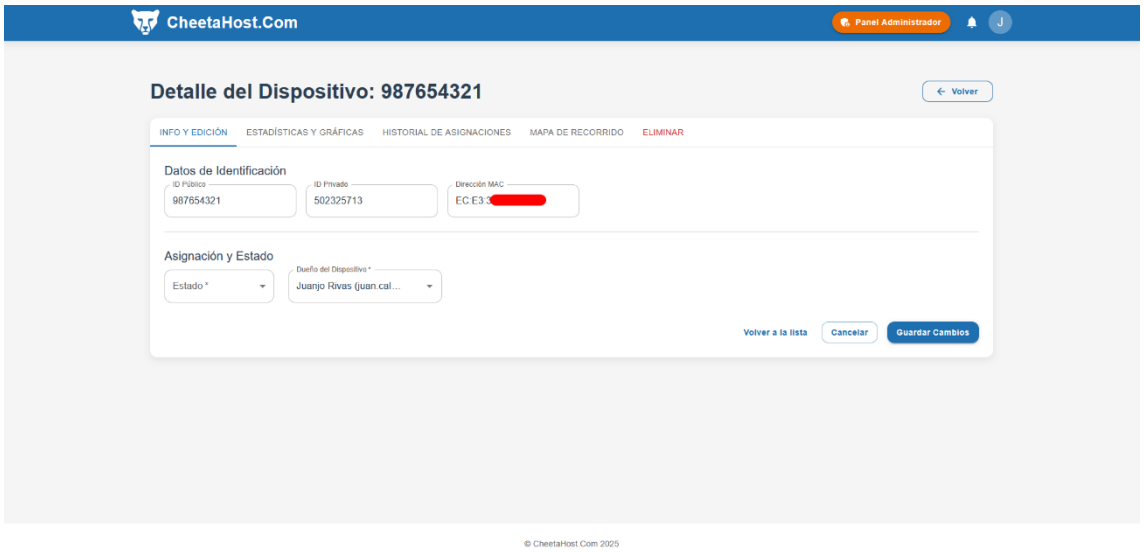


Ilustración 61 - Edición información dispositivo.

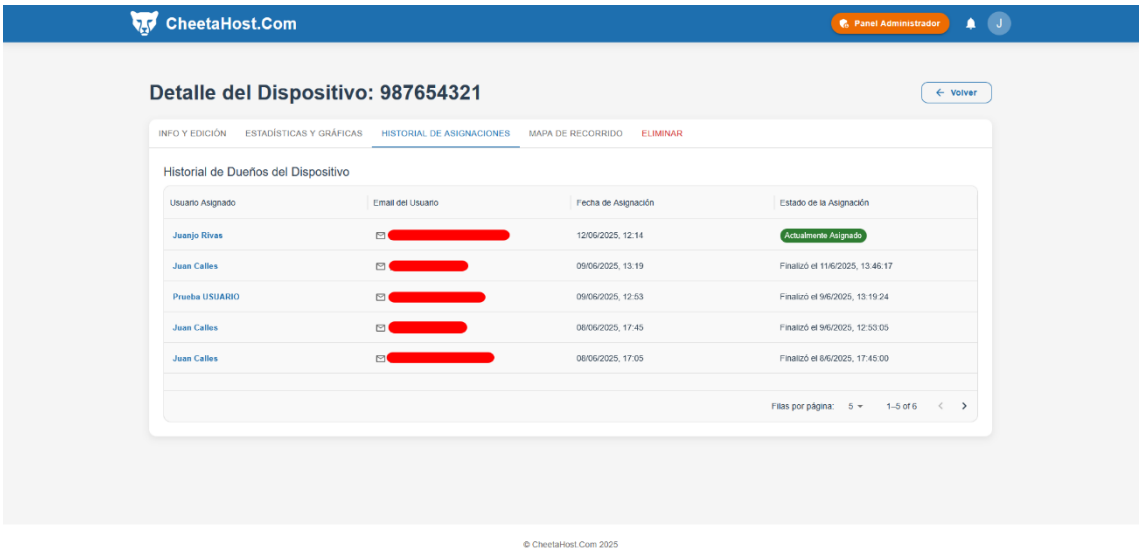


Ilustración 62 - Historial de usuarios asignados a un dispositivo.

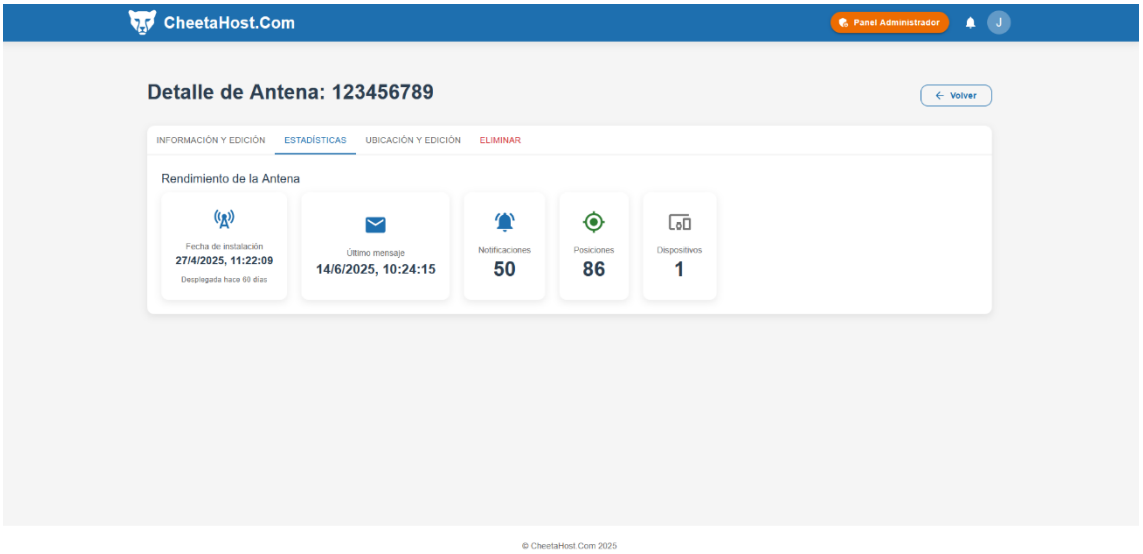


Ilustración 63 - Vista estadísticas antena.

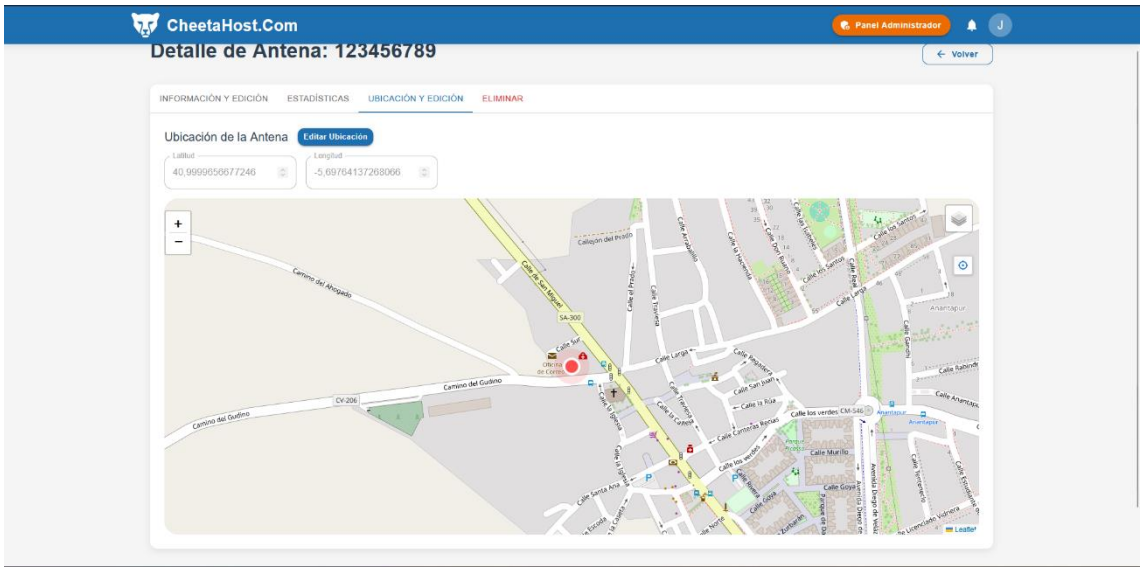


Ilustración 64 - Consulta y edición de posición de la antena.

6. Trabajos relacionados

Durante la fase de investigación para la realización del Anexo VIII (Plan de negocio TCUE), se realizó un análisis de del estado en el que se encuentra el sector de teleasistencia en España. Es cierto que existen muchas soluciones tanto públicas como comerciales, de las cuales se ha realizado un análisis de contraste de sus soluciones propuestas, pudiendo detectar en su mayoría deficiencias significativas en lo que respecta al contexto de la "España vaciada" a la cual se dirige este proyecto.

A continuación, se analizarán las principales alternativas existentes en el mercado, siendo agrupadas en dos categorías: proveedores de servicios de teleasistencia y las plataformas tecnológicas especializadas. Para cada una de ellas de señala el diferencial clave que proporciona este proyecto.

6.1. Servicios de teleasistencia comercial y pública

Esta categoría engloba las organizaciones públicas y grandes compañías que ofrecen soluciones de teleasistencia integrales, normalmente como suplemento de otros productos como servicios sociosanitarios u alarmas de seguridad.

Proveedor	Descripción del servicio	Diferencial clave del proyecto
Cruz Roja	Servicio de teleasistencia domiciliaria y móvil de gran implantación actualmente. Utiliza un pulsador de emergencia y ofrece localización *GPS*. Es uno de los servicios de referencia a nivel nacional.	**Independencia de la red móvil:** el servicio de Cruz Roja, como la mayoría, depende de la cobertura de telefonía móvil para su funcionamiento fuera del hogar, una limitación crítica en las zonas rurales a las que este proyecto da solución con su infraestructura de red *LoRa* propia.
SICOR (El Corte Inglés) / Movistar Prosegur / ADT / Securitas Direct)	Son servicios privados que integran la teleasistencia dentro de sus sistemas de alarmas para el hogar. Ofrecen pulsadores SOS, aplicaciones móviles, y algunos, como Movistar, relojes con detección de caídas.	**Modelo de negocio y accesibilidad:** estas soluciones están ligadas a costosas suscripciones mensuales de servicios de seguridad. El sistema de este proyecto está diseñado como una infraestructura de bajo coste, potencialmente gestionable por entidades locales (ayuntamientos), sin necesidad de tener vincularse a servicios externos (servicios de alarmas).
ISECO	Empresa tecnológica que ofrece un terminal para el medio rural que llama directamente a teléfonos familiares preprogramados,	**Automatización y escalabilidad:** la solución de ISECO es manual (depende de que el usuario pulse). La solución propuesta en este proyecto añade la detección

	eliminando la necesidad de una centralita de teleasistencia.	automática de caídas mediante IA, un factor crítico para los usuarios a los que va destinado. Además, la solución propuesta aplica un sistema centralizado que permite una gestión escalable por parte de servicios de emergencia/sociales locales, no solo de familiares.

Tabla 9 - Servicios de teleasistencia comercial y pública.

6.2. Dispositivos y plataformas tecnológicas especializadas

En esta sección se centra en las empresas especializadas que desarrollan soluciones comerciales de dispositivos portátiles y plataformas de monitorización de personas mayores.

Proveedor	Descripción del producto/plataforma	Diferencial clave del proyecto
SeniorDomo / Neki / Tunstall / Legrand Care	Ofrecen dispositivos específicos como relojes inteligentes, colgantes o cinturones con *GPS*, detección de caídas y comunicación directa con familiares o una central.	**Protocolo de comunicación propio (*ISHA*):** Todos estos dispositivos comerciales utilizan tecnologías estándar (4G, *WiFi*, *Bluetooth*) y protocolos propietarios de "caja negra". La fortaleza del proyecto reside en el propio protocolo *ISHA*, que garantiza un control absoluto sobre la seguridad y privacidad de los datos. Es una solución abierta y auditable, no una solución comercial cerrada.
VitalOn / Qida	Son plataformas que combinan la tecnología (monitorización de constantes, actividad) con la prestación de servicios de cuidado domiciliario profesional.	**Enfoque en la infraestructura, no solo en el servicio:** mientras estas empresas se centran en el modelo de negocio del cuidado, el proyecto se enfoca en resolver el problema tecnológico de base a la conectividad en las zonas rurales (red de radiofrecuencia propia).
Kwido	Son consultores y desarrolladores de *software* de IA e IoT que se integran con diferentes fabricantes de *hardware* para ofrecer soluciones de teleasistencia.	**Solución integral y vertical:** Kwido ofrece el *software*, pero depende del *hardware* de terceros. El proyecto propuesto es una solución vertical completa, se ha diseñado el *hardware*, el *firmware*, el protocolo de comunicación y la

	plataforma de gestión. Esto permite una optimización total del sistema que no es posible al integrar componentes de diferentes proveedores.

Tabla 10 - Dispositivos y plataformas tecnológicas especializadas.

7. Problemas en el desarrollo

Durante la realización del proyecto, se han sufrido problemas y dificultades técnicas, los cuales se expondrán los de un grado mayor de relevancia en esta sección. Además, se plantearán las posibles líneas de trabajo futuras.

7.1. Problemas con componentes *hardware*

Debido a la falta de experiencia del alumno en el área de diseño y desarrollo de componentes *hardware*, se han sufrido diferentes inconvenientes entre los que destacan la falta de herramientas, problemas con módulos *LoRa*, problemas al afrontar el diseño de la PCB, entre otros. A continuación, se detallan con mayor detalle todos ellos.

7.1.1. Falta de herramientas

Durante el proceso de montaje de los primeros módulos de radiofrecuencia *LoRa* (descartados por la emisión en una banda no permitida en España), el alumno utilizó las herramientas que tenía en casa para poder montar dichos módulos, la cual consistía en un soldador de cable para instalaciones eléctricas. Este hecho sumado a la inexperiencia propiciaba que el proceso de montaje se prolongue alrededor de las seis horas de duración. Además de incluso llegar a dañar algún modulo (los módulos *LoRa* utilizados en el proyecto no siguen el estándar de las placas de prototipado, necesitando muchísima más precisión durante el proceso de montaje).

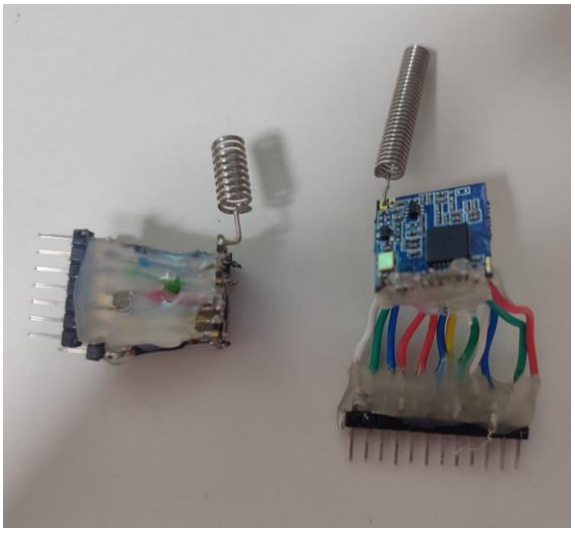


Ilustración 65 - Módulos LoRa ensamblados sin herramientas ni técnicas especializadas.

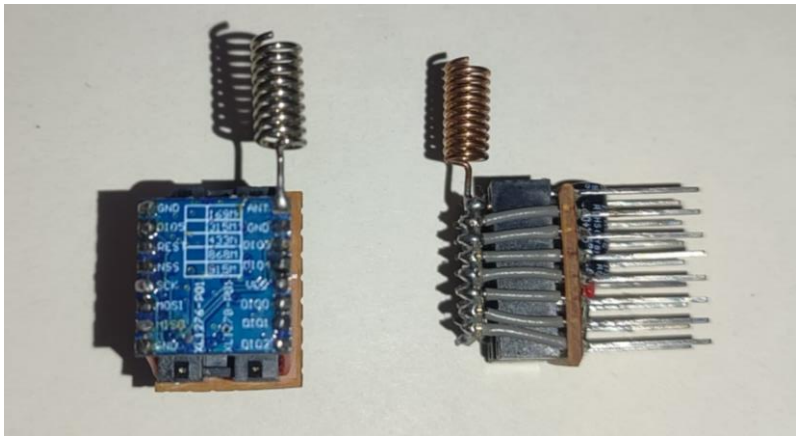


Ilustración 66 - Módulos LoRa ensamblados con herramientas y técnicas especializadas.

7.1.2. Problema modulo *LoRa*

Para la realización del proyecto se utilizaron finalmente los SX1276 montados en un zócalo adaptador gracias a la ayuda del Técnico de Laboratorio de Física Aplicada, José Antonio García García. Tras realizar numerosas pruebas durante la implementación básica de emisión de mensajes, se detectó un bloqueo del dispositivo tras el intento de envíos consecutivos, dejando al sistema colgado. Este ha sido el error que sin duda más problemas ha causado. La solución inicial fue cambiar el modo de emisión a asíncrono no bloqueante, evitando el bloqueo de la ejecución del *firmware*. Se observó que no funcionaba tampoco del todo bien, viendo que la lógica del *firmware* estaba correcta, se investigó en la red encontrándose con la solución en un foro de *GitHub* [49] donde se comentaba que el módulo *LoRa* tras la emisión de paquetes, generaba oscilaciones pudiendo reiniciar el módulo y desconfigurándolo, causando un mal funcionamiento. Tras la modificación del zócalo añadiendo un condensador de 220uF en paralelo al módulo, esta incidencia fue resuelta permitiendo continuar con el proyecto.

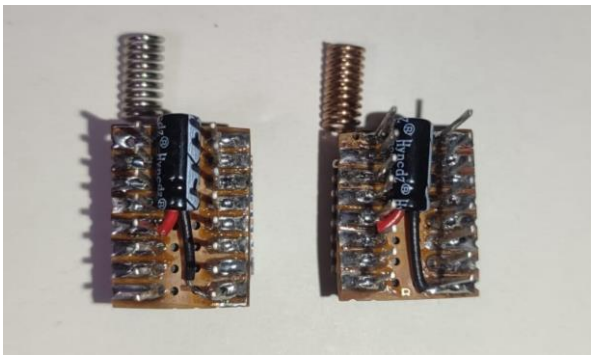


Ilustración 67 - Esquema de montaje del condensador.

7.1.3. Inconveniente de diseño en la PCB

Dada la nula experiencia previa del alumno en el desarrollo de *hardware, a* la hora de la realización del circuito, se seleccionaron resistencias del tipo SMD (*Surface Mount Device*) en vez de la selección de resistencias de tipo axial que dificultaron el montaje, teniendo que pedir ayuda durante este proceso a José Antonio García García.

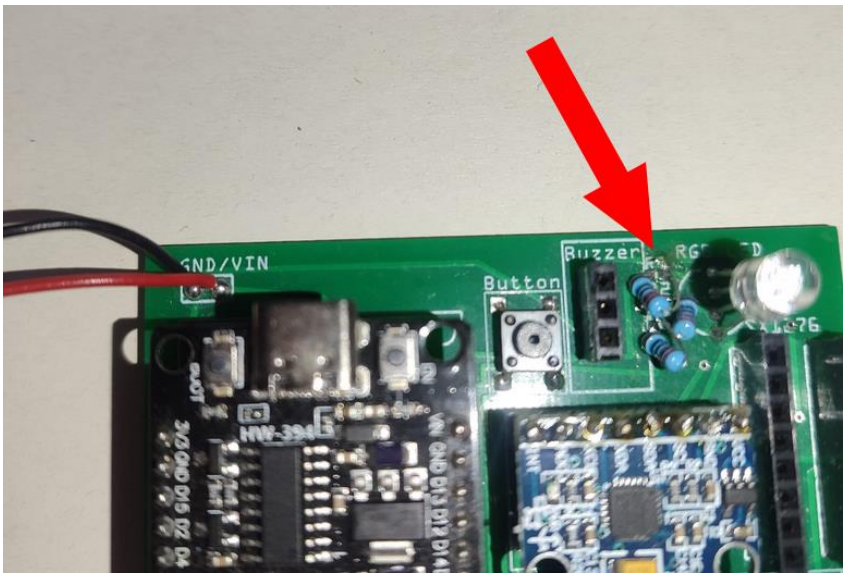


Ilustración 68 - Resistencias axial soldada en interfaz SMD.

7.1.4. Descomposición de componentes.

El alumno, a causa de un despiste al no aplicar cinta aislante a los conectores de la PCB, permitió que estos entraran en contacto, produciendo un cortocircuito que afectó a una de las placas ESP-32, quemándole un diodo de protección y dejándola inutilizada hasta que se realizara su respectiva reparación.

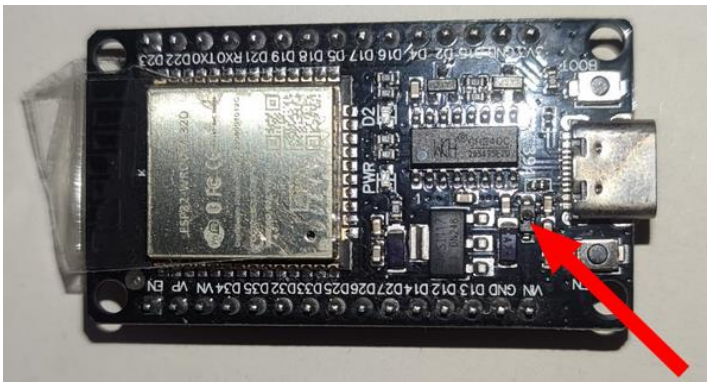


Ilustración 69 - ESP-32 dañado.

7.1.5. Diseño 3D profesional.

El alumno tenía experiencia en el desarrollo de modelos 3D artísticos de forma proactiva pero no de forma profesional. Por lo que, para evitar programas con una gran curva de aprendizaje, se optó por utilizar la herramienta de diseño 3D de *Tinkercad*. Permitiendo realizar el diseño completo de la carcasa del dispositivo en 48 horas. Esta versión final contaba con algunos fallos de diseño salvables a través de devastación de material, pero otros inviables por lo que, tras otras 12 horas de modificaciones, se realizó la versión 2.0 de la carcasa y base de carga, cumpliendo con las expectativas previstas.

Imágenes de las versiones erróneas

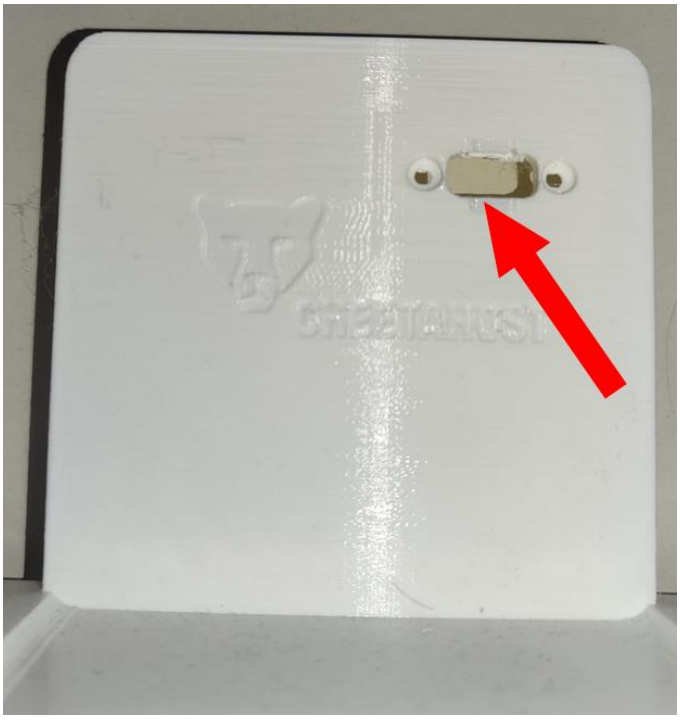


Ilustración 70 - Base de carga con conector no correspondiente.

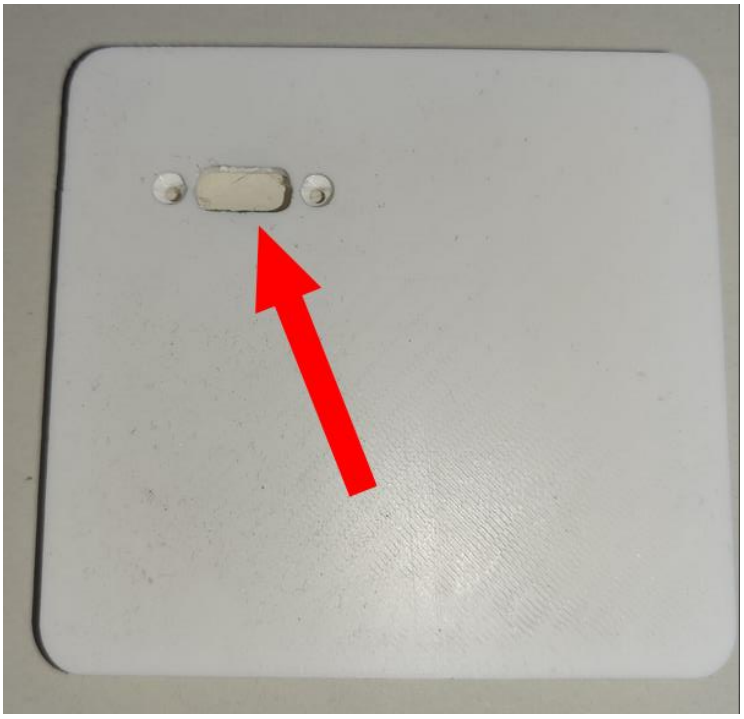


Ilustración 71 - Tapadera de carga con conector no correspondiente.

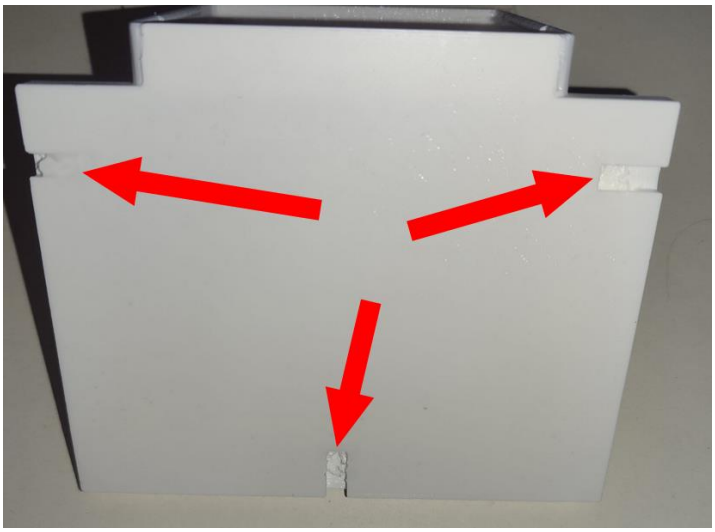


Ilustración 72 - Orificio fallido a causa de soportes de impresión y fallo de diseño.

Ilustración 73 - Medidas erróneas y poco refuerzo.

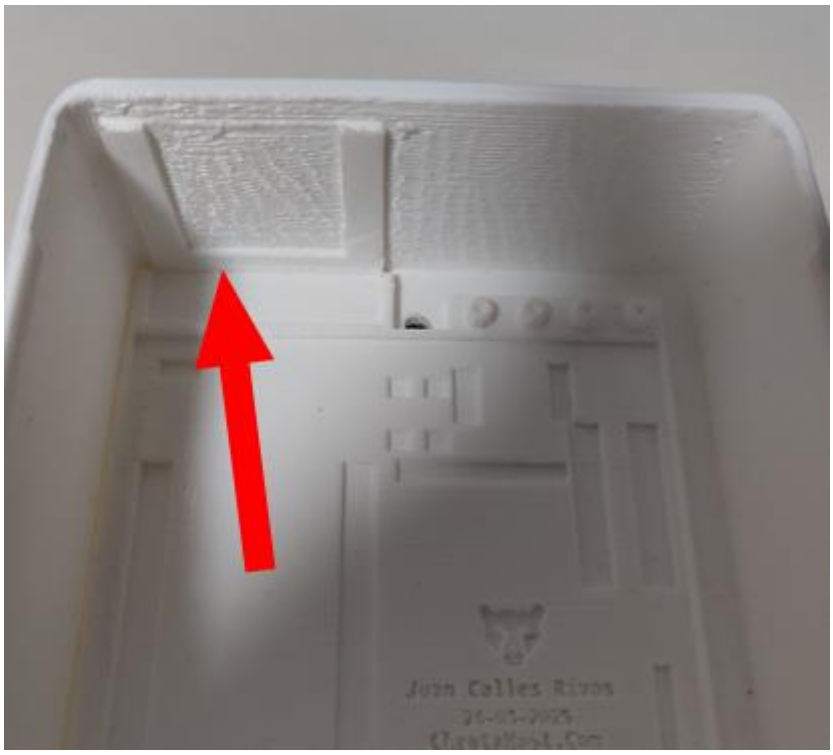


Ilustración 74 - Soporte superior antena GPS demasiado grande.

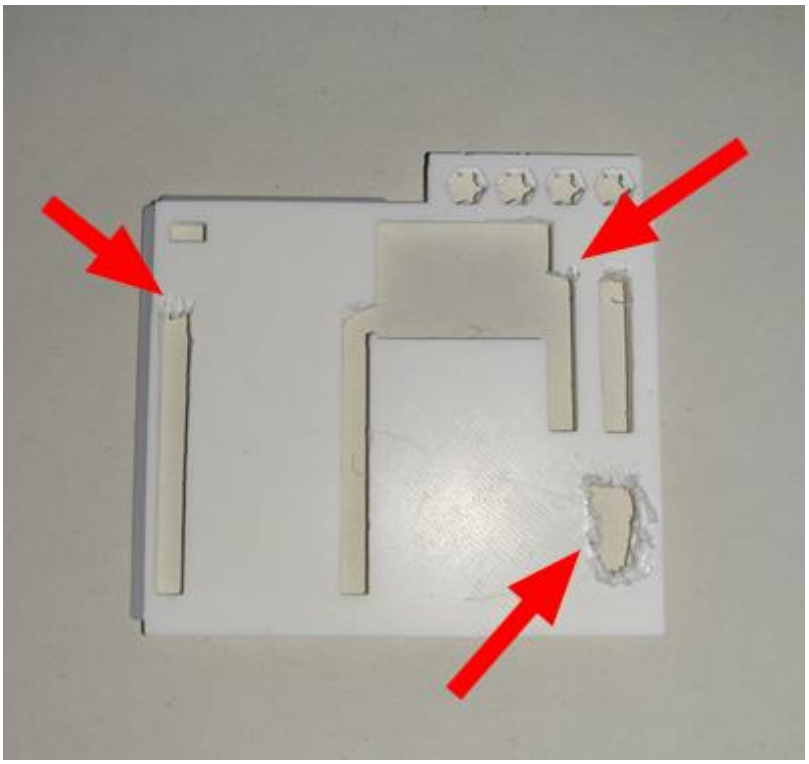


Ilustración 75 - Escudo de PCB mal diseñado (modificado para reutilización).

Imágenes de la versión corregida

Ilustración 76 - Base de carga con conector correspondiente.

Ilustración 77 - Tapadera de carga con conector no correspondiente.

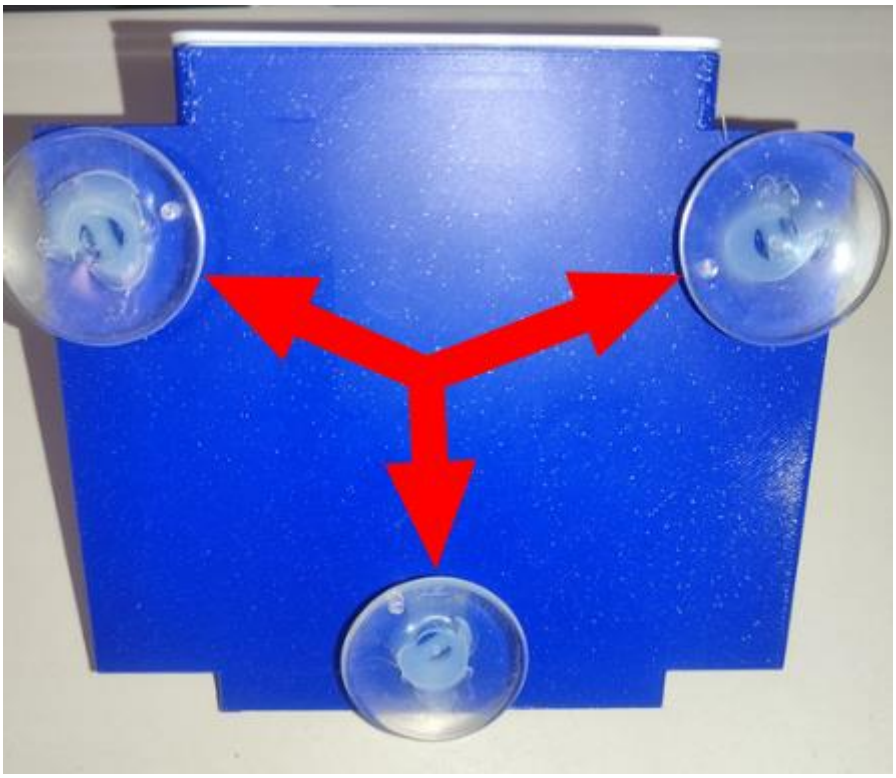


Ilustración 78 - Orificio de sujeción correctos para la instalación de ventosas.

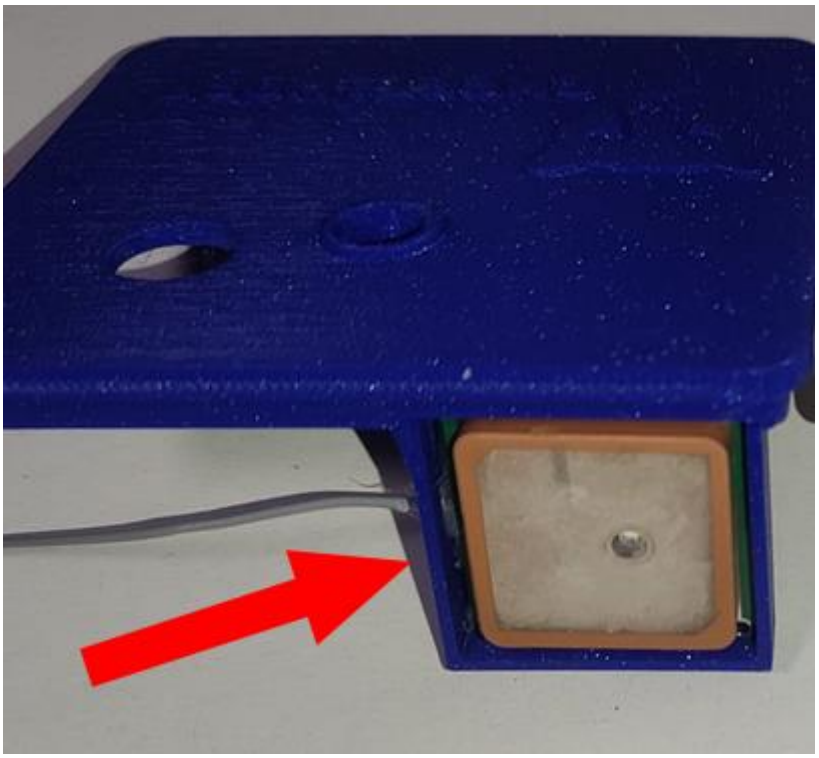


Ilustración 79 - Medidas correctas y con refuerzo.

Ilustración 80 - Reducción del soporte superior antena GPS.

Ilustración 81 - Escudo PCB correcto (debajo del hardware).

7.1.6. Demora de compilación del *firmware* en Windows.

La compilación del *firmware* en el sistema operativo Windows empezó a ser un problema en el momento que el tamaño del *firmware* fue aumentando, teniendo un tiempo de espera para la compilación y grabado del *firmware* de unos 5 minutos siendo inasumible durante el desarrollo por la gran cantidad de cambios que hay que aplicar durante el desarrollo de un sistema así. La solución fue migrar el desarrollo del *firmware* a un sistema Linux donde el tiempo de espera se reducción de los 5 minutos que tardaba en Windows a apenas 20 segundos.

8. Conclusiones

La realización de este Trabajo de Fin de Grado ha culminado en la creación de un sistema de emergencia integral, prototípico, pero funcional, diseñado para dar una solución tecnológica robusta y viable a la problemática de la teleasistencia que sufren las personas mayores en entornos rurales con conectividad deficiente, en la denominada "España Vaciada". Como resultado final se ha materializado el sistema, demostrando la viabilidad de construcción de una infraestructura de teleasistencia eficaz, segura y de bajo coste, independiente de las redes de telefonía móvil convencionales.

Todos los objetivos técnicos definidos han sido completados satisfactoriamente, validando la arquitectura y las tecnologías seleccionadas para el proyecto:

- **Gestión de usuarios y autenticación:** se ha implementado un sistema completo que gestiona diferentes roles (Portador, Familiar, Gestor, Administrador) con vistas y permisos específicos en el sistema. La autenticación se maneja de forma segura tanto para los usuarios de la plataforma *web* como para los dispositivos *hardware* (antenas y dispositivos de usuario), utilizando la infraestructura de Supabase.
- **Comunicación segura:** se ha logrado el cifrado de extremo a extremo entre el dispositivo de usuario y el servidor mediante el diseño e implementación del protocolo ***ISHA* (Inter-System Hybrid Alerting)**. Este protocolo utiliza cifrado simétrico AES-256 con claves de rotación diaria y vectores de inicialización únicos por trama, garantizando la confidencialidad, integridad y privacidad de todos los datos sensibles transmitidos sobre *LoRa* e Internet
- **Detección autónoma de caídas**: se ha desarrollado un modelo de Inteligencia Artificial capaz de distinguir “caídas” de “actividades cotidianas”. El modelo consiste en un ensamblado ligero de dos árboles de decisión, los cuales realizan la detección de “caída” a través de un sistema de votación por acumulación, esta es una solución eficiente para operar con los recursos limitados del microcontrolador. Tras una fase de exploración inicial con *datasets* públicos, se determinó que la máxima fiabilidad se obtenía entrenando el modelo exclusivamente con un *dataset* propio, grabado con el mismo hardware del dispositivo. Esta decisión fue clave para minimizar los falsos positivos y asegurar la robustez del sistema en condiciones reales.
- **Portabilidad y funcionalidad del dispositivo:** se ha diseñado y fabricado una PCB especifica y una carcasa y base de carga mediante impresión 3D, logrando un dispositivo portable dentro del marco del prototipado, aunque existe margen de miniaturización, este cumple con los requisitos definidos para el proyecto. El dispositivo integra correctamente un pulsador para realizar alertas manuales, *GPS* para geolocalización, y un sistema de indicadores visuales (LED RGB) y sonoros (buzzer).

- **Gestión centralizada (Plataforma *Web*):** se ha desarrollado una aplicación *web* completa que permite la administración total del sistema, pudiendo realizar todas las operaciones administrativas posibles tal y como la gestión de usuarios y sus vínculos familiares, asignación de dispositivos, monitorización de antenas y la

gestión íntegra de las alertas por parte de los servicios de emergencia, tal y como se demuestra las capturas de la interfaz final.

Los objetivos personales alcanzados, al igual que los técnicos, finalmente fueron todos. El alumno fue capaz de diseñar el sistema *hardware* que resolviera las necesidades del proyecto a partir de componentes convencionales, y además fue capaz de diseñar y fabricar un circuito integrado, no perfecto pero funcional. La programación del *firmware* de los dispositivos fue exitosa, permitiendo que el sistema funcionara y que los dispositivos *hardware* trabajaran de formas distintas bajo el mismo *firmware* dependiendo de su modo de operación. Se considera que el alumno hizo uso de los conocimientos obtenidos durante el desarrollo del grado para la realización del proyecto.

Todos estos objetivos, se han cumplido tras una gran perseverancia y determinación con el ánimo de poder sacar el proyecto adelante.

La realización del proyecto ha sido bastante extensa, compleja y multidisciplinar pudiendo abrumar al alumno en bastantes situaciones, pero finalmente se ha conseguido materializar en forma de una solución funcional.

Todo el esfuerzo del diseño, desarrollo, construcción de *hardware* habrá merecido la pena si en algún momento el proyecto pudiera evolucionar a un producto final pudiendo cumplir su objetivo principal el de ayudar a personas mayores/dependientes de los entornos rurales ya que con ese fin fue concebido.

La realización de este ha sido bastante enriquecedora para el alumno demostrando realmente la capacidad multidisciplinar adquirida durante el desarrollo del grado y también gracias a su proactividad en cuanto respecta a la búsqueda de soluciones.

8.1. Líneas a futuro

En esta sección se comentará la funcionalidad extra que se implementaría a futuro para mejorar el sistema.

8.1.1. Protocolo *ISHAv2*

Para esta nueva versión se aplicarían los conocimientos adquiridos durante la realización de la asignatura de Sistemas Distribuidos [50], no solamente utilizando a las antenas como enrutadores, sino también a otros dispositivos disponibles para poder generar una red descentralizada *LoRa* de mayor rango de acción con la misma o menos infraestructura.

También se plantearía la simplificación del sistema criptográfico ya que es bastante complejo, implantando uno más sencillo ya sea de clave asimétrica (curvas elípticas [51]) o de intercambio de claves Diffie-Hellman [52].

8.1.2. Reducción del *hardware*

Actualmente el prototipo es bastante grande, esto es debido al planteamiento de montaje ya que los componentes no están soldados directamente a la PCB produciendo un aumento de tamaño del 40%. Además, la batería está conformada por dos 18650 que dan al dispositivo autonomía de varios días, la idea sería reducir a 1 batería dando la autonomía de una jornada completa de trabajo reduciendo el peso del dispositivo.

Para la reducción aún más del dispositivo se optará por cambiar el microcontrolador ESP-32-WROOM por el Arduino Nano ESP.

A continuación, se mostrarán algunas imágenes del dispositivo final y si base de carga:

Ilustración 82 – Dispositivos de Usuario.

Ilustración 83 - Dispositivo de usuario encendido.

Ilustración 84 - Base de carga de dispositivo de usuario.

Ilustración 85 - Carcasa contenedora antena IP67.

Ilustración 86 - Cierre de cinturón.

Ilustración 87 - Dispositivo siendo portado por usuario.

8.1.3. Adición de altavoz

El actual sistema consta con un *buzzer* que realiza señales sonoras, lo cual puede ser un poco molesto para algunos usuarios por lo que la adicción de un altavoz con un mensaje de audio grabado en castellano les permitiría entender con más facilidad y calma la situación e incluso calmar al usuario durante ese periodo de estrés.

8.1.4. Sistema de retroalimentación

Crear una interfaz capaz de extraer datos de la MPU utilizados exclusivamente para la generación de modelos de predicción de caídas más precisos y efectivos. Permitiendo mejorar el sistema y hacerlo más seguro para los usuarios.

9. Referencias

[1] "Indicadores Demográficos Básicos, Comunidades y Ciudades Autónomas, 65 y más años, 2024." Accessed: Jul. 02, 2025. [Online]. Available: https://ine.es/jaxiT3/Datos.htm?t=1451#_tabs-mapa

[2] "Indicadores Demográficos Básicos, Provincias, 65 y más años, 2024." Accessed: Jul. 02, 2025. [Online]. Available: https://ine.es/jaxiT3/Datos.htm?t=1451#_tabs-mapa

[3] "Tasa de Dependencia de la población mayor de 64 años, por comunidad autónoma", Accessed: Jul. 02, 2025. [Online]. Available: https://ine.es/jaxiT3/Datos.htm?t=1488#_tabs-grafico

[4] "Defunciones por causas (capítulos), sexo y grupos de edad", Accessed: Jul. 02, 2025. [Online]. Available: https://ine.es/jaxiT3/Tabla.htm?t=6609

[5] John Scourias, "Overview of the Global System for Mobile Communications," *University of Waterloo*, May 1995, Accessed: Jul. 02, 2025. [Online]. Available: https://www.gbppr.net/scanner/gsm.pdf

[6] N. Korkmaz Yaylagul *et al.*, "Trends in Telecare Use among Community-Dwelling Older Adults: A Scoping Review," *Int J Environ Res Public Health*, vol. 19, no. 24, p. 16672, Dec. 2022, doi: 10.3390/ijerph192416672.

[7] "Mimov." Accessed: Jul. 02, 2025. [Online]. Available: https://www.mimov.com/

[8] "essencesmartcare", Accessed: Jul. 02, 2025. [Online]. Available: https://www.essencesmartcare.com/es/

[9] J. M. Graves, D. A. Abshire, S. Amiri, and J. L. Mackelprang, "Disparities in Technology and Broadband Internet Access Across Rurality," *Fam Community Health*, vol. 44, no. 4, pp. 257–265, Oct. 2021, doi: 10.1097/FCH.0000000000000306.

[10] Fakhruddin Mangkusasmito, Dista Yoel Tadeus, Heru Winarno, and Eko Winarno, "Peningkatan Akurasi Sensor GY-521 MPU-6050dengan MetodeKoreksiFaktor Drift," Nov. 2020, Accessed: Jul. 02, 2025. [Online]. Available: https://ejournals.umn.ac.id/index.php/SK/article/view/1791

[11] "naylampmechatronics." Accessed: Jul. 02, 2025. [Online]. Available: https://naylampmechatronics.com/blog/45_tutorial-mpu6050-acelerometro-y-giroscopio.html

[12] T. Wu *et al.*, "Research Status and Development Trend of Piezoelectric Accelerometer," *Crystals (Basel)*, vol. 13, no. 9, p. 1363, Sep. 2023, doi: 10.3390/cryst13091363.

[13] Anders O. Persson, "The Coriolis Effect: Four centuries of conflict between common sense and mathematics, Part I: A history to 1885", Accessed: Jul. 02, 2025. [Online]. Available: https://www.journal.meteohistory.org/index.php/hom/article/view/30/30

[14] “Coriolis Effect.” Accessed: Jul. 02, 2025. [Online]. Available: https://www.journal.meteohistory.org/index.php/hom/article/view/30/30

[15] A. Sucerquia, J. López, and J. Vargas-Bonilla, “SisFall: A Fall and Movement Dataset,” *Sensors*, vol. 17, no. 1, p. 198, Jan. 2017, doi: 10.3390/s17010198.

[16] M. D. Fiuza Pérez and J. C. Rodríguez Pérez, “La regresión logística: una herramienta versátil,” *Nefrología*, vol. 20, no. 6, pp. 495–500, 2000, [Online]. Available: https://www.revistanefrologia.com/es-la-regresion-logistica-una-herramienta-articulo-X0211699500035664

[17] N. Sánchez Anzola, “Máquinas de soporte vectorial y redes neuronales artificiales en la predicción del movimiento USD/COP spot intradiario,” *ODEON*, no. 9, p. 113, Feb. 2016, doi: 10.18601/17941113.n9.04.

[18] Luis Norberto Zúñiga Morales, “Ensemble Learning,” Feb. 2024, Accessed: Jul. 02, 2025. [Online]. Available: https://lzun.github.io/downloads/id/ensemble_learning_notas.pdf

[19] S. Devalal and A. Karthikeyan, “LoRa Technology - An Overview,” in *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, IEEE, Mar. 2018, pp. 284–290. doi: 10.1109/ICECA.2018.8474715.

[20] A. Zourmand, A. L. Kun Hing, C. Wai Hung, and M. AbdulRehman, “Internet of Things (IoT) using LoRa technology,” in *2019 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS)*, IEEE, Jun. 2019, pp. 324–330. doi: 10.1109/I2CACIS.2019.8825008.

[21] M. Rizzi, P. Ferrari, A. Flammini, E. Sisinni, and M. Gidlund, “Using LoRa for industrial wireless networks,” in *2017 IEEE 13th International Workshop on Factory Communication Systems (WFCS)*, IEEE, May 2017, pp. 1–4. doi: 10.1109/WFCS.2017.7991972.

[22] J. Haxhibeqiri, E. De Poorter, I. Moerman, and J. Hoebeke, “A Survey of LoRaWAN for IoT: From Technology to Application,” *Sensors*, vol. 18, no. 11, p. 3995, Nov. 2018, doi: 10.3390/s18113995.

[23] N. Ashby, “Relativity in the Global Positioning System,” *Living Rev Relativ*, vol. 6, no. 1, p. 1, Dec. 2003, doi: 10.12942/lrr-2003-1.

[24] Ahmed El-Rabbany, “Introduction to GPS, The Global Positioning System,” 2002.

[25] ÁNGEL LUIS SÁNCHEZ LÁZARO, “Apuntes de Seguridad en sistemas Informáticos,” *Algoritmos de Cifrado*.

[26] T. V. Ramabadran and S. S. Gaitonde, “A tutorial on CRC computations,” *IEEE Micro*, vol. 8, no. 4, pp. 62–75, Aug. 1988, doi: 10.1109/40.7773.

[27] “postgresql.” Accessed: Jul. 02, 2025. [Online]. Available: https://www.postgresql.org/

[28] “mariadb”, Accessed: Jul. 02, 2025. [Online]. Available: https://mariadb.org/

[29] “mongodb.” Accessed: Jul. 02, 2025. [Online]. Available: https://www.mongodb.com/

[30] “firebase.” Accessed: Jul. 02, 2025. [Online]. Available: https://firebase.google.com/?hl=es-419

[31] Joshen Lim, “Supabase,” https://github.com/supabase/supabase. Accessed: Jul. 02, 2025. [Online]. Available: https://github.com/supabase/supabase

[32] A. Z. (2022) Ayezabu, “Supabase vs Firebase : Evaluation of performance and development of Progressive Web Apps”, Accessed: Jul. 02, 2025. [Online]. Available: https://www.theseus.fi/handle/10024/771009

[33] “Supabase Logo.” Accessed: Jul. 02, 2025. [Online]. Available: https://supabase.com/brand-assets

[34] Swagger, “About Swagger Specification | Documentation,” Swagger by SmartBear. Accessed: Jul. 02, 2025. [Online]. Available: https://swagger.io/

[35] “Fast Api Logo.” Accessed: Jul. 02, 2025. [Online]. Available: https://fastapi.tiangolo.com/

[36] “GitHub Logo.” Accessed: Jul. 02, 2025. [Online]. Available: https://github.com/logos

[37] “Git Logo.” Accessed: Jul. 02, 2025. [Online]. Available: https://git-scm.com/downloads/logos

[38] “angular”, Accessed: Jul. 02, 2025. [Online]. Available: https://angular.dev/

[39] “vuejs”, Accessed: Jul. 02, 2025. [Online]. Available: https://vuejs.org/

[40] “react”, Accessed: Jul. 02, 2025. [Online]. Available: https://es.react.dev/

[41] “React Logo”, Accessed: Jul. 02, 2025. [Online]. Available: https://es.react.dev/images/brand/logo_dark.svg

[42] “Arduino Logo.” Accessed: Jun. 15, 2025. [Online]. Available: https://www.arduino.cc/en/trademark/community-logo/

[43] “fritzing.” Accessed: Jul. 02, 2025. [Online]. Available: https://fritzing.org/

[44] “jlcpcb.” Accessed: Jul. 02, 2025. [Online]. Available: https://jlcpcb.com/

[45] “fablab usal”, Accessed: Jul. 02, 2025. [Online]. Available: https://fablab.usal.es/

[46] “tinkercad”, Accessed: Jul. 02, 2025. [Online]. Available: https://www.tinkercad.com/

[47] J. Nordby, M. Cooke, and A. Horvath, “emlearn: Machine Learning inference engine for Microcontrollers and Embedded Devices ,” Apr. 2025, *Zenodo*. doi: 10.5281/zenodo.15225305.

[48] J. Calles Rivas, “CheetHost cloud computing.,” https://cheetahost.com/. Accessed: Jul. 02, 2025. [Online]. Available: https://cheetahost.com/

[49] sandeepmistry, “LoRa.endPacket().” Accessed: Jul. 02, 2025. [Online]. Available: https://github.com/sandeepmistry/arduino-LoRa/issues/363

[50] R. Santamaría, *Cómo se organizan las máquinas. Una introducción a los sistemas distribuidos*. Ediciones Universidad de Salamanca, 2024. doi: 10.14201/0BC0092.

[51] Carlos Ivorra Castillo, “CURVAS ELIPTICAS”, Accessed: Jul. 02, 2025. [Online]. Available: https://biblioteca.utec.edu.sv/siab/virtual/interactiva/980000007.pdf

[52] U. M. Maurer and S. Wolf, “The Diffie–Hellman Protocol,” *Des Codes Cryptogr*, vol. 19, no. 2/3, pp. 147–171, 2000, doi: https://doi.org/10.1023/A:1008302122286.