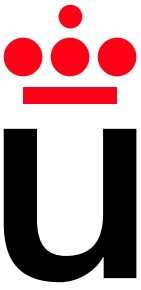

Universidad
Rey Juan Carlos

Escuela Técnica Superior
de Ingeniería Informática

Grado en Ingeniería del Software e Ingeniería Informática

Curso 2024-2025

Trabajo Fin de Grado

SISTEMA DE NAVEGACIÓN HÍBRIDA MÓVIL PARA INTERIORES Y EXTERIORES CON BLE Y GOOGLE MAPS

Autor: Álvaro Serrano Rodrigo
Tutores: Holger Billhardt e Iván Bernabé Sánchez

Agradecimientos

Quisiera expresar mi más sincero agradecimiento a Alberto Fernández Gil, Holger Billhardt e Iván Bernabé Sánchez por su apoyo e implicación en el desarrollo de este Trabajo de Fin de Grado. Su orientación desde la fase inicial fue clave para consolidar la idea del proyecto y definir su enfoque técnico.

En especial, agradezco profundamente a mis tutores Holger Billhardt e Iván Bernabé Sánchez por su constante acompañamiento, sugerencias acertadas y paciencia a lo largo de todo el proceso. Su experiencia y dedicación han sido fundamentales para guiarme en la evolución y refinamiento de esta propuesta, desde sus primeras fases hasta la implementación final.

Gracias a ellos, este proyecto ha sido una oportunidad de aprendizaje valiosa y enriquecedora.

Resumen

Este Trabajo de Fin de Grado desarrolla una aplicación móvil de navegación híbrida que combina posicionamiento exterior, basado en la API de Google Maps, con guiado interior mediante balizas BLE tipo Eddystone. La solución está diseñada para ofrecer una transición fluida entre ambos entornos, adaptándose de forma contextual al entorno del usuario sin necesidad de intervención manual constante.

Código fuente y documentación: Este proyecto está disponible públicamente en GitHub en `https://github.com/AlvaroS3rrano/GuidingApp/`.

Palabras clave:

- Aplicaciones móviles
- Navegación en interiores
- Localización híbrida
- Beacons BLE (Eddystone)
- Google Maps API
- Posicionamiento basado en RSSI
- Sistemas de guiado inteligentes
- Interfaz de usuario y experiencia (UX)

Índice de contenidos

Índice de tablas

Índice de figuras

1 Introducción

1.1. Contextualización del problema

La navegación mediante dispositivos móviles ha evolucionado significativamente en los últimos años, facilitando la localización y desplazamiento de los usuarios en entornos exteriores gracias a tecnologías como el GPS y servicios como Google Maps. Sin embargo, en espacios interiores —como centros comerciales, aeropuertos, hospitales o edificios educativos— el posicionamiento se vuelve impreciso o directamente inviable debido a la pérdida de señal satelital. Esta limitación afecta negativamente a la experiencia del usuario, especialmente cuando se requiere encontrar ubicaciones específicas en entornos complejos.

Para abordar este problema, surgen soluciones basadas en tecnologías alternativas como los *beacons* con Bluetooth Low Energy (BLE), que permiten estimar posiciones en interiores mediante la recepción de señales y algoritmos de proximidad.

En este contexto, el presente trabajo propone el desarrollo de una aplicación móvil híbrida capaz de ofrecer navegación tanto en exteriores como en interiores. La aplicación integrará dos tecnologías complementarias: por un lado, el sistema de posicionamiento global (GPS) y la API de Google Maps para la localización en exteriores; por otro, un sistema de posicionamiento interior basado en balizas, que permitirá guiar al usuario en espacios cerrados.

El objetivo principal de la aplicación es permitir una transición fluida entre ambos entornos sin necesidad de intervención manual, detectando automáticamente cuándo el usuario entra en una zona interior y adaptando el modo de

navegación en consecuencia. El sistema ofrecerá instrucciones visuales en tiempo real, rutas optimizadas y una interfaz intuitiva que se adaptará al nivel del edificio en el que se encuentre el usuario. Esta solución está especialmente pensada para mejorar la accesibilidad, orientación y experiencia de los usuarios en instalaciones de gran tamaño o con múltiples niveles.

1.2. Objetivos del trabajo

El objetivo general de este Trabajo de Fin de Grado es desarrollar una aplicación móvil que proporcione guiado de usuarios en entornos tanto exteriores como interiores, integrando tecnologías complementarias de localización. Para lograrlo, se establecen los siguientes objetivos específicos:

- Implementar navegación exterior utilizando la API de Google Maps.
- Diseñar un sistema de guiado interior basado en *beacons* BLE.
- Unificar ambos modos de navegación en una sola interfaz de usuario fluida e intuitiva.
- Validar el sistema mediante pruebas reales en diferentes entornos, evaluando su precisión y usabilidad.

1.3. Estructura del documento

El contenido de este Trabajo de Fin de Grado se organiza en siete capítulos que siguen una secuencia lógica desde la contextualización del problema hasta el diseño, desarrollo, validación y conclusiones del sistema propuesto. A continuación se detalla la estructura del documento:

- **Capítulo 1: Introducción.** Presenta el contexto del problema, los objetivos del trabajo y la motivación del sistema de navegación híbrida, destacando la necesidad de integrar localización interior y exterior.
- **Capítulo 2: Fundamentos Teóricos y Estado del Arte.** Analiza las tecnologías existentes de localización, tanto exteriores (GPS) como interiores (Wi-Fi, BLE, RFID, UWB), así como los principios técnicos (RSSI, trilateración, etc.) y ejemplos de aplicaciones actuales relevantes. También se describen los protocolos estándar y la arquitectura general de los sistemas de posicionamiento.

- **Capítulo 3: Diseño del Sistema Propuesto.** Define los requisitos funcionales y técnicos, la metodología de desarrollo adoptada, las tecnologías seleccionadas (React Native, Spring Boot, BLE), y se describe la arquitectura general cliente-servidor del sistema.
- **Capítulo 4: Implementación.** Detalla el proceso de desarrollo técnico, incluyendo la configuración del entorno Android, la detección y gestión de balizas BLE, la integración con Google Maps, la generación de rutas mediante algoritmos A*, y la estructura del backend y su modelo de datos.
- **Capítulo 5: Interfaz de Usuario y Flujo de Navegación.** Describe las pantallas principales de la aplicación, el comportamiento del sistema ante eventos de localización, el uso de la barra de búsqueda, la visualización por pisos y los flujos de transición entre navegación exterior e interior.
- **Capítulo 6: Pruebas y Validación.** Expone los escenarios de prueba, la metodología de evaluación, los resultados obtenidos en términos de precisión, detección de balizas y consumo energético, y un análisis crítico del rendimiento del sistema.
- **Capítulo 7: Conclusiones y Trabajos Futuros.** Resume los logros alcanzados, las limitaciones detectadas y plantea posibles líneas de mejora, incluyendo autenticación, aprendizaje automático y mejoras en accesibilidad o escalabilidad.
- **Apéndices.** Incluyen material complementario como la lógica interna del sistema de búsqueda semántica y capturas adicionales de la interfaz gráfica. Estos contenidos amplían la comprensión de aspectos técnicos descritos en los capítulos centrales del documento.

2

Fundamentos Teóricos y Estado del Arte

Este capítulo presenta una revisión de las principales tecnologías y sistemas utilizados en el posicionamiento de usuarios tanto en entornos exteriores como interiores. Se analizan las tecnologías existentes, el uso de balizas electrónicas y algunos ejemplos representativos de aplicaciones actuales.

2.1. Tecnologías de localización existentes

El posicionamiento de usuarios en aplicaciones móviles puede abordarse mediante diferentes tecnologías, cuya idoneidad depende del entorno en el que se encuentren. En espacios abiertos, la solución predominante es el **GPS (Global Positioning System)**, un sistema de navegación global basado en satélites. Cada uno transmite señales de radiofrecuencia que contienen su ubicación y una marca temporal precisa. Un receptor, como el de un teléfono móvil, puede determinar su posición geográfica mediante un proceso de trilateración, calculando el tiempo de llegada de las señales desde al menos cuatro satélites. Este método proporciona una precisión razonable —generalmente entre 3 y 10 metros— en condiciones ideales.

Sin embargo, el rendimiento del GPS se ve afectado por la necesidad de línea de vista directa con los satélites. En entornos urbanos densos, túneles, interiores o espacios subterráneos, la señal puede debilitarse o perderse completamente, volviendo inoperativo el sistema. Esto limita su eficacia precisamente en los escenarios donde una guía precisa resulta más necesaria.

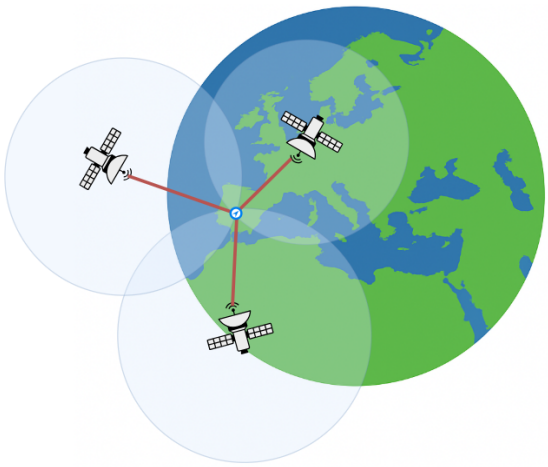

Figura 2.1: Esquema ilustrativo del concepto de trilateración utilizando señales GPS.

Para superar estas limitaciones, en espacios cerrados se recurre a tecnologías de localización alternativas que emplean infraestructura local. Estas soluciones permiten estimar la posición de un usuario a partir de señales generadas por dispositivos cercanos, utilizando distintos enfoques como *fingerprinting*, detección por proximidad o medición de tiempos de llegada (ToA).

Entre las principales tecnologías disponibles, se destacan las siguientes:

Wi-Fi Positioning System (WPS): Aprovecha la red Wi-Fi existente para estimar la ubicación mediante triangulación o comparación con una base de datos de huellas digitales de señal. No requiere hardware adicional, lo que facilita su implementación, aunque su precisión depende de la densidad y estabilidad de los puntos de acceso, situándose generalmente entre 5 y 15 metros.

RFID (Radio Frequency Identification): Utiliza etiquetas electrónicas pasivas o activas que son detectadas por lectores estratégicamente ubicados. Su bajo coste y simplicidad son ventajas importantes, aunque su alcance limitado y la necesidad de infraestructura específica restringen su escalabilidad en entornos amplios.

UWB (Ultra-Wideband): Emplea pulsos de radiofrecuencia de alta precisión para calcular distancias mediante ToA. Destaca por ofrecer precisiones del orden de 10 centímetros y una elevada resistencia a interferencias. No obstante, su despliegue implica costes considerables y su adopción aún es limitada en dispositivos móviles convencionales.

Bluetooth Low Energy (BLE): Consiste en balizas —conocidas como *beacons*— que transmiten identificadores únicos mediante señales BLE. Los dispositivos móviles pueden detectarlas y estimar la distancia mediante el análisis del valor de intensidad recibida (RSSI). Es una tecnología ampliamente soportada,

económica y de bajo consumo, con una precisión típica de entre 1 y 5 metros.

Aunque estas tecnologías permiten ofrecer localización funcional en interiores, también plantean desafíos técnicos relevantes. Entre ellos, destacan la atenuación y dispersión de señales dentro de edificios, el efecto multipath (rebote de señales en superficies), las variaciones de precisión según el entorno y los costes asociados a la instalación y mantenimiento de la infraestructura, especialmente en soluciones como RFID o UWB. Por tanto, la elección de una tecnología debe considerar no solo la precisión, sino también su viabilidad técnica y económica en el contexto específico de uso.

2.2. Ejemplos de aplicaciones actuales

El posicionamiento en interiores ha ganado protagonismo en aplicaciones comerciales, de transporte, culturales y de accesibilidad. A continuación se presentan algunas soluciones relevantes que integran tecnologías de localización mixtas como Wi-Fi, BLE, sensores inerciales y mapeo previo:

- **Google Indoor Maps**: Este servicio de Google extiende Google Maps al interior de edificios como aeropuertos, centros comerciales, estaciones de transporte y hospitales. Utiliza una combinación de señales Wi-Fi, sensores del dispositivo y mapas previamente trazados por los propietarios del recinto. El usuario puede ver plantas por niveles, obtener instrucciones paso a paso y ubicar tiendas o servicios dentro del recinto. **Tecnología:** Wi-Fi, datos crowdsourced, sensores internos. **Enlace:** Sitio oficial de Google Indoor Maps
- **Apple Indoor Positioning**: Apple ofrece una solución de posicionamiento en interiores basada en la tecnología iBeacon y mapeo personalizado. Los administradores de espacios pueden enviar sus planos a Apple y usar balizas BLE calibradas para facilitar una navegación precisa en recintos como centros comerciales, museos o aeropuertos. La tecnología está integrada en iOS desde iOS 8 en adelante. **Tecnología:** BLE (iBeacon), sensores del dispositivo. **Enlace:** Página de Apple Indoor Positioning
- **Aplicaciones de museos y exposiciones**: Diversos museos han implementado aplicaciones móviles que utilizan posicionamiento interior mediante balizas BLE para ofrecer contenido contextual e interactivo. Por ejemplo:
  - El *Louvre Lens* y el *Reina Sofía* permiten que los visitantes reciban información específica al acercarse a una obra de arte.
  - El *American Museum of Natural History* en Nueva York ofrece una app con navegación interior mediante Wi-Fi y balizas.

Estas apps combinan accesibilidad, educación y tecnología para enriquecer la experiencia del visitante. **Tecnología:** BLE, Wi-Fi, GPS (exteriores), sensores móviles. **Ejemplo:** AMNH Explorer App

Estos ejemplos demuestran cómo la localización en interiores se ha convertido en una herramienta transversal para facilitar la orientación, enriquecer la experiencia del usuario y ofrecer servicios adaptados al contexto. Su aplicación va desde la accesibilidad y el turismo hasta el marketing geolocalizado.

2.3. Principios y arquitectura de los sistemas de localización

Los sistemas de localización, tanto en exteriores como interiores, se basan en principios físicos como la trilateración, la proximidad o la medición de la intensidad de señal (RSSI).

La **trilateración** calcula la posición de un dispositivo mediante su distancia respecto a tres o más emisores de posición conocida. Es utilizada en sistemas GPS y BLE, estimando distancias según el tiempo de llegada o la pérdida de potencia de la señal [1](véase la figura Figure 2.1).

La **proximidad** detecta simplemente la cercanía de un dispositivo a una baliza. Es menos precisa pero útil en escenarios simples como activación de contenido o control de accesos.

La técnica **RSSI (Received Signal Strength Indicator)** estima la distancia a una baliza en función de la intensidad de la señal recibida. Es muy empleada en soluciones BLE por su bajo coste, aunque es sensible a interferencias, reflexiones y barreras físicas [2].

En cuanto a la arquitectura técnica, un sistema típico de localización con **balizas BLE** incluye:

- **Balizas:** Dispositivos emisores que transmiten paquetes de datos periódicamente (por ejemplo, UUIDs o URLs).
- **Dispositivo móvil:** Escanea señales BLE cercanas y calcula ubicación mediante los principios mencionados anteriormente.
- **Servidor backend (opcional):** Administra mapas interiores, perfiles de usuario y eventos contextuales.

Este tipo de arquitectura es eficiente, modular y fácilmente desplegable en interiores, como universidades, museos o estaciones de transporte [3].

2.4. Protocolos y estándares de localización

En la actualidad existen varias alternativas basadas en BLE (y una opción UWB) que facilitan la implementación de localización en interiores:

- **iBeacon (Apple):** Sistema propietario de Apple que emite identificadores UUID, mayor y menor para que una app detecte la proximidad a balizas. Ideal para activar contenido contextual o guiar al usuario al entrar en zonas delimitadas. Ofrece una integración nativa y optimizada en dispositivos iOS. **Más información:** Documentación oficial de iBeacon
- **Eddystone (Google):** Protocolo abierto de Google capaz de transmitir varios tipos de paquetes: `UID` (identificador), `URL` (para navegación directa) y `TLM` (telemetría de la baliza). Su diseño multiplataforma facilita su uso en Android e iOS sin ataduras de fabricante. **Más información:** Repositorio y especificación de Eddystone
- **AltBeacon (Radius Networks):** Especificación completamente libre y abierta, creada para evitar dependencias de proveedores. Permite personalizar el formato de los paquetes BLE y ajustar parámetros de señal para distintos casos de uso. **Más información:** Guía y especificaciones de AltBeacon
- **UWB (Ultra-Wideband):** Tecnología de radio de muy alta precisión basada en medición de tiempo de vuelo (ToF), capaz de localizar objetos con errores por debajo de 30 cm. Aunque no usa BLE, comienza a incorporarse en Smartphones modernos (p. ej. Apple U1, algunos Android) para casos donde se requiere máxima exactitud [4]. **Más información general:** estándares IEEE 802.15.4a y 802.15.4z para UWB

Cada protocolo presenta un compromiso distinto entre precisión, cobertura, consumo energético y facilidad de integración. Para soluciones generales de bajo coste y amplio soporte de dispositivos, BLE (iBeacon, Eddystone o AltBeacon) constituye la opción más equilibrada. Si la aplicación exige precisión centimétrica, UWB es la alternativa recomendada.

3

Diseño del Sistema Propuesto

Este capítulo describe el diseño técnico del sistema de guiado desarrollado. Se detallan los requisitos, la arquitectura general del sistema, las tecnologías seleccionadas y el modelo de despliegue.

3.1. Requisitos del sistema

El sistema propuesto debe satisfacer una serie de requisitos funcionales y técnicos orientados a garantizar una experiencia de navegación fluida, precisa y energéticamente eficiente tanto en exteriores como en interiores. A continuación se enumeran los principales:

- Proporcionar guiado de usuarios en entornos exteriores mediante la integración con las APIs existentes de Google Maps, incluyendo posicionamiento GPS y generación de rutas.
- Implementar un sistema de localización interior basado en balizas BLE (Bluetooth Low Energy), que permita estimar la posición del usuario en tiempo real dentro de edificios [2, 3].
- Disponer de una base de datos de mapas interiores accesible desde la aplicación cliente, incluyendo información estructural del edificio y nodos relevantes para el guiado.

- Detectar automáticamente la transición entre navegación exterior e interior mediante la presencia o ausencia de balizas BLE, y conmutar entre ambos modos sin requerir intervención manual del usuario [4].
- Minimizar el consumo energético en dispositivos móviles, optimizando el uso de Bluetooth, sensores y geolocalización, al tiempo que se mantiene una experiencia de usuario fluida e intuitiva.

3.2. Metodología empleada

El desarrollo del sistema se ha llevado a cabo siguiendo una metodología iterativa e incremental. Esta estrategia permitió avanzar en ciclos sucesivos de análisis, diseño, implementación, pruebas y mejora continua, incorporando retroalimentación a lo largo del proceso.

El desarrollo se centró exclusivamente en la plataforma **Android**, lo que permitió aprovechar características específicas del sistema operativo como el acceso flexible a los servicios de localización y al hardware Bluetooth Low Energy (BLE).

Se utilizó el framework **React Native** junto con la plataforma **Expo**, lo que proporcionó un entorno moderno y eficiente para desarrollar aplicaciones móviles con JavaScript, facilitando la integración de bibliotecas y herramientas.

La API de **Google Maps** se empleó para gestionar la navegación en exteriores mediante GPS, mientras que la detección de balizas BLE para posicionamiento en interiores se implementó usando la librería `react-native-ble-plx`, permitiendo escanear dispositivos cercanos y estimar la distancia por intensidad de señal (RSSI), tal como se detalla en [3].

Todas las pruebas del sistema se realizaron en dispositivos físicos reales con Android, en distintos escenarios tanto interiores como exteriores. Se evaluaron aspectos clave como la precisión del posicionamiento, la estabilidad del escaneo BLE, el rendimiento de la navegación y la experiencia de usuario.

3.3. Tecnologías empleadas en el sistema

El sistema desarrollado combina tecnologías modernas de desarrollo móvil, localización inalámbrica y servicios backend para ofrecer una solución de guiado interior y exterior integrada.

La aplicación cliente fue desarrollada utilizando **React Native** junto con la plataforma **Expo**, lo cual permitió un desarrollo multiplataforma enfocado principalmente en dispositivos Android. Para la estructura de la interfaz y na-

vegación entre pantallas se emplearon componentes como `expo-router`, `Slot`, `GestureHandlerRootView` y `StyleSheet`, que ofrecen control sobre estilos y gestos táctiles.

En cuanto a la localización, el cliente móvil integra dos subsistemas complementarios. Para entornos exteriores se utilizó la API oficial de **Google Maps**, accedida mediante bibliotecas como `react-native-maps` y `MapViewRoute` para la visualización de mapas, trazado de rutas y colocación de marcadores. La ubicación GPS del usuario se obtiene con `@react-native-community/geolocation`, lo que permite guiar al usuario desde el exterior hasta un punto de acceso interior, haciendo uso de técnicas de rutas mínimas como las propuestas en [5].

Para la navegación en interiores, el sistema se basa en el uso de balizas **Bluetooth Low Energy (BLE)** configuradas bajo el protocolo **Eddystone-UID**, que emite un identificador único por dispositivo. La app móvil detecta estas balizas mediante la biblioteca `react-native-ble-plx`, y estima la distancia según el valor RSSI recibido. El procesamiento y codificación de los datos se realiza con `react-native-base64`, y la lógica de posicionamiento se basa en modelos de atenuación calibrados al entorno físico, como los revisados en [2, 3].

Por último, el backend fue implementado usando **Spring Boot** en **Java**, configurando una API REST que se comunica con la aplicación móvil para proporcionar datos estructurales del edificio (nodos, enlaces, puntos de interés) y rutas óptimas. Estos datos se almacenan en una base de datos **MySQL**, que modela la estructura del edificio en forma de grafo, junto con las posiciones registradas de las balizas BLE. Esta arquitectura permite desacoplar la lógica de posicionamiento del cliente y mantener el sistema escalable para diferentes edificios o mapas.

En conjunto, esta infraestructura tecnológica permite una experiencia de navegación continua entre entornos exteriores e interiores, con transición automática basada en la detección de balizas y el contexto de ubicación del usuario.

3.4. Arquitectura del sistema

La arquitectura del sistema sigue un modelo **cliente-servidor modular**, como se muestra en la Figura 3.1. El cliente está representado por la aplicación móvil instalada en el teléfono del usuario, que se encarga de escanear balizas BLE para obtener posicionamiento en interiores, acceder al GPS para ubicarse en exteriores, visualizar mapas, generar rutas interiores y comunicarse con el servidor a través de API REST.

El servidor se divide en dos partes: por un lado, los servidores de Google, que proporcionan funcionalidades externas mediante APIs como los servicios de

mapas y geolocalización; por otro, un servidor propio que gestiona la lógica de la aplicación, atiende las peticiones REST y accede a una base de datos MySQL, donde se almacena la información estructural del edificio, entre otra información 4.5. La interacción entre estos componentes se detalla en el diagrama de secuencia incluido en el Anexo A.

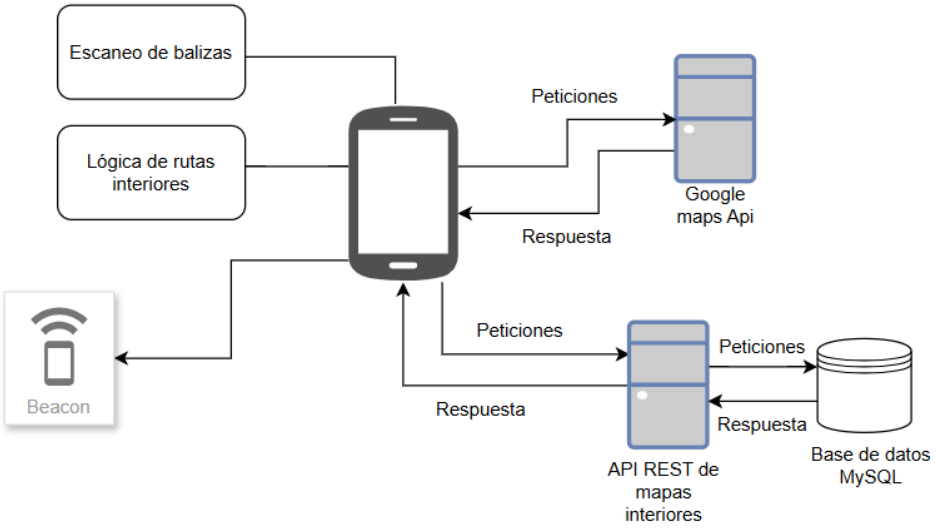

Figura 3.1: Arquitectura general del sistema de guiado interior y exterior.

3.5. Despliegue e integración

El sistema se implementa en dos contextos:

- **Entorno exterior**: no requiere infraestructura adicional. El dispositivo usa GPS y las APIs de Google Maps para posicionar al usuario y guiarlo hasta la entrada del edificio.
- **Entorno interior**: se instalan balizas BLE **Eddystone** en ubicaciones clave (entradas, pasillos, escaleras). Estas emiten identificadores que la app detecta automáticamente. En función de los RSSI recibidos, se estima la ubicación y se consulta el backend para trazar rutas internas.

El sistema conmuta de forma automática entre navegación exterior e interior según la detección o pérdida de señal de balizas registradas, garantizando una transición fluida y sin necesidad de intervención por parte del usuario.

4

Implementación

Este capítulo detalla los aspectos técnicos clave del sistema desarrollado. Se abordan desde la configuración del entorno Android y la integración de bibliotecas externas, hasta la lógica de escaneo BLE, cálculo de rutas interiores, comunicación con el backend y estructura de los datos. El objetivo es presentar de forma estructurada las decisiones tecnológicas adoptadas y su implementación.

4.1. Configuración de permisos y metadatos en Android

Para habilitar las funcionalidades esenciales del sistema, se configuraron diversos permisos y metadatos en el archivo `AndroidManifest.xml` de la aplicación. Esta configuración garantiza el acceso a sensores, servicios de red, Bluetooth y mapas, necesarios para el posicionamiento híbrido interior-exterior.

Permisos declarados

- **Localización**: `ACCESS_FINE_LOCATION`, `ACCESS_COARSE_LOCATION` y `ACCESS_BACKGROUND_LOCATI` para permitir el escaneo de balizas BLE en primer y segundo plano, y la obtención precisa de la ubicación mediante GPS.

- **Bluetooth y escaneo BLE**: `BLUETOOTH`, `BLUETOOTH_ADMIN`, `BLUETOOTH_CONNECT`,

`BLUETOOTH_SCAN`, necesarios para interactuar con dispositivos BLE, detectar balizas y gestionar conexiones inalámbricas.

- **Red e Internet**: `INTERNET` y `ACCESS_NETWORK_STATE`, para la comunicación con el servidor backend y la carga dinámica de mapas.
- **Servicios en primer plano**: `FOREGROUND_SERVICE` y `FOREGROUND_SERVICE_LOCATION`, permiten mantener activo el servicio de localización incluso cuando la aplicación se ejecuta en segundo plano.
- **Otros permisos auxiliares**: `RECEIVE_BOOT_COMPLETED`, `VIBRATE`, `SYSTEM_ALERT_WINDOW`, `READ_EXTERNAL_STORAGE` y `WRITE_EXTERNAL_STORAGE`, algunos de los cuales pueden estar reservados para funciones futuras como notificaciones contextuales o gestión de recursos multimedia.

Metadatos relevantes

- `com.google.android.geo.API_KEY`: se especifica la clave de API para permitir la integración con Google Maps en navegación exterior.
- `expo.modules.updates.*`: parámetros propios de Expo para la gestión de actualizaciones OTA (*over the air*).

4.2. Gestión de balizas BLE y flujo de transición de modo

El comportamiento del sistema ante la detección de balizas BLE es gestionado principalmente por el módulo `beaconScannerService.ts`, el cual se encarga del escaneo, clasificación y actualización dinámica de dispositivos Bluetooth de baja energía.

Este proceso no actúa de forma aislada, sino que está estrechamente integrado con los servicios de obtención de datos desde el backend, como `mapDataService.ts` y `nodeService.ts`, permitiendo actualizar en tiempo real la visualización del entorno y la lógica de navegación interior.

Escaneo y clasificación de balizas

Al iniciar la aplicación, se solicita al usuario la concesión de los permisos necesarios para el funcionamiento del sistema, incluyendo Bluetooth, localización y escaneo en segundo plano. Una vez habilitados, comienza un escaneo continuo de dispositivos BLE cercanos mediante la librería `react-native-ble-plx`. Cada

baliza detectada es analizada a partir de su identificador único (`UUID`) y comparada contra el conjunto de balizas previamente registradas en el backend. Si la baliza coincide con una entrada conocida, se clasifica como *reconocida* y se añade a la lista de balizas conocidas, quedando vinculada a un nodo del grafo de navegación interior. En cambio, si la baliza no forma parte del conjunto registrado, se clasifica como *desconocida* y se almacena en una lista separada sin afectar al comportamiento del sistema.

La aplicación mantiene de forma dinámica la lista de balizas conocidas, descartando automáticamente aquellas cuya señal no se recibe durante más de cinco segundos. Esta lógica de depuración reduce la probabilidad de mostrar posiciones erróneas (conocidas como "posiciones fantasma") y mejora significativamente la fiabilidad general del guiado interior.

Flujo de transición y carga de mapa interior

Cada vez que se identifica una baliza conocida, se activa el siguiente flujo:

1. Se identifica la **baliza activa más cercana**, es decir, aquella con mayor valor de `RSSI` dentro del conjunto de balizas conocidas detectadas durante el escaneo BLE.

2. Si esta baliza está asociada a un edificio distinto del actualmente cargado, se invoca el servicio `mapDataService.ts` para solicitar al backend el `MapDataDTO` correspondiente a ese nuevo entorno.

3. Una vez descargado el objeto `MapDataDTO`, se localiza el **nodo vinculado a la baliza**, es decir, el nodo cuyo campo `beaconId` coincide con el identificador de la señal detectada.

4. Finalmente, el sistema actualiza el estado interno asignando este nuevo objeto como `currentMapData`, y establece el nodo correspondiente a la baliza como `currentBeacon`, lo que permite adaptar dinámicamente la visualización del mapa y el cálculo de rutas interiores.

La Figura 4.1 resume este flujo técnico desde la detección inicial hasta la transición final.

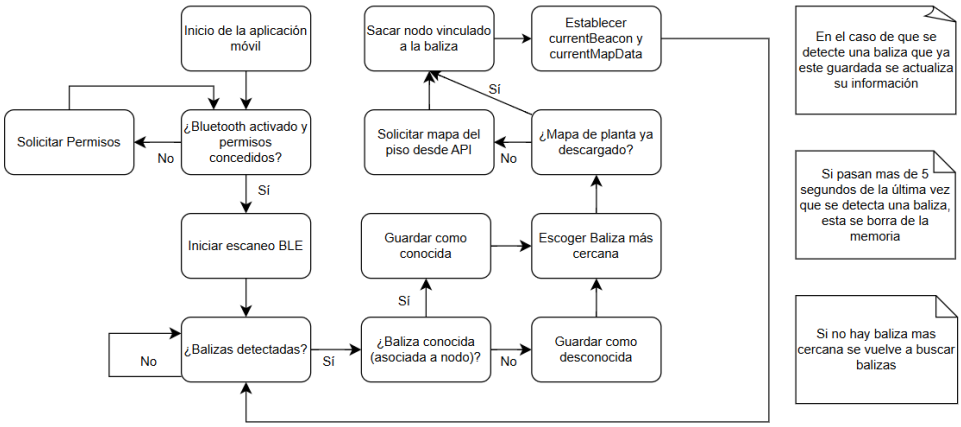

Figura 4.1: Flujo técnico desde la detección de balizas hasta la activación del guiado interior.

4.3. Integración con servicios de Google Maps

Para la navegación exterior, se integraron herramientas del ecosistema **Google Maps**, como:

- `react-native-maps` y `MapViewRoute`: empleados para renderizar mapas interactivos y superponer rutas en la interfaz.
  Documentación oficial: github.com/react-native-maps/react-native-maps
- `@react-native-community/geolocation`: módulo encargado de acceder a los servicios de geolocalización del dispositivo para obtener la posición actual del usuario.
  Documentación oficial: github.com/react-native-geolocation/react-native-geolocation
- `react-native-maps-routes`: permite calcular rutas entre coordenadas GPS y visualizarlas sobre el mapa.
  Documentación oficial: npmjs.com/package/react-native-maps-routes

El uso de estos servicios requiere registrar una **clave de API** que identifica al proyecto en Google Cloud. Esta clave es obligatoria para autenticar peticiones, aplicar restricciones de uso y controlar el consumo de recursos. Google ofrece un crédito gratuito mensual de **200 USD**, tras el cual se aplica una tarifa por uso. Más información en: **Google Maps Platform Pricing**.

Alternativas: para el desarrollo de aplicaciones de guiado tanto en exteriores como en interiores, existen diversas plataformas que ofrecen servicios cartográficos y de localización. Una opción destacada es **Mapbox**[1], una solución comercial que permite crear mapas altamente personalizables con soporte para navegación, posicionamiento y visualización 3D, lo cual es especialmente útil en entornos complejos como aeropuertos o centros comerciales. Otra alternativa es **OpenStreetMap**[2], una plataforma de cartografía colaborativa y abierta, ideal para proyectos donde se requiere independencia de licencias comerciales. Combinada con bibliotecas como Leaflet u OpenLayers, permite una integración sencilla y efectiva en aplicaciones web o móviles, aunque el soporte para interiores requiere extensiones adicionales. Finalmente, **Here Maps**[3] ofrece un conjunto robusto de APIs para localización en tiempo real, navegación peatonal y cobertura en interiores, incluyendo mapeo de plantas y posicionamiento basado en Wi-Fi o Bluetooth, lo cual la convierte en una alternativa potente para aplicaciones de guiado híbrido.

4.4. Generación de rutas

El sistema de guiado se divide en dos subsistemas de navegación: rutas en exteriores, basadas en servicios de Google Maps, y rutas en interiores, generadas localmente mediante algoritmos sobre grafos.

Rutas en exteriores

Para el trazado de rutas entre coordenadas GPS en espacios abiertos, se emplea la biblioteca `react-native-maps-routes`, que extiende `react-native-maps`, permitiendo calcular y visualizar rutas directamente sobre el mapa proporcionado por la API de Google Maps. Esta herramienta gestiona automáticamente aspectos como curvas, direcciones y cambios de nivel de zoom, mejorando la experiencia de navegación en exteriores.

Rutas en interiores

El guiado en interiores se basa en un modelo de **dígrafo ponderado**, en el cual los **nodos** representan ubicaciones clave como puertas, escaleras o cruces de pasillos, y las **aristas dirigidas** corresponden a los caminos que conectan estos nodos, estando ponderadas según la distancia real en metros entre ellos.

[1] `https://www.mapbox.com/`
[2] `https://www.openstreetmap.org/`
[3] `https://www.here.com/`

Los datos del grafo (nodos, enlaces, pesos) se reciben desde el backend y son procesados por los módulos `mapDataService.ts` y `nodeService.ts`. El cálculo de rutas se realiza en dos fases, ambas implementadas con variantes del algoritmo **A*** [5]:

1. **Selección del camino óptimo entre nodos:** se utiliza un A* clásico para determinar la secuencia ideal de nodos desde el origen hasta el destino. Este módulo está implementado en `findPathAStar.ts`. La efectividad de este enfoque ha sido demostrada en aplicaciones de navegación multiescenario y multiedificio [6, 7].

2. **Interpolación y trazado gráfico:** una vez seleccionados los nodos clave, se aplica un segundo A* sobre una matriz de celdas que representa el mapa interno del edificio. Este algoritmo (en `findShortestPathInMatrix.ts`) permite generar la línea visual realista sobre el plano, adaptándose a la geometría del entorno. Este tipo de modelado y representación ha sido ampliamente utilizado en entornos robóticos y móviles para mejorar la visualización del trayecto en tiempo real [8, 9, 10].

Esta arquitectura permite una navegación precisa en entornos interiores complejos, con rutas que evitan obstáculos y siguen los trazados reales del edificio.

4.5. Modelo de datos y estructura del backend

El sistema implementa una arquitectura de tipo **cliente-servidor modular**, ya descrita en la sección 3.4. En ella, la lógica de presentación, cálculo de rutas y escaneo BLE reside completamente en el cliente móvil, mientras que el backend expone una **API REST** que sirve como punto de acceso a los datos estructurales del entorno interior.

La API está desarrollada en **Spring Boot** y ofrece datos en formato `JSON` sobre `HTTPS`. Estos datos incluyen la definición completa del grafo de navegación, las balizas BLE asociadas a cada nodo, y las matrices de celdas que representan cada planta del edificio.

Toda la información estructurada se transmite mediante objetos DTO (*Data Transfer Objects*), que encapsulan de forma eficiente los distintos elementos necesarios para el guiado interior. La Figura 4.2 presenta un esquema entidad-relación simplificado:

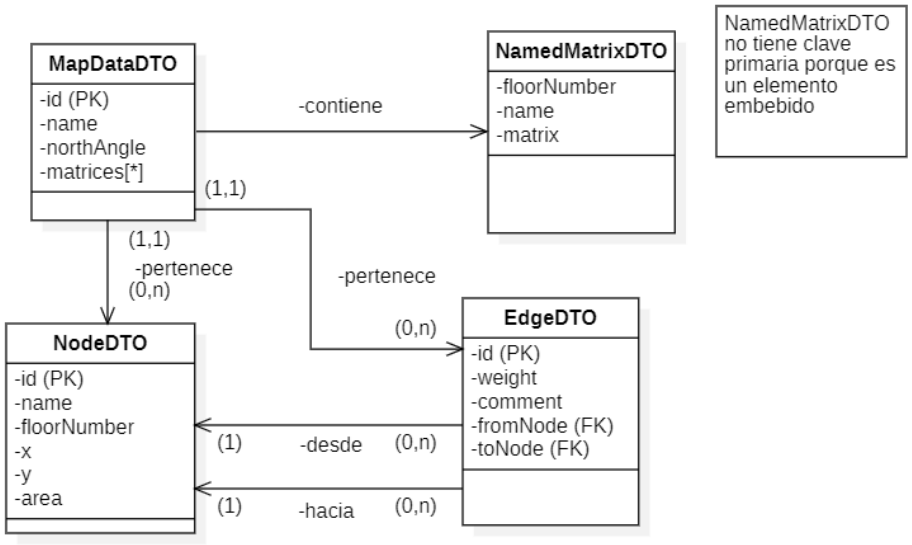

Figura 4.2: Diagrama entidad-relación del modelo de datos del sistema de guiado interior.

- **MapDataDTO**: contenedor principal que agrupa nodos, conexiones (edges) y mapas de planta (matrices binarizadas).
- **NodeDTO**: define cada punto de interés del grafo. Incluye identificador, coordenadas, planta, y atributos como `entry`, `exit` o `beaconId`.
- **EdgeDTO**: describe conexiones entre nodos, indicando origen, destino y distancia (peso).
- **NamedMatrixDTO**: matriz binaria (`0/1`) asociada a cada planta, utilizada para el trazado gráfico preciso mediante A*.

4.6. Servicios del cliente y gestión de actualizaciones

En el cliente, la descarga de datos desde el backend se gestiona mediante dos servicios principales. El primero, `mapDataService.ts`, se encarga de solicitar y almacenar en caché el objeto completo `MapDataDTO` correspondiente al edificio actual. Su ejecución se activa automáticamente al detectar una baliza asociada a un nuevo entorno. El segundo, `nodeService.ts`, permite acceder a información detallada sobre nodos individuales y ofrece un método específico para realizar consultas semánticas, las cuales son empleadas por el componente de búsqueda

unificado descrito en la sección 5.3. La lógica detallada de estas consultas se encuentra en el Apéndice B.

Todos los datos se transmiten en formato `JSON` a través de conexiones `HTTPS`, cumpliendo así con los requisitos de seguridad descritos en la sección 4.7.

Persistencia y actualizaciones

Actualmente, el sistema funciona en modo **solo lectura**: los datos de mapas y nodos son precargados en el servidor y no pueden ser modificados desde la app. Esta decisión simplifica la implementación y evita problemas relacionados con sincronización o edición en tiempo real. No obstante, se deja abierta la posibilidad de habilitar escritura o actualización dinámica en futuras versiones, por ejemplo para registrar rutas recorridas, personalizar puntos de interés o realizar auditoría de uso.

4.7. Seguridad y privacidad del sistema

El diseño del sistema prioriza la protección de la información personal y operativa del usuario, anticipando y mitigando posibles riesgos. Entre los principales peligros se encuentran: **Filtración de ubicación**, al evitar transmitir o almacenar la posición GPS del usuario; **Acceso no autorizado**, prevenido mediante el uso exclusivo de conexiones seguras `HTTPS`; **Exposición de datos personales**, mitigado al no exigir autenticación ni almacenar registros de uso; **Uso indebido de permisos**, resuelto solicitando únicamente los permisos estrictamente necesarios para la funcionalidad (como localización y Bluetooth); y **Riesgos derivados de bibliotecas externas**, controlado evitando dependencias con rastreo o publicidad embebida. Todo el enfoque adoptado sigue los lineamientos del **Reglamento General de Protección de Datos (GDPR)** [11], promoviendo así un tratamiento responsable y transparente de cualquier dato potencialmente sensible.

Configuración de comunicaciones seguras en desarrollo

Dado que el sistema exige que toda la comunicación entre la aplicación móvil y el backend se realice a través de **conexiones HTTPS** para prevenir accesos no autorizados y proteger los datos transmitidos, fue necesario abordar ciertos retos específicos durante el entorno de desarrollo.

En particular, Android impone restricciones estrictas sobre el uso de HTTP sin cifrado, y React Native no permite desactivar la validación TLS de manera

trivial. Además, los certificados autofirmados comúnmente empleados durante el desarrollo local no son considerados confiables por dispositivos Android. Estos factores impiden realizar pruebas locales con tráfico no cifrado o con certificados inválidos.

Para resolver estas limitaciones, se utilizó **Ngrok**[4], una herramienta que crea túneles seguros entre la máquina de desarrollo (donde se ejecuta el backend *Spring Boot*) y un dominio público con HTTPS válido. La app móvil puede conectarse a ese dominio sin conflicto, permitiendo probar funcionalidades como navegación, posicionamiento y solicitud de rutas bajo condiciones realistas.

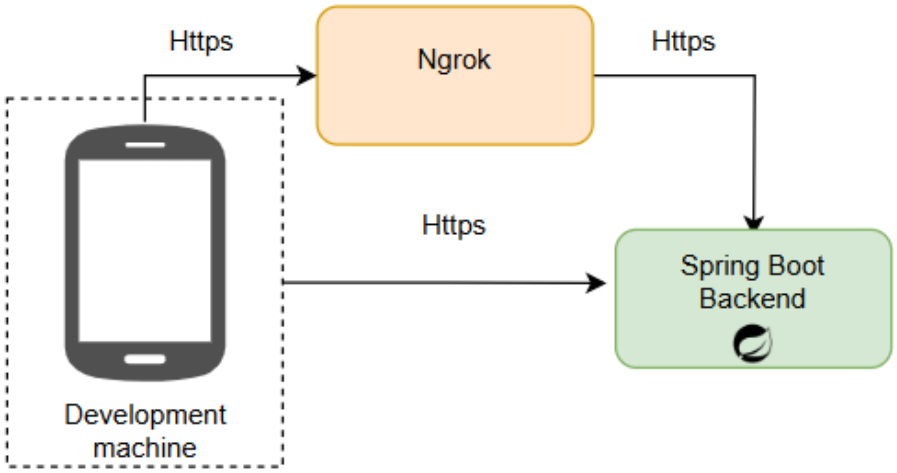

Figura 4.3: Túnel HTTPS mediante Ngrok

Gracias a esta configuración, se logró mantener un entorno de pruebas seguro, compatible con las políticas de Android y representativo de la arquitectura final del sistema en producción.

[4] `https://ngrok.com/`

4.8. Resumen de tecnologías utilizadas

Tabla 4.1: Resumen de tecnologías utilizadas en el sistema

Frontend – Aplicación móvil	
Plataforma	React Native + Expo
Escaneo de balizas BLE	`react-native-ble-plx`: escaneo, detección y clasificación de dispositivos BLE
Mapas exteriores	`react-native-maps`, `MapViewRoute`: visualización sobre Google Maps
Geolocalización GPS	`@react-native-community/geolocation`: adquisición de coordenadas GPS
Generación de rutas exteriores	`react-native-maps-routes`: cálculo de trayectorias entre puntos GPS
Visualización de rutas interiores	Lógica interna personalizada: interpolación sobre matriz de celdas
Comunicación con backend	HTTPS: peticiones REST seguras
Servicios de datos	`mapDataService.ts`, `nodeService.ts`: descarga de mapas y metadatos
Backend – Servidor y lógica de datos	
Framework de backend	Spring Boot (Java): servidor API REST
Base de datos	MySQL: almacenamiento de nodos, aristas y matrices
Interfaz de comunicación	API REST (solo lectura, formato JSON)
Gestión estructurada de datos	DTOs: `MapDataDTO`, `NodeDTO`, `EdgeDTO`, `NamedMatrixDTO`
Serialización de matrices	Conversor de `int[][]` a texto plano en JSON
Lógica y procesamiento	
Algoritmo de rutas entre nodos	A* clásico (`findPathAStar.ts`)
Trazado gráfico sobre el mapa	A* sobre matriz de celdas (`findShortestPathInMatrix.ts`)
Clasificación y gestión de balizas	Lógica en `beaconScannerService.ts`
Entorno de desarrollo y seguridad	
Entorno seguro local	Ngrok: túnel HTTPS con certificado válido
Configuración Android	`AndroidManifest.xml`, `network_security_config.xml`
Claves API	`com.google.android.geo.API_KEY` para Google Maps
Cumplimiento normativo	Reglamento General de Protección de Datos (GDPR)

5

Interfaz de Usuario y Flujo de Navegación

Este capítulo muestra las principales pantallas de la aplicación, ilustrando el flujo de navegación que sigue el usuario desde la vista inicial hasta el guiado en interiores. También se describen las funcionalidades asociadas a cada componente visual y se justifica su relevancia dentro del sistema.

5.1. Inicio de la aplicación y detección contextual

Al abrir la aplicación, el sistema pone en marcha un proceso de inicialización integral que abarca múltiples componentes críticos para el funcionamiento contextual del guiado. Este proceso incluye la lectura y calibración de los sensores del dispositivo (como el acelerómetro, giroscopio y magnetómetro), la activación de los servicios de localización global (GPS, redes móviles y Wi-Fi), así como la búsqueda activa de balizas BLE (Bluetooth Low Energy) previamente registradas en la base de datos del sistema. La finalidad de este mecanismo es permitir una detección inmediata del entorno físico del usuario y adaptar el modo de navegación más adecuado en función de dicho contexto, mejorando tanto la precisión del guiado como la experiencia de uso.

En condiciones normales, es decir, cuando no se detecta ninguna baliza BLE en

las proximidades del usuario durante la fase de arranque, la aplicación asume que este se encuentra en un entorno exterior. En consecuencia, se inicia automáticamente en la vista de navegación exterior basada en Google Maps (ver sección 5.2). Esta vista está específicamente optimizada para recorridos al aire libre, ofreciendo funcionalidades como visualización dinámica del mapa, interacción mediante gestos multitáctiles (desplazamiento, ampliación y rotación), y actualización en tiempo real de la posición del usuario sobre el entorno cartográfico global. Esta configuración permite una exploración libre del entorno circundante.

No obstante, si durante el proceso de inicialización se detecta automáticamente la presencia de una o más balizas BLE válidas previamente registradas, la aplicación interpreta de forma inmediata que el usuario se encuentra en el interior de un edificio cubierto por el sistema de guiado inteligente. En ese caso, se realiza una transición directa y automática hacia la vista de navegación interior, sin necesidad de interacción adicional o confirmación manual por parte del usuario (ver sección 5.5). Esta lógica responde a un principio de eficiencia y contextualización contextual: se asume que la detección de una baliza específica en el primer momento implica que el entorno físico del usuario corresponde a un espacio cerrado, justificado para activar directamente el modo de guiado interior.

Este mecanismo inteligente evita pasos intermedios innecesarios, como selecciones de modo o pantallas de carga redundantes, y permite ofrecer una experiencia más fluida y ágil desde el primer instante de uso. Además, si el sistema reconoce que la baliza corresponde a un edificio de múltiples niveles, se procede automáticamente a cargar el plano del piso correspondiente a la baliza detectada, posicionando al usuario en su localización inicial estimada dentro del edificio. Este posicionamiento se representa visualmente en la interfaz de navegación interior, facilitando la orientación inmediata y la planificación de rutas en entornos complejos como hospitales, centros comerciales o campus universitarios.

5.2. Pantalla principal: mapa global

En aquellos casos en los que no se detecta una baliza BLE durante el arranque, la aplicación inicia en el modo de navegación exterior, presentando una interfaz basada en Google Maps orientada a desplazamientos en espacios abiertos. Esta pantalla ofrece al usuario una representación clara de su entorno y acceso inmediato a herramientas de orientación y búsqueda.

La ubicación actual del usuario se indica mediante un punto azul sobre el mapa. En la esquina inferior derecha se sitúa un botón que permite recentrar la vista en su posición, mientras que en la parte superior izquierda se encuentra un icono de lupa que activa la funcionalidad de búsqueda de destinos (Figura 5.1).

En caso de que se detecte una baliza BLE posteriormente, el sistema notifica al

usuario mediante un *popup* con la opción de cambiar al modo de guiado interior. Si el usuario decide rechazar esta sugerencia, la aplicación incorpora un botón adicional en la esquina superior derecha que permanece visible mientras la baliza esté activa. Este botón permite reconsiderar la transición al modo interior en cualquier momento durante la sesión.

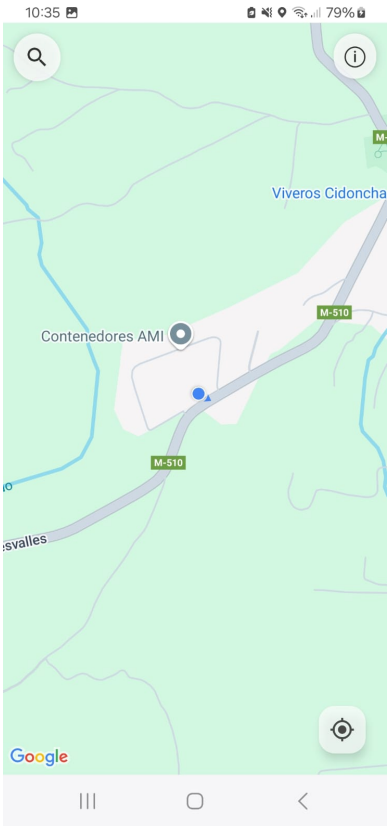

Figura 5.1: Pantalla principal.

5.3. Búsqueda de destinos

El sistema de navegación implementado en la aplicación incorpora un único **componente de búsqueda reutilizable**, accesible tanto desde la vista de navegación exterior como desde la interfaz de guiado interior. Esta decisión de diseño permite ofrecer una experiencia de usuario coherente y unificada, independientemente del contexto espacial en el que se encuentre el usuario.

En el modo de **navegación exterior**, el componente de búsqueda se activa al pulsar el **icono de la lupa**, ubicado de forma visible en la parte superior de la interfaz (Figura 5.1). Esta acción despliega la barra de búsqueda sobre el mapa, permitiendo al usuario introducir manualmente el nombre del destino deseado.

Por el contrario, en el modo de **guiado interior**, la barra de búsqueda permanece **siempre visible** en la interfaz, ya que el usuario puede estar en constante desplazamiento dentro del edificio y requerir cambiar su destino de forma inmediata (Figura 5.6). Esta diferencia busca optimizar la accesibilidad a la funcionalidad según el contexto de uso.

En ambos casos, a medida que se escribe texto en la barra, el componente lanza una consulta automática al backend de la aplicación. Este servicio responde dinámicamente con un conjunto de hasta cinco recomendaciones, seleccionadas en función de coincidencias parciales con los destinos registrados. Cada recomendación muestra el **nombre del nodo** y el **edificio correspondiente**, lo cual facilita una selección precisa del destino incluso cuando hay nombres similares en distintos edificios.

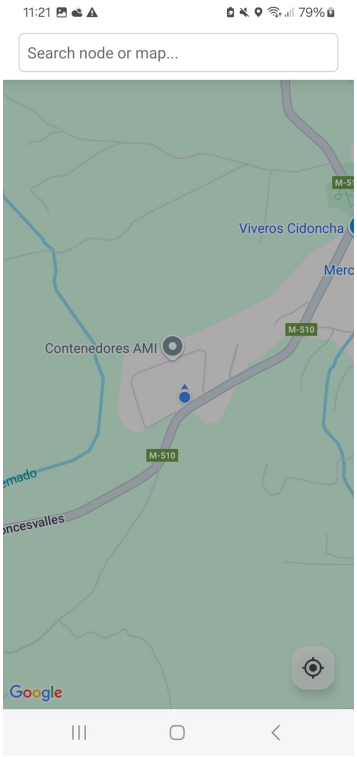

Figura 5.2: Barra de búsqueda desplegada al pulsar el icono de lupa.

Una vez generadas las recomendaciones, estas se muestran en forma de lista inmediatamente bajo la barra de entrada (Figura 5.3). El usuario puede seleccionar cualquiera de ellas tocando sobre su nombre. Al hacerlo, el sistema interpreta esta selección como la definición del destino, y automáticamente calcula la ruta óptima desde la ubicación actual hasta el punto seleccionado.

El trazado de la ruta varía en función del contexto: si el usuario se encuentra en el exterior, la ruta se calcula mediante el servicio de mapas; si está en un

entorno interior con cobertura BLE, se activa el guiado mediante el algoritmo A*. En ambos casos, la ruta se dibuja directamente sobre el mapa (Figura 5.4).

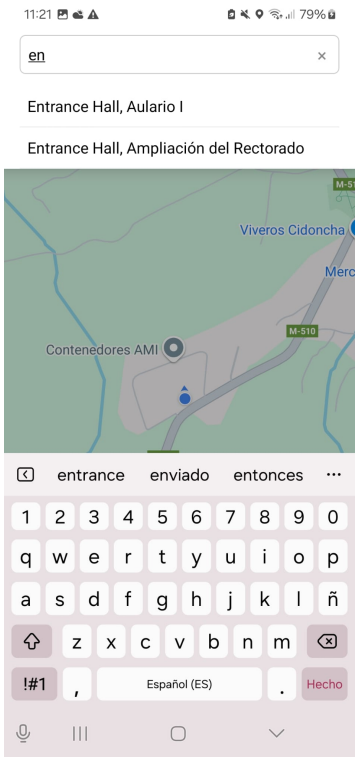

Figura 5.3: Lista de recomendaciones con nodo y edificio.

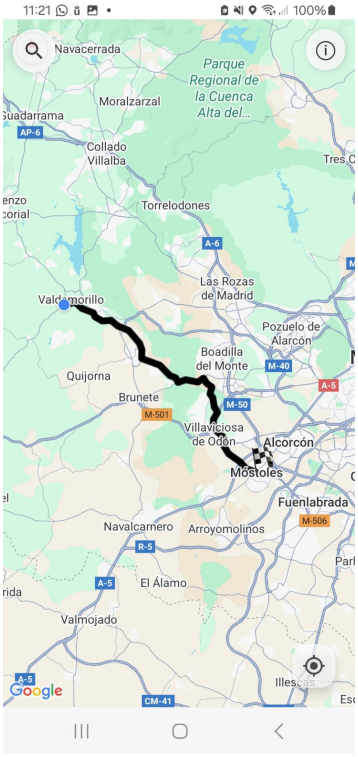

Figura 5.4: Ruta trazada desde la ubicación actual hasta el destino seleccionado.

Una vez seleccionado el destino, la barra de búsqueda permanece visible con el nombre del nodo elegido, acompañado por un icono en forma de **"X"** a la derecha. Este icono actúa como botón de cancelación: al pulsarlo, se elimina el destino actual y se restablece la barra para introducir un nuevo término. Alternativamente, el usuario puede borrar el contenido manualmente para reiniciar el proceso de búsqueda.

En la pantalla de **navegación exterior**, también es posible ocultar temporalmente la interfaz de búsqueda tocando cualquier parte del mapa fuera del área de la barra. Esta acción no cancela la navegación en curso ni elimina el destino seleccionado; simplemente minimiza la interfaz para ofrecer una vista más limpia del entorno y una mayor visibilidad del trazado exterior. La guía permanece activa en segundo plano hasta que el usuario decida modificar o cancelar el destino. Esta funcionalidad no está disponible en el modo de guiado interior, donde la

barra de búsqueda se mantiene fija como parte esencial de la interfaz.

En resumen, este componente unificado de búsqueda constituye un elemento clave dentro del sistema de guiado. Su diseño adaptable permite responder a distintos contextos de navegación sin fragmentar la experiencia del usuario. Además, al centralizar las consultas en un único punto de interacción con el backend, se optimiza la eficiencia en la comunicación de datos y se simplifica la lógica interna de la aplicación, facilitando el mantenimiento y la extensibilidad futura del sistema.

5.4. Detección de beacons y transición a modo interior

Cuando la aplicación detecta una baliza previamente registrada en su sistema, puede generar un *popup* informativo que notifica al usuario sobre la disponibilidad de navegación interior en la zona actual. Sin embargo, esta notificación no se muestra de forma indiscriminada: solo se activa si en ese momento el usuario no tiene seleccionado un destino en la aplicación. Esta condición evita interrupciones innecesarias durante rutas activas, y garantiza que el cambio de contexto se proponga únicamente cuando no haya una navegación en curso. Si el usuario no tiene seleccionado un destino, el sistema presenta una sugerencia visual que permite elegir entre permanecer en el modo de navegación exterior o activar el modo de guiado interior (Figura 5.5).

En caso de que el usuario decida rechazar el guiado interior en el momento de la detección de la baliza, la aplicación mantiene visible un botón persistente en la parte superior derecha de la pantalla que le permite reactivar la sugerencia más adelante si así lo desea (Figura 5.1).

Por otro lado, si ya se ha seleccionado previamente un destino y se detecta un nodo asociado a la baliza con el atributo `entrada` configurado como `true` para dicho destino, el sistema interpreta automáticamente que el usuario ha llegado a un acceso válido al edificio. En este caso, se realiza la transición directa al modo de guiado interior, sin necesidad de confirmación manual. Esto permite una experiencia fluida y proactiva, eliminando pasos innecesarios cuando la intención del usuario ya ha sido expresada mediante la elección del destino.

Este mecanismo mejora la continuidad del guiado, asegurando que la aplicación reaccione de forma inteligente ante el contexto físico del usuario, activando la visualización del mapa interior y recalculando la ruta desde la entrada detectada hasta el destino final dentro del edificio.

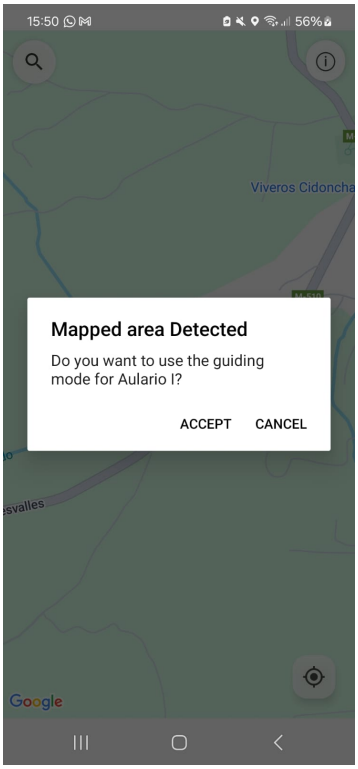

Figura 5.5: Popup que aparece al detectar una baliza Eddystone sin destino seleccionado.

5.5. Pantalla de guiado interior

Cuando el usuario accede al modo de guiado interior, ya sea inmediatamente al iniciar la aplicación, tras detectar una baliza cercana, o como parte de una transición desde el entorno exterior, la aplicación presenta una pantalla especializada para la navegación en espacios interiores. Esta vista incorpora el mismo **componente de búsqueda unificado** descrito anteriormente, que permanece visible en la parte superior de la interfaz para facilitar la consulta o cambio de destino en cualquier momento. El usuario es representado mediante una **flecha verde** que indica tanto su posición actual como la orientación real del dispositivo, permitiendo interpretar con mayor precisión la dirección hacia la que se está desplazando.

El mapa interior muestra el plano del edificio con una representación binaria: las zonas **transitables** aparecen en color **blanco**, mientras que las **paredes u obstáculos** se representan en **negro**, respetando la estructura real del entorno físico. En el encabezado se muestra, centrado, el **nombre del edificio actual**,

y en la esquina superior izquierda aparece un icono de **mapa** que, al ser pulsado, permite regresar a la pantalla de navegación exterior.

En la parte inferior de la interfaz de navegación interior, cuando el edificio cuenta con múltiples plantas, se muestran unas **flechas de navegación vertical**, acompañadas de una etiqueta textual que indica de forma clara el piso actualmente visible. Estas flechas permiten al usuario realizar un cambio manual de nivel, lo cual resulta especialmente útil en situaciones donde se desea planificar rutas alternativas, identificar servicios ubicados en otros pisos o simplemente explorar libremente el entorno interior antes de comenzar el guiado. La transición entre pisos se realiza de forma inmediata y sin recarga completa de la vista, manteniendo la continuidad visual y minimizando interrupciones en la experiencia del usuario.

Figura 5.6: Vista inicial del mapa interior en modo guiado.

Además, para mantener la legibilidad del trayecto, durante el guiado la aplicación representa únicamente el segmento de ruta correspondiente a la planta seleccionada. Esto significa que los tramos de ruta ubicados en otros niveles del edificio permanecen ocultos hasta que el usuario accede manualmente o automáticamente a ese piso. Esta segmentación visual contribuye de manera significativa a una interpretación clara y no ambigua del trayecto, evitando la superposición de caminos entre niveles y reduciendo la sobrecarga cognitiva. Una explicación detallada de esta lógica por piso, junto con ejemplos gráficos que ilustran el com-

portamiento dinámico de la visualización, puede consultarse en el **Anexo C** (ver sección C.1).

Asimismo, en la esquina inferior derecha de la pantalla, se encuentra un **botón de centrado** que reposiciona automáticamente la vista del mapa sobre la ubicación actual estimada del usuario. Esta funcionalidad es especialmente útil en escenarios donde el usuario ha desplazado o alejado el mapa mediante gestos táctiles, por ejemplo, para consultar otras áreas del plano, y desea volver rápidamente a su posición actual sin pérdida de orientación. Esta herramienta se integra bajo la misma lógica de interacción ya empleada en la navegación exterior, ofreciendo consistencia entre modos. Entre las funcionalidades interactivas disponibles se incluyen acciones como **zoom** mediante gesto de pinza, **paneo** con desplazamiento directo, y ajuste del nivel de detalle mediante controles de escala adaptativa.

Una vez que el usuario ha seleccionado un destino y se inicia el guiado, la ruta interior se traza gráficamente sobre el plano mediante una combinación de elementos visuales intuitivos. Los puntos intermedios del trayecto, denominados **nodos de ruta**, se representan mediante **círculos de colores**, los cuales facilitan la identificación visual de puntos clave, como cambios de dirección, cruces o accesos a escaleras o ascensores. Para indicar la dirección del desplazamiento, entre estos nodos se dibuja una **flecha naranja** continua, que guía de forma explícita el camino que el usuario debe seguir. Esta flecha actúa como hilo conductor visual del recorrido, asegurando claridad en entornos complejos con múltiples conexiones. Al llegar al último nodo visible del piso actual, que puede ser el destino o un punto de transición hacia otro nivel, la flecha termina en una punta triangular dirigida al nodo correspondiente, marcando el final de ese tramo. El destino final se distingue del resto mediante un **icono de bandera**, el cual se ubica sobre el último nodo de la ruta y permite al usuario identificar de forma inequívoca el objetivo del guiado. Ejemplos ilustrativos de esta representación pueden observarse en las Figuras 5.8 y 5.9, que muestran respectivamente un nodo de salida sin bandera y un nodo de destino claramente identificado. Además, la Figura 5.10 ilustra cómo se representa al usuario si este se encuentra en un nivel distinto al que se está visualizando actualmente.

Además del guiado visual, la interfaz incorpora elementos interactivos contextuales que enriquecen la experiencia general de navegación. Durante el recorrido, el usuario puede acceder a información adicional sobre el tramo actual o el destino mediante un botón informativo, y al finalizar la ruta se presenta una pantalla resumen con opciones de confirmación. Esta incluye acciones como cerrar el guiado o mantener la ruta visible. Estos mecanismos permiten al usuario conservar el control sobre la navegación en todo momento. Una descripción más detallada de esta lógica de interacción contextual, acompañada de capturas representativas, se ofrece en el **Anexo C** (ver sección C.2).

Finalmente, cuando el sistema detecta que el usuario ha alcanzado el nodo correspondiente al destino, se activa automáticamente un *popup* de confirmación

(ver Figura 5.7). Este mensaje informa al usuario de que se ha llegado al punto final del recorrido y presenta dos opciones: al pulsar **OK**, el sistema finaliza el guiado, oculta la ruta del mapa y deja la vista limpia para futuras acciones; al seleccionar **Cancel**, el trazado permanece visible en pantalla, permitiendo al usuario seguir explorando el plano, verificar puntos cercanos o corregir posibles errores de posicionamiento. Este enfoque contribuye a una experiencia de navegación flexible, fluida y centrada en el control por parte del usuario, adaptándose tanto a escenarios de uso dirigidos como a exploraciones libres.

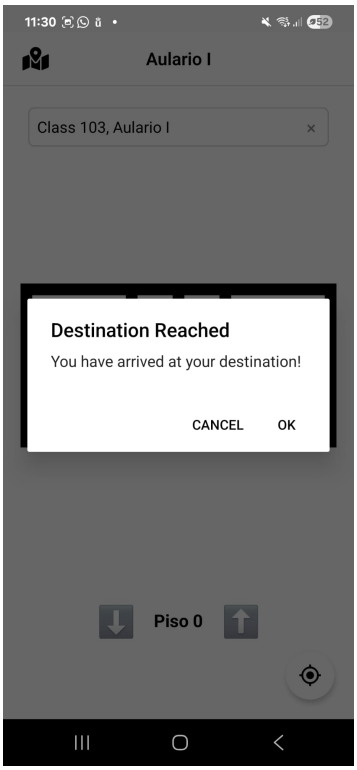

Figura 5.7: Popup mostrado al alcanzar el destino, con opciones para finalizar o mantener el guiado.

5.6. Transición entre edificios con cambio automático de modo

El sistema también contempla de forma inteligente los casos en los que el destino seleccionado se encuentra en un edificio diferente al actual. Si el usuario se encuentra en modo de **guiado interior** y establece como destino un nodo perteneciente a otro edificio, la aplicación inicia automáticamente una secuencia de navegación encadenada entre interiores y exteriores. En primer lugar, se identifica el nodo de salida más cercano dentro del edificio actual, es decir, aquel que posee

el atributo `exit=true`, y se genera una ruta hasta dicho punto. Este nodo actúa como punto de transición hacia el exterior, pero no se presenta como destino final (no muestra el icono de bandera), como se observa en la Figura 5.8.

Una vez alcanzado este nodo de salida, el sistema cambia al modo de **navegación exterior** y calcula una ruta utilizando mapas globales para guiar al usuario desde su posición actual hasta las inmediaciones del edificio de destino.

Durante el trayecto exterior, la aplicación permanece atenta a señales BLE. Cuando detecta una baliza asociada a un nodo con el atributo `entry=true` dentro del edificio de destino, conmuta automáticamente al modo de **guiado interior**, carga el plano correspondiente y genera la ruta desde ese nodo de entrada hasta el destino original. En este último tramo, se aplica la misma lógica de representación descrita en la sección anterior: los nodos intermedios se indican con círculos de colores, el trayecto se visualiza con flechas naranjas entre nodos, y el nodo de destino se destaca con un icono de bandera (véanse Figuras 5.10 y 5.9).

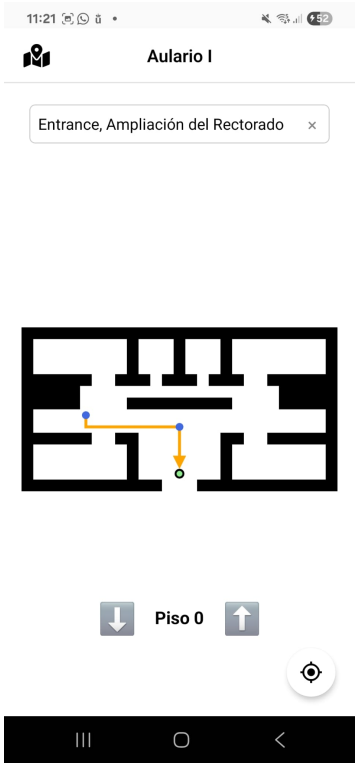

Figura 5.8: Guiado interior hacia el nodo de salida (`exit=true`).

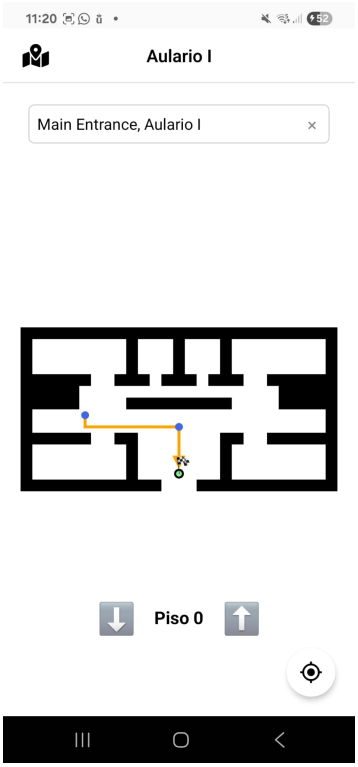

Figura 5.9: Guiado interior dentro del edificio de destino. El nodo de llegada está marcado con el icono de bandera.

En ambas imágenes anteriores (5.8 y 5.9), no se muestra el triángulo verde que representa al usuario. Esto se debe a que, en los ejemplos presentados, el usuario se encuentra físicamente en una planta superior a la visualizada. Esta situación puede darse, por ejemplo, al revisar manualmente una planta diferente. La Figura 5.10 muestra cómo aparece el triángulo del usuario cuando se visualiza el piso correspondiente.

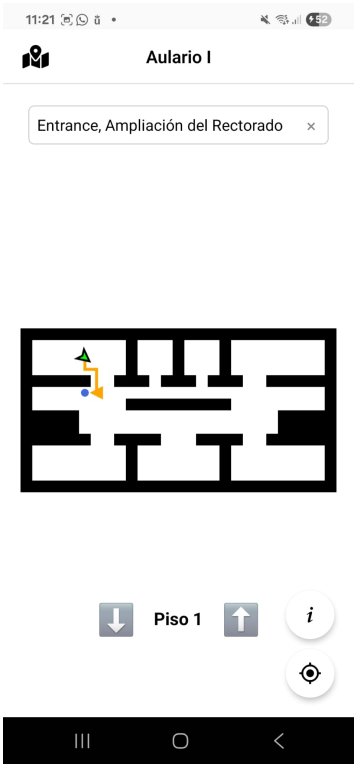

Figura 5.10: Representación del usuario mediante triángulo verde al visualizar el piso correcto.

Este comportamiento automatizado permite que el usuario recorra trayectos complejos, que incluyan cambios entre edificios y modos de navegación, sin necesidad de realizar acciones adicionales. Todo el proceso de transición está diseñado para ser fluido y contextual, maximizando la continuidad del guiado y adaptándose dinámicamente al entorno físico detectado por el dispositivo.

5.7. Resumen del flujo de navegación

La interfaz del sistema ha sido diseñada para guiar al usuario de forma fluida a lo largo de trayectos que pueden abarcar tanto espacios exteriores como interiores, incluso con transiciones entre edificios. La navegación se estructura en pasos progresivos que se adaptan dinámicamente al contexto:

1. El usuario inicia en la pantalla de navegación exterior, donde puede buscar un destino utilizando un componente de búsqueda unificado.

2. Si el destino se encuentra en otro edificio, se calcula una ruta exterior mediante Google Maps y se visualiza directamente sobre el mapa.

3. Al acercarse a una zona con cobertura BLE, la aplicación detecta automáticamente las balizas y, si se identifica una entrada válida relacionada con el destino, se activa el guiado interior sin necesidad de confirmación adicional.

4. Dentro del edificio, se muestra una vista especializada que representa el entorno en un plano por pisos. La ruta se adapta dinámicamente al nivel visualizado, destacando solo el tramo correspondiente a esa planta (ver Anexo C.1).

5. Si el trayecto interior requiere pasar por varias plantas, el usuario puede navegar manualmente entre niveles mediante flechas de control.

6. Durante la ruta, se ofrece información contextual sobre el tramo actual mediante un botón específico, y al alcanzar el nodo destino, se presenta un popup de confirmación con opciones para finalizar o mantener el guiado.

Este flujo asegura una transición continua y autónoma entre entornos exteriores e interiores, manteniendo siempre visible el destino y permitiendo al usuario interactuar con la navegación sin complejidad ni interrupciones.

5.8. Vídeos demostrativos

Para complementar la descripción detallada del sistema, se han elaborado dos vídeos demostrativos que ilustran su funcionamiento en escenarios reales y prácticos. Estas grabaciones permiten observar de forma directa la interacción con la aplicación, la lógica del guiado, y el comportamiento adaptativo del sistema en función del contexto físico del usuario. A continuación, se describen brevemente ambos casos:

- **Vídeo 1 – Guiado interior–exterior–interior con transiciones automáticas:**
  Este vídeo muestra un recorrido completo que combina navegación interior y exterior con transiciones automáticas gestionadas por la detección de balizas BLE. En este caso concreto, el usuario parte desde el interior del **Aulario II** y se dirige a una clase situada en el **Aulario I**. Durante el trayecto, la aplicación cambia automáticamente entre modos de guiado en función del entorno (interior o exterior), mostrando la adaptabilidad del

sistema y la continuidad en la experiencia de navegación.
Enlace directo: `https://vimeo.com/1095542197/f129138e35?share=copy`

- **Vídeo 2 – Navegación en interiores detallada:**
  Este segundo vídeo se centra exclusivamente en la navegación dentro de un edificio, ilustrando el guiado planta a planta y el uso de balizas para el posicionamiento preciso. El escenario representado corresponde al acceso desde la **entrada principal del Aulario I** hasta una **aula ubicada en la segunda planta**. La aplicación calcula la ruta más adecuada, traza el recorrido por niveles y adapta la visualización en función del piso mostrado. Se destacan elementos clave como el cambio automático de plantas, el uso de iconos contextuales, la claridad del trazado sobre el plano, así como la visualización del funcionamiento del botón de información adicional durante el recorrido.
  Enlace directo: `https://vimeo.com/1094743646/f17dacca9b?share=copy`

Estos recursos audiovisuales refuerzan la comprensión del sistema desde una perspectiva práctica, complementando la documentación técnica escrita con ejemplos reales de uso en entornos universitarios.

6

Pruebas y Validación

Este capítulo presenta los procedimientos realizados para verificar el correcto funcionamiento del sistema, así como los resultados obtenidos en distintos escenarios de prueba. El objetivo es evaluar la eficacia, precisión y robustez de la aplicación en condiciones reales y simuladas.

6.1. Escenarios de prueba

Para validar el comportamiento integral del sistema, se definieron distintos escenarios combinados que reproducen trayectos reales en los que se alternan entornos exteriores e interiores. Estos escenarios reflejan los flujos de navegación descritos en el capítulo anterior, incluyendo la detección automática de entradas y salidas , la visualización por pisos, el uso del componente de búsqueda unificado y la activación de elementos contextuales.

Las pruebas se llevaron a cabo en dos entornos reales:

- **Exterior:** un entorno urbano abierto, donde se evaluó el uso de la API de Google Maps con geolocalización GPS y la capacidad de trazar rutas entre edificios desde la pantalla de navegación global.
- **Interior:** un edificio académico de múltiples pasillos y plantas, equipado con balizas BLE tipo Eddystone colocadas estratégicamente en puntos clave como intersecciones, accesos, escaleras y salidas.

Los escenarios de prueba incluyeron rutas con cambios de planta, entradas y salidas automáticas entre edificios, pérdida o recuperación de señal BLE, así como decisiones manuales del usuario (como cancelación de destino o cambio de nivel visualizado). Esta cobertura permitió validar tanto la robustez del guiado continuo como la flexibilidad de la interfaz en condiciones reales de uso.

6.2. Metodología de validación

La metodología de validación consistió en medir tres variables clave: la **precisión de localización**, entendida como la distancia promedio entre la posición estimada por el sistema de RSSI y la posición real; la **fiabilidad del escaneo BLE**, definida como el porcentaje de veces que una baliza activa es correctamente detectada dentro de su área prevista; y el **consumo de batería**, medido a partir de la variación del nivel de batería durante el uso de la aplicación en escenarios de navegación interior. Para cada prueba se utilizó un cronómetro, así como herramientas de *logging* interno integradas en la propia aplicación, con el fin de registrar eventos, coordenadas y tiempos de forma precisa.

6.3. Resultados obtenidos

Los resultados promedio de las pruebas realizadas se presentan a continuación:

- **Precisión media de localización (RSSI):** ± 3.4 metros.
- **Tasa de detección de balizas esperadas:** 95.2 %.
- **Consumo de batería en modo guiado interior (15 minutos):** 6–8 % en dispositivo Android.

Estos resultados se consideran adecuados para un sistema de guiado no crítico, enfocado en usabilidad y bajo consumo.

6.4. Análisis y discusión

Los experimentos mostraron que el sistema ofrece una experiencia fluida de navegación, con un margen de precisión razonable para entornos cerrados. La elección del algoritmo **A*** garantiza rutas óptimas y eficientes, adaptadas a la geometría real del entorno. Sin embargo, en edificios con múltiples niveles y una alta densidad de nodos, el tiempo de cómputo puede incrementarse, por lo que

se recomienda aplicar optimizaciones como la reducción de nodos intermedios o el uso de heurísticas específicas por planta.

El escaneo BLE resultó funcional, aunque susceptible a interferencias provocadas por estructuras metálicas, presencia de personas o paredes gruesas. Para mitigar estos efectos, se concluyó que la colocación óptima de las balizas debe hacerse en zonas despejadas, con la menor cantidad posible de obstrucciones físicas entre emisor y receptor. Además, la estrategia actual, que permite transicionar automáticamente al modo interior al detectar un nodo de entrada válido, ha demostrado ser más robusta que sistemas basados en notificaciones manuales, reduciendo la carga de decisión para el usuario.

Respecto al consumo energético, los niveles registrados son consistentes con el uso continuo de Bluetooth y servicios de localización. La duración de la batería puede optimizarse ajustando la frecuencia del escaneo BLE y reduciendo las actualizaciones visuales en pantalla cuando el usuario permanece estático.

6.5. Limitaciones encontradas

Durante las pruebas funcionales del sistema se identificaron varias limitaciones que, si bien no afectan de forma crítica a la operatividad general, sí evidencian áreas claras de mejora técnica y escalabilidad:

- **Variabilidad del RSSI:** La intensidad de la señal recibida (`RSSI`) se vio afectada por obstáculos físicos, interferencias ambientales y la orientación del dispositivo. Aunque el sistema es capaz de mantener la funcionalidad de guiado, estas variaciones pueden generar imprecisiones momentáneas en entornos interiores densos.
- **Ubicación de balizas:** Se constató que la disposición física de las balizas BLE influye de manera significativa en la estabilidad y fiabilidad de la detección. En zonas con obstáculos, circulación frecuente o interferencias, se recomienda su instalación en ubicaciones elevadas, abiertas y libres de elementos metálicos.
- **Escalabilidad computacional:** Aunque el algoritmo **A*** ofrece rutas eficientes y óptimas, su rendimiento puede verse comprometido en grafos muy densos con múltiples niveles y conexiones interplanta. En estos casos, puede ser necesario aplicar estrategias de optimización, como partición por pisos o simplificación del grafo.

Estas limitaciones han sido documentadas como parte del proceso de validación y constituyen la base para futuras mejoras del sistema, que se detallan en el capítulo siguiente.

7

Conclusiones y Trabajos Futuros

En este capítulo se resumen los principales resultados alcanzados durante el desarrollo del sistema de guiado en interiores y exteriores, se identifican las limitaciones encontradas y se proponen posibles mejoras y líneas de investigación futuras.

7.1. Conclusiones

El presente Trabajo de Fin de Grado ha culminado con el diseño e implementación de una aplicación móvil capaz de proporcionar guiado tanto en exteriores, mediante el uso de la API de Google Maps, como en interiores, mediante el uso de balizas BLE tipo Eddystone.

Uno de los principales logros ha sido el desarrollo de una aplicación funcional utilizando React Native con Expo. La interfaz resultante es clara, accesible y compatible con múltiples plataformas. Asimismo, se ha logrado una integración adecuada del sistema de localización exterior, basándose en GPS y Google Maps, lo que permite al usuario realizar búsquedas de direcciones, recibir recomendaciones y visualizar rutas de manera eficiente.

En el ámbito del guiado en interiores, se ha implementado un sistema de detección de balizas BLE, acompañado de un mecanismo de notificación contextual que ofrece al usuario la posibilidad de cambiar al modo de navegación interior. Para representar los mapas de espacios cerrados, se ha diseñado un modelo de

grafo que facilita el cálculo de rutas óptimas mediante el algoritmo de A*.

El sistema también cuenta con una API REST desarrollada con Spring Boot, orientada exclusivamente a la lectura de datos. Esta API proporciona la información necesaria para el cálculo de rutas, de forma modular y eficiente. Además, se ha llevado a cabo una validación funcional en entornos reales, cuyos resultados han sido satisfactorios en términos de precisión, tiempos de respuesta y experiencia de usuario. Finalmente, se han considerado aspectos fundamentales relacionados con la seguridad y privacidad del sistema, evitando el almacenamiento de datos personales o de localización en servidores externos.

En conjunto, el trabajo desarrollado ha permitido comprobar la viabilidad de integrar tecnologías abiertas y accesibles en una solución coherente de guiado híbrido. La arquitectura del sistema, estructurada en módulos claramente definidos, permite futuras ampliaciones sin comprometer la estabilidad del conjunto. La lógica de transición automática entre contextos interiores y exteriores, así como la visualización adaptativa de rutas por planta, suponen una aportación relevante desde el punto de vista de la experiencia de usuario.

Aunque se han alcanzado los objetivos planteados inicialmente, el sistema no está exento de retos. Las decisiones de diseño adoptadas han priorizado la simplicidad, la estabilidad y la demostrabilidad, lo que ha permitido establecer una base sólida sobre la cual se pueden construir mejoras significativas. Las limitaciones detectadas durante la implementación abren un abanico de posibilidades de evolución, que se detallan en las siguientes secciones, junto con propuestas concretas para ampliar la funcionalidad del sistema y explorar nuevas líneas de desarrollo.

7.2. Limitaciones detectadas

Durante el proceso de desarrollo e implementación del sistema, se han identificado diversas limitaciones que ofrecen oportunidades claras de mejora para versiones futuras. Algunas de estas ya fueron tratadas con mayor detalle en el capítulo de validación experimental (véase sección 6.5), mientras que otras derivan del análisis global del diseño y el alcance funcional actual del sistema.

En primer lugar, la técnica de estimación de distancia mediante el valor RSSI, si bien funcional, ha demostrado ser sensible a obstáculos físicos y su rendimiento puede degradarse en entornos densos. Aunque útil en general, su estabilidad podría mejorarse mediante la incorporación de otras señales complementarias o sensores inerciales.

En segundo lugar, el sistema no cuenta con mecanismos de autenticación o gestión de perfiles, lo que limita su aplicación en contextos donde se requiera

personalización del guiado, control de accesos o registro individualizado del uso.

En cuanto al backend, la arquitectura actual opera únicamente en modo de lectura, sin capacidad para registrar eventos, personalizar datos o reaccionar a patrones de uso. Esta limitación reduce la flexibilidad del sistema ante cambios dinámicos en el entorno o las preferencias de los usuarios.

Por último, aunque la implementación final opta por el algoritmo A* (frente al inicialmente considerado Dijkstra), siguen existiendo retos relacionados con la escalabilidad en grafos complejos, especialmente aquellos de alta densidad y múltiples niveles. Esta cuestión fue abordada en la validación técnica y continúa siendo una línea de mejora relevante para futuros desarrollos.

7.3. Trabajos futuros

A partir de los resultados obtenidos, se plantean diversas líneas de trabajo e investigación orientadas a mejorar y evolucionar el sistema actual. Una de ellas consiste en incorporar algoritmos de localización híbridos que combinen la tecnología BLE con otras como Wi-Fi fingerprinting o UWB, con el fin de aumentar la precisión en espacios interiores. Asimismo, se propone la implementación de mecanismos de autenticación de usuarios y gestión de roles, lo que permitiría ofrecer rutas personalizadas, restringir accesos o adaptar la experiencia según las preferencias individuales.

Otra mejora relevante sería la integración de un sistema de gestión de contenidos desde el backend, que permita añadir o modificar puntos de interés, balizas o mapas a través de una interfaz web. También se contempla la aplicación de técnicas de aprendizaje automático para predecir rutas habituales, optimizando así la experiencia del usuario. En paralelo, se sugiere diseñar un módulo de navegación accesible para personas con movilidad reducida, que priorice trayectos con ascensores, rampas u otros elementos adaptados.

Además, se plantea la incorporación de un sistema de analítica de uso, siempre con respeto a la privacidad de los usuarios, con el objetivo de evaluar el comportamiento del sistema en entornos reales y facilitar decisiones basadas en datos. Por otro lado, se propone como alternativa al uso de mapas interiores la visualización de una simple flecha que indique la dirección hacia el siguiente nodo. Esta solución implicaría un menor esfuerzo de implementación y mantenimiento, al eliminar la necesidad de digitalizar planos y actualizarlos ante posibles cambios estructurales. No obstante, podría presentar la desventaja de ofrecer al usuario un contexto espacial limitado, dificultando la orientación en entornos complejos o con varios niveles.

En definitiva, el sistema desarrollado constituye una base sólida, escalable

y adaptable a distintos tipos de edificios, como campus universitarios, centros comerciales u hospitales, sobre la cual pueden implementarse futuras mejoras que amplíen su funcionalidad y alcance.

Bibliografía

[1] S. Fortune, "Voronoi diagrams and delaunay triangulations," in *Computing in Euclidean Geometry*. World Scientific, 1995, pp. 225–265.

[2] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "A survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1067–1080, 2007.

[3] R. Faragher and R. Harle, "Location fingerprinting with Bluetooth Low Energy beacons," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2418–2428, 2015.

[4] M. Alzantot and M. Youssef, "Robust wireless localization using phase information," in *Proceedings of the 2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2017, pp. 1–10.

[5] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[6] Y. Veres, "Conceptual model of the information system for indoor navigation and positioning based on two-dimensional matrix codes," *Visnyk of the Lviv Polytechnic National University*, pp. 117–132, 2025, accessed June 2025. [Online]. Available: https://science.lpnu.ua/sites/default/files/journal-paper/2025/may/39093/maket25066219052025ves-117-132.pdf

[7] S. Ramamoorthy, A. Yadav *et al.*, "Enhanced indoor pathfinding navigation in multistoried buildings using a hybrid of rrt-connect and dijkstra's algorithm," in *Proceedings of the IEEE Conference on Multi-Agent Systems*. IEEE, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10894337

[8] M. Casuccio, "Ros2-based amr system for mapping and navigation in unknown indoor environments," Master's thesis, Politecnico di Torino, 2024. [Online]. Available: https://webthesis.biblio.polito.it/33156/

[9] T. N. Canh, D. M. Do, and X. HoangVan, "Development of an indoor localization and navigation system based on monocular slam for mobile robots," *arXiv preprint arXiv:2411.05337*, 2024. [Online]. Available: https://arxiv.org/abs/2411.05337

[10] A. M. Yasar, R. L. Voûte, and E. Verbree, "Direct use of indoor point clouds for path planning and navigation exploration in emergency situations," *Delft University of Technology Repository*, 2024. [Online]. Available: https://repository.tudelft.nl/record/uuid:735385eb-535b-4971-b88b-f02d3ea6e744

[11] "Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 (general data protection regulation)," https://eur-lex.europa.eu/eli/reg/2016/679/oj, 2016, official Journal of the European Union, L119, 1–88.

Apéndices

A

Diagrama de secuencia del sistema

La Figura A.1 muestra un diagrama de secuencia que representa un caso de uso completo en el que un usuario desea ir desde su ubicación actual en el interior del edificio A hasta una habitación específica (habitacion X) en el edificio B.

El proceso comienza cuando el usuario enciende la aplicación móvil. Esta verifica si existen balizas BLE cercanas y, en caso afirmativo, envía la información al servidor propio para comprobar si se trata de una baliza registrada. Si se confirma que es válida, la aplicación muestra la pantalla de guiado interior.

El usuario introduce el destino deseado (habitacion X del edificio B) mediante la barra de búsqueda. Cada vez que escribe en esta barra, el cliente envía al servidor propio el texto introducido y el número deseado de sugerencias. El servidor responde con una lista de coincidencias de nodos y edificios registrados. Este comportamiento se describe en detalle en la Sección 5.3. Una vez el usuario selecciona una de las sugerencias, el teléfono genera la ruta interior de salida del edificio A.

Al salir, el dispositivo obtiene la posición actual mediante GPS y la coordenada del edificio de destino, obtenida previamente durante la búsqueda. Con estos datos, el cliente solicita una ruta exterior a los servidores de Google, que devuelven el trazado correspondiente en el mapa.

Cuando el usuario se aproxima al edificio B, la aplicación detecta una nueva baliza. Esta se verifica nuevamente con el servidor propio, y si se confirma que se trata de una baliza válida y clasificada como baliza de entrada, la aplicación cambia automáticamente a la interfaz de guiado interior del edificio B y genera la ruta hasta la habitación X.

Cabe destacar que el diagrama de la Figura A.1 muestra una secuencia representativa del flujo de navegación, pero no refleja explícitamente ciertos procesos que se ejecutan de forma continua o en segundo plano. En particular, la aplicación móvil mantiene activo en todo momento un escaneo pasivo de balizas BLE, incluso cuando el usuario no está interactuando directamente con el sistema. Este comportamiento, descrito detalladamente en la Sección 4.2, permite detectar balizas nuevas o desconocidas y clasificarlas en tiempo real sin interrumpir la experiencia

del usuario. Cada vez que se identifica una baliza no registrada, se comunica con el servidor propio para su validación, sin afectar al estado actual del guiado.

Del mismo modo, en la pantalla de navegación exterior, la aplicación realiza peticiones periódicas a los servidores de Google para actualizar la posición del usuario sobre el mapa, especialmente cuando este se encuentra en movimiento. Este comportamiento es parte del funcionamiento descrito en la Sección 5.2, y garantiza que la visualización del mapa se mantenga sincronizada con la ubicación real del usuario. En los casos en que la aplicación se inicia y no se detecta ninguna baliza cercana, el sistema activa por defecto este modo exterior, priorizando la interacción con el mapa global y el acceso a las herramientas de búsqueda, como se detalla en las Secciones 5.2.

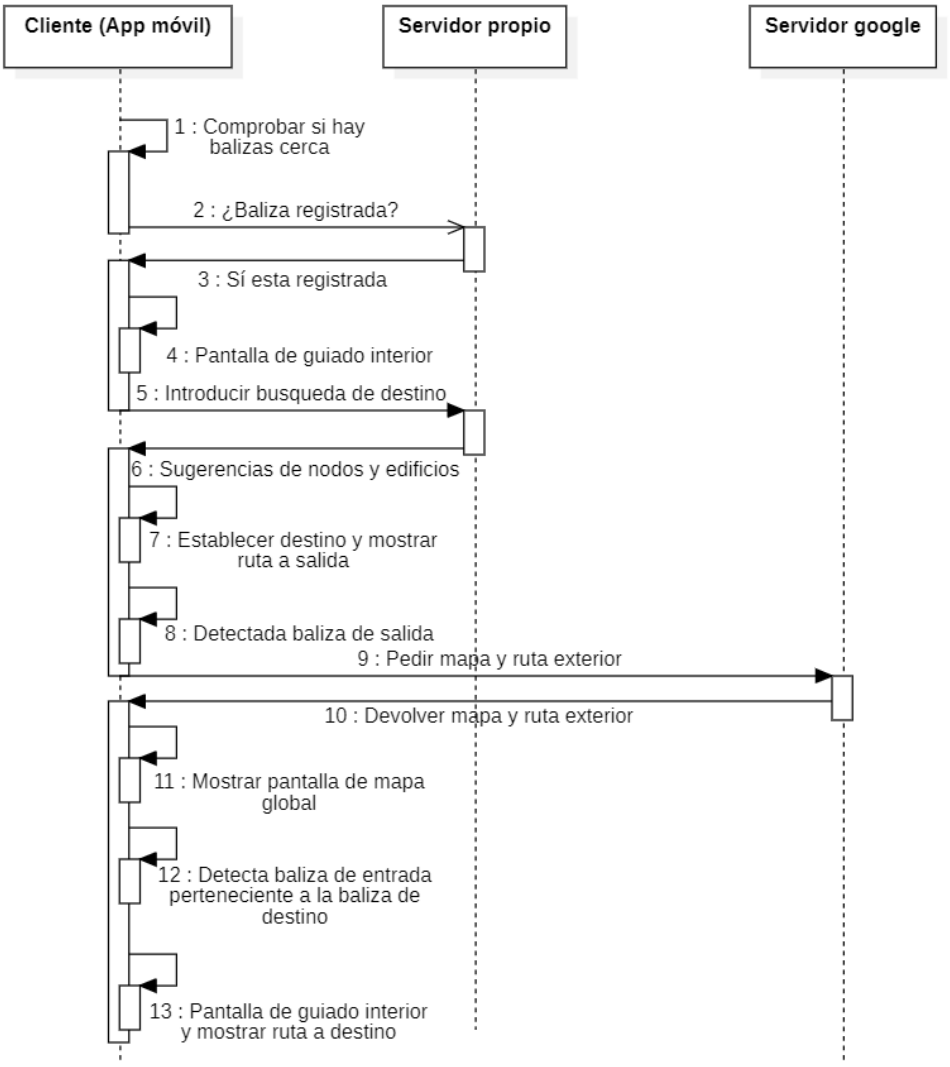

Figura A.1: Diagrama de secuencia para la navegación de un usuario entre dos edificios.

B

Lógica de búsqueda semántica en recomendaciones

El sistema de búsqueda textual implementado en `NodeService.java` permite generar recomendaciones inteligentes en el componente de búsqueda de la aplicación móvil. Esta funcionalidad consiste en analizar el texto introducido por el usuario, dividirlo en palabras clave, y buscar coincidencias parciales tanto en los nombres de los nodos como en los nombres de los edificios o mapas en los que se encuentran.

El proceso comienza con el método `searchByText`, que recibe una cadena de texto introducida por el usuario. Esta entrada se normaliza convirtiéndola a minúsculas y se fragmenta por espacios en una lista de palabras clave. Cualquier término vacío se descarta. A continuación, se construye una consulta dinámica basada en especificaciones JPA, definida en el método auxiliar `buildContainsSpecification`. Esta consulta genera una condición compuesta por predicados `LIKE` con comodines sobre los campos `name` del nodo y del mapa asociado, permitiendo así localizar coincidencias parciales en cualquiera de ellos.

La búsqueda resultante devuelve una lista de nodos candidatos. Para cada uno de ellos, se calcula una puntuación de relevancia. Esta puntuación corresponde al número total de coincidencias de palabras clave tanto en el nombre del nodo como en el del mapa. Así, los resultados más relevantes (aquellos con más coincidencias) obtienen una puntuación superior.

Posteriormente, la lista se ordena en función de dicha puntuación de forma descendente, y en caso de empate, por el identificador del nodo (para garantizar orden estable). Finalmente, se transforma cada nodo en un objeto DTO que contiene su información básica, la del mapa al que pertenece, y la puntuación calculada. La lista resultante se devuelve al cliente, limitada por el número máximo de resultados solicitados.

Esta búsqueda flexible y semántica mejora la experiencia del usuario permitiéndole encontrar destinos relevantes incluso con entradas incompletas o parciales. Su implementación se integra completamente con el componente descrito en la sección 5.3, y forma parte de la lógica de recomendaciones descrita en la sección 4.6.

Pseudocódigo del proceso de búsqueda textual:

```
function searchByText(inputText, maxResults):
    keywords = limpiarYDividir(inputText)
    if keywords está vacío:
        return []

    spec = construirLIKEspec(keywords)
    candidates = buscarNodos(spec)

    scoredList = []
    for nodo in candidates:
        score = 0
        for palabra in keywords:
            if palabra en nombreNodo o nombreMapa:
                score += 1
        scoredList.append((nodo, score))

    ordenar scoredList por score descendente
    ordered = tomar primeros maxResults de scoredList

    results = []
    for nodo, score in ordered:
        dto = crearDTO(nodo, mapa, score)
        results.append(dto)

    return results
```

C

Capturas adicionales de la interfaz

Este apéndice presenta capturas complementarias que ilustran aspectos visuales clave del funcionamiento de la aplicación, centrándose en la interacción del usuario durante el guiado interior, la representación de rutas por niveles y la disponibilidad de información contextual a lo largo del trayecto.

C.1. Visualización de rutas por piso

Cuando el usuario se encuentra en modo de guiado interior, la aplicación permite visualizar la ruta desglosada por niveles del edificio. Cada planta se presenta de forma independiente, mostrando exclusivamente los segmentos del trayecto correspondientes a dicho nivel. Esta organización facilita la comprensión espacial de rutas complejas, especialmente aquellas que implican desplazamientos verticales mediante escaleras o ascensores.

Durante esta navegación, el usuario puede interactuar con el mapa mediante gestos táctiles para desplazarse, hacer zoom o ajustar el encuadre. Además, dispone de un botón de centrado en la esquina inferior derecha para recuperar rápidamente la vista de su posición actual.

La Figura C.3 muestra dos capturas consecutivas de una misma ruta que atraviesa diferentes niveles del edificio: primero en el piso inferior, y luego en el piso superior. Cada imagen representa solo el tramo correspondiente al piso mostrado, manteniendo claridad y coherencia visual.

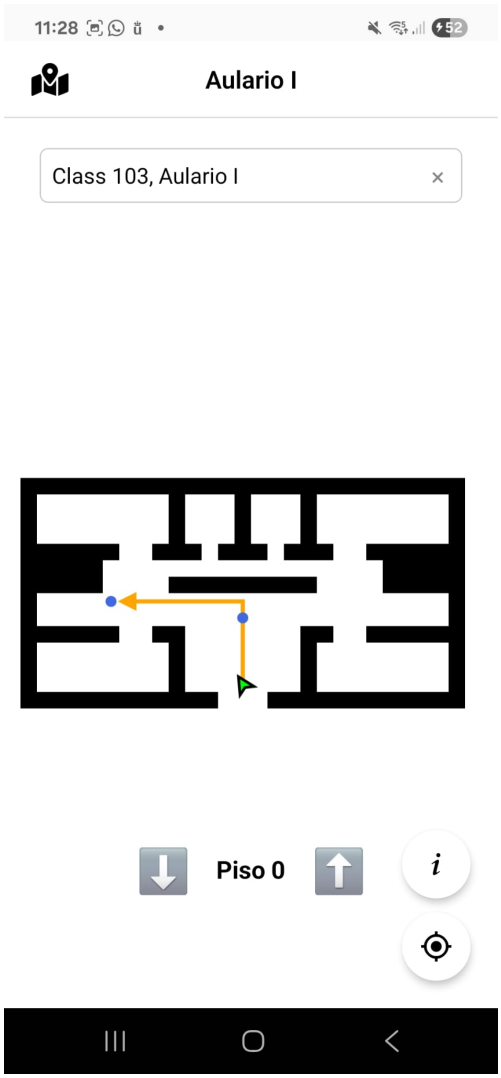

Figura C.1: *
Ruta en el primer piso del edificio.

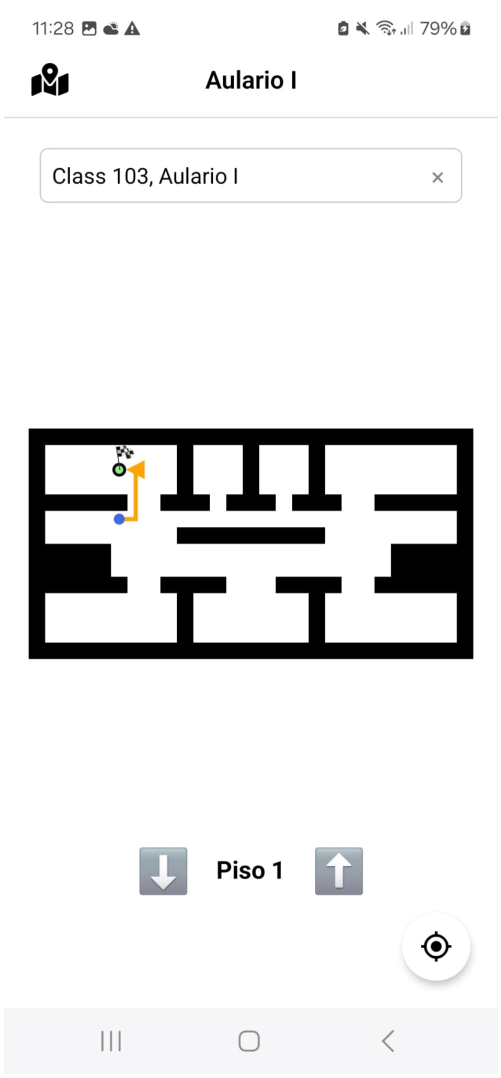

Figura C.2: *
Ruta en el segundo piso del edificio.

Figura C.3: Visualización independiente de la ruta por pisos.

C.2. Interacción contextual durante el guiado interior

En el modo de guiado interior, cuando el usuario tiene un destino activo, aparece un botón adicional sobre el botón de centrado. Este elemento permite acceder a información contextual relacionada con el tramo actual de la ruta.

Al pulsarlo, se despliega un *popup* con una breve descripción del segmento donde el usuario se encuentra, como se muestra en la Figura C.4. Esta funcionalidad está diseñada para mejorar

la comprensión del recorrido, proporcionando indicaciones adicionales en puntos clave como intersecciones, escaleras o zonas de acceso restringido.

Figura C.4: Popup con información adicional sobre el tramo del trayecto.