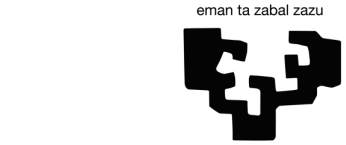

Memoria de Proyecto: SkyVisit

Aitor Gonzalo

22 de julio de 2025

Índice

Índice de figuras

1. Introducción

Esta memoria documenta el desarrollo de una aplicación móvil nativa para Android, desarrollada con Android Studio. El tema de la aplicación es **Lugares de Interes**; en ella podrás ver la localización de distintos sitios relevantes. La aplicación permite visualizar la ubicación de cada lugar en un mapa interactivo, acceder a detalles específicos de cada sitio y gestionar la información mediante una base de datos local. Además, incorpora funcionalidades avanzadas como la capacidad de compartir ubicaciones, notificaciones locales, cambio de idioma, modo oscuro y exportación de datos a un archivo de texto, ofreciendo una experiencia de usuario completa y moderna.

1.1. Nota sobre la funcionalidad del mapa y Google Maps

La aplicación está diseñada para funcionar completamente **sin conexión a internet**, cumpliendo con todos los requisitos mínimos exigidos en la evaluación.
Las funciones de mapa y apertura de Google Maps son elementos **opcionales** que mejoran la experiencia del usuario, pero su uso depende de una conexión a internet. En caso de no haber conexión, la aplicación sigue siendo completamente funcional en la gestión de lugares, almacenamiento en base de datos, notificaciones y demás características locales.

2. Repositorio de Código

El proyecto completo se encuentra disponible en el siguiente repositorio:

`https://github.com/aitorGonzalo/SkyVisit_v1`

3. Elementos de Valoración y Funcionalidades

La aplicación incorpora una serie de elementos que permiten cumplir tanto con los requisitos mínimos obligatorios, como con funcionalidades avanzadas que aportan valor añadido. A continuación, se describen estos elementos, indicando para qué se utilizan y su categorización.

3.1. Requisitos Mínimos

- **RecyclerView + CardView:** Se utiliza para mostrar listados de elementos de forma personalizada. Cada lugar se presenta en un CardView que permite incluir información relevante (nombre y descripción) y facilita la interacción mediante clics para ver detalles, editar o eliminar el elemento.
- **Base de Datos Local:** Implementada mediante SQLite usando las clases `LugarDBHelper` y `LugarDBManager`. Esta base de datos permite listar, añadir, modificar y eliminar lugares, asegurando que la aplicación funcione de manera local sin necesidad de conexión a Internet.
- **Diálogos:** Se utilizan AlertDialogs para confirmar acciones críticas (como la eliminación o edición de un lugar) y para mostrar mensajes informativos o de error, mejorando la interacción y seguridad en las acciones del usuario.

- **Notificaciones Locales:** Se implementan notificaciones para informar al usuario sobre la adición de nuevos lugares. Esto se realiza utilizando la API de notificaciones de Android, asegurando que el usuario reciba feedback inmediato sobre ciertas acciones.
- **Control de la Pila de Actividades:** La navegación entre actividades y fragments se gestiona mediante Intents y un adecuado manejo del ciclo de vida. Esto garantiza que el usuario pueda navegar de forma coherente entre las diferentes pantallas sin perder el estado de la aplicación.

3.2. Elementos Avanzados

- **Fragments:** Se utilizan para adaptar la interfaz según la orientación y el tamaño del dispositivo. La pantalla de detalles se divide en dos fragments: `InfoLugarFragment` (que muestra información del lugar) y `MapaLugarFragment` (que muestra un MapView con la localización). Esto mejora la modularidad y la experiencia de usuario en dispositivos con diferentes dimensiones.
- **Multiidioma:** La aplicación permite cambiar el idioma de la interfaz a través de un diálogo. Mediante el uso de recursos de cadenas y `SharedPreferences`, se guarda la preferencia del usuario y se aplica de forma dinámica, ofreciendo una experiencia personalizada a usuarios de distintos idiomas.
- **Barra de Acciones (Toolbar) Personalizada:** Se ha implementado una Toolbar en la actividad principal que facilita la navegación y la interacción, incorporando elementos interactivos como un switch para activar o desactivar el modo oscuro.
- **Estilos y Temas Personalizados:** Se han definido estilos y temas propios en los archivos `styles.xml` y `themes.xml`, permitiendo personalizar fondos, botones, textos y otros elementos visuales. Esto mejora la apariencia general de la aplicación y la diferencia de los temas por defecto del framework.
- **Preferencias:** Mediante `SharedPreferences` se guardan configuraciones del usuario (como el modo oscuro o el idioma seleccionado), lo que permite personalizar y mantener la experiencia del usuario entre sesiones.
- **Intents Implícitos:** Se utilizan para interactuar con otras aplicaciones. Por ejemplo, se emplean para abrir Google Maps y mostrar la ubicación de un lugar o para compartir la dirección a través de otras aplicaciones de mensajería o correo electrónico.
- **Ficheros de Texto:** Se ha añadido la funcionalidad para exportar los datos almacenados en la base de datos a un archivo de texto. Esto permite respaldar la información o utilizarla en otros contextos.

4. Descripción de Clases y Bases de Datos

4.1. Arquitectura

La aplicación sigue un enfoque basado en la Arquitectura en Tres Capas (Three-Tier Architecture), lo que permite separar la presentación, la lógica de negocio y el acceso a datos. Esta separación facilita la mantenibilidad, la escalabilidad y la modularidad del sistema.

- **Presentación:** Se compone de las actividades y fragments que gestionan la interfaz de usuario. Los archivos XML definen los layouts, estilos y temas, ofreciendo una apariencia personalizada y adaptable.
- **Lógica de Negocio:** Se encarga de la navegación y la interacción del usuario, gestionando notificaciones, diálogos e intents implícitos para integrar funcionalidades adicionales (como abrir Google Maps o compartir ubicaciones).
- **Acceso a Datos:** Utiliza una base de datos local (SQLite) para almacenar y gestionar la información de los lugares. Las clases `LugarDBHelper` y `LugarDBManager` permiten realizar operaciones de manera centralizada.

Estos componentes trabajan en conjunto para ofrecer una experiencia de usuario robusta y coherente, garantizando un funcionamiento local completo sin depender de la conexión a Internet.

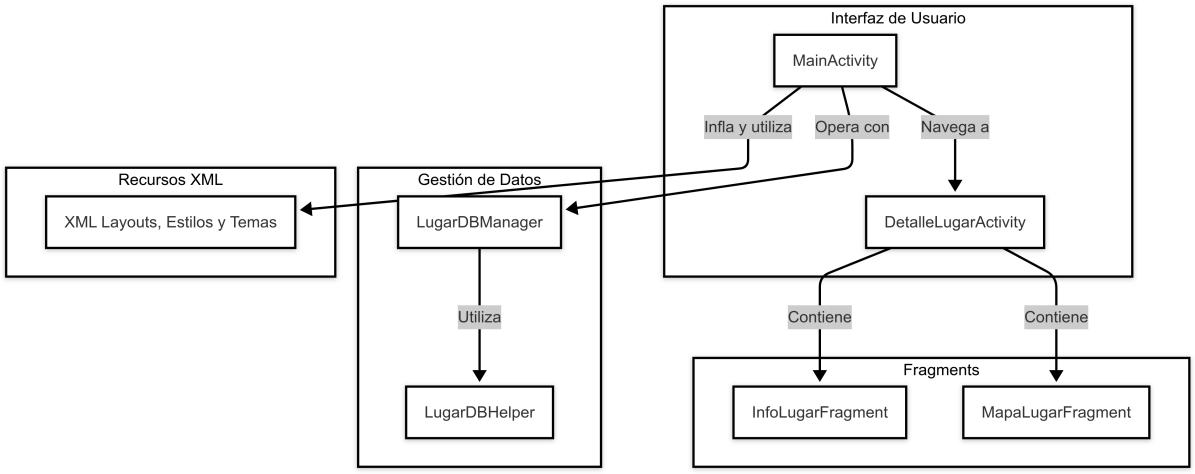

Figura 1: Arquitectura de la Aplicación

4.2. Diagrama de Clases

A continuación se muestra un diagrama de clases que resume el flujo y la relación entre las clases de la aplicación:

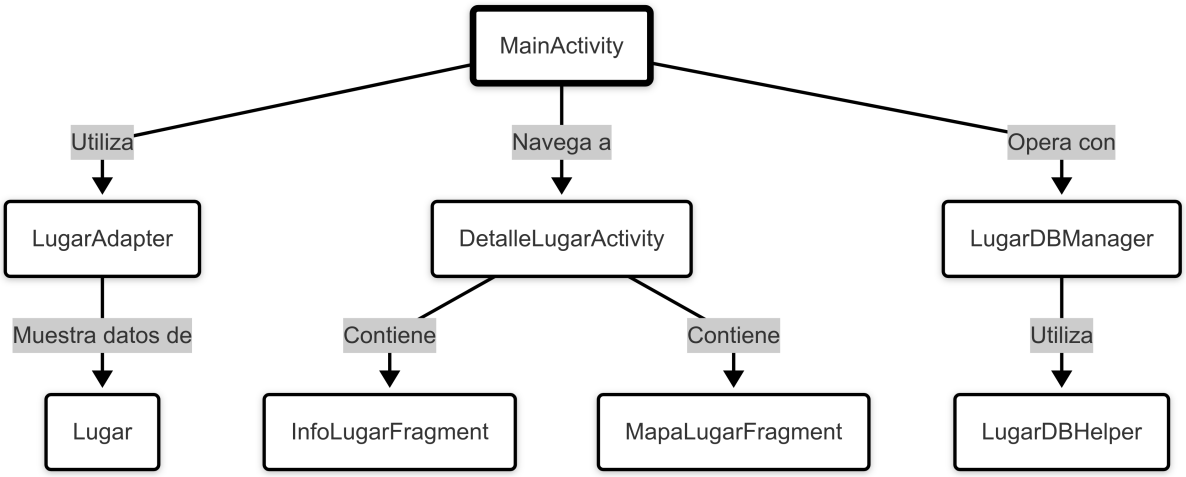

Figura 2: Diagrama de Clases de la Aplicación

4.2.1. Descripción de las Clases Principales

- **MainActivity:** Es la actividad principal que actúa como punto de entrada a la aplicación. En esta clase se inicializa el RecyclerView que muestra la lista de lugares. Además, se configuran elementos de la interfaz como la Toolbar, el botón flotante para agregar nuevos lugares y las opciones de configuración (modo oscuro y cambio de idioma). MainActivity también se encarga de la navegación, redirigiendo al usuario a la pantalla de detalles (`DetalleLugarActivity`) al seleccionar un ítem de la lista, y de coordinar la exportación de datos a un archivo de texto y de la creación y gestión de notificaciones locales. Entre sus métodos se encuentran la inicialización de componentes, la actualización de la lista tras operaciones de borrado, agregación...y la gestión de eventos de usuario.

- **DetalleLugarActivity:** Esta actividad se encarga de mostrar información detallada del lugar seleccionado. Al iniciarse, recibe mediante Intent los datos básicos del lugar (nombre y descripción,latitud, longitud) para mostrar la localización en un mapa. La pantalla se divide en dos secciones mediante el uso de fragments:

  - `InfoLugarFragment`: Muestra información textual detallada, incluyendo el nombre, la descripción y la distancia calculada entre la ubicación del usuario y la del lugar.
  - `MapaLugarFragment`: Incorpora un MapView que visualiza la ubicación geográfica del lugar, permitiendo acciones adicionales como abrir la ubicación en Google Maps o compartirla.

  Esta separación facilita la adaptación de la interfaz a diferentes orientaciones y tamaños de pantalla, mejorando la experiencia del usuario.

- **Lugar:** Es la clase modelo,representa un lugar. Define atributos fundamentales como:
  - `id`: Identificador único del lugar.
  - `nombre`: Nombre del lugar.
  - `descripcion`: Descripción detallada del lugar.

  La clase incluye métodos getters y setters para acceder y modificar estos atributos, permitiendo que otros componentes de la aplicación puedan trabajar de forma estructurada con la información.

- **LugarAdapter:** Es el adaptador que vincula la lista de objetos `Lugar` con el RecyclerView. Este componente se encarga de:
  - Inflar el layout de cada ítem (definido en un archivo XML para el CardView).
  - Asignar los datos del objeto `Lugar` a los elementos de la interfaz (por ejemplo, establecer el nombre y la descripción en los TextViews).
  - Gestionar eventos de interacción, tales como clics simples para abrir la pantalla de detalles, y eventos de clic largo o en botones específicos para activar acciones como editar o eliminar un lugar.

  Esta clase es crucial para lograr una presentación dinámica y personalizada de la lista de lugares.

- **LugarDBHelper & LugarDBManager:** Estas clases se encargan de la gestión de la base de datos local usando SQLite:
  - `LugarDBHelper`: Es una subclase de `SQLiteOpenHelper` y se utiliza para crear, actualizar y definir la estructura de la base de datos. Define la tabla **lugares** con columnas para el `id`, `nombre` y `descripcion`. Este componente se encarga de ejecutar las sentencias SQL necesarias para la creación y actualización de la base de datos.
  - `LugarDBManager`: Es la interfaz principal para realizar operaciones CRUD (Crear, Leer, Actualizar y Eliminar) sobre la base de datos. Esta clase encapsula la lógica para abrir y cerrar la base de datos, insertar nuevos registros, eliminar o modificar los existentes y consultar la información. Gracias a esta abstracción, otras partes de la aplicación pueden interactuar con la base de datos de manera sencilla y centralizada, sin preocuparse por los detalles de la implementación SQL.

4.3. Diagrama de la Base de Datos

La base de datos consta de una única tabla denominada **lugares** en la que se almacenan los diferentes lugares de interés, con la siguiente estructura:

Campo	**Tipo**
id	INTEGER (PRIMARY KEY AUTOINCREMENT)
nombre	TEXT NOT NULL
descripcion	TEXT NOT NULL
latitud	REAL NOT NULL
longitud	REAL NOT NULL

5. Manual de Usuario

La aplicación SkyVisit tiene como objetivo ayudar al usuario a localizar los diferentes lugares de interés de manera sencilla y permitir compartir la ubicación de ese lugar para que un amigo/familiar o conocido también pueda disfrutar de él.

La aplicación cuenta con una serie de lugares de interés de base, se ofrece la opción de seguir añadiendo lugares de interés, modificarlos o eliminarlos(los de base si se eliminan al volver a lanzarse se vuelven a crear para que la app no quede vacía).

La interfaz y uso de las distintas opciones que ofrece la aplicación se ha creado de tal manera que sea lo más intuitiva y sencilla para el usuario. Al abrir la aplicación encontramos los lugares de interés listados, un botón flotante que nos permite añadir más lugares de interés, una ToolBar, con un switch que permite cambiar al modo oscuro y un botón desplegable.

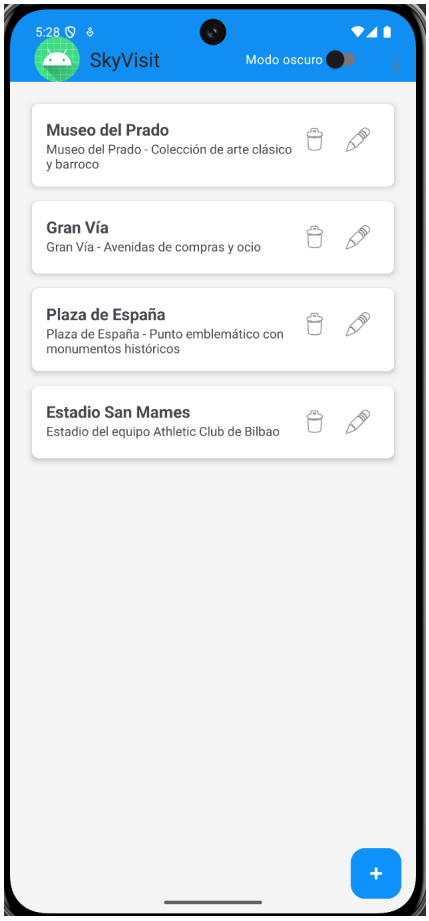

Figura 3: Menú principal

Si le damos al botón desplegable de los 3 puntos, permite cambiar el idioma, exportar los lugares de interés a un txt en la localización deseada y un diálogo de información general de la web.

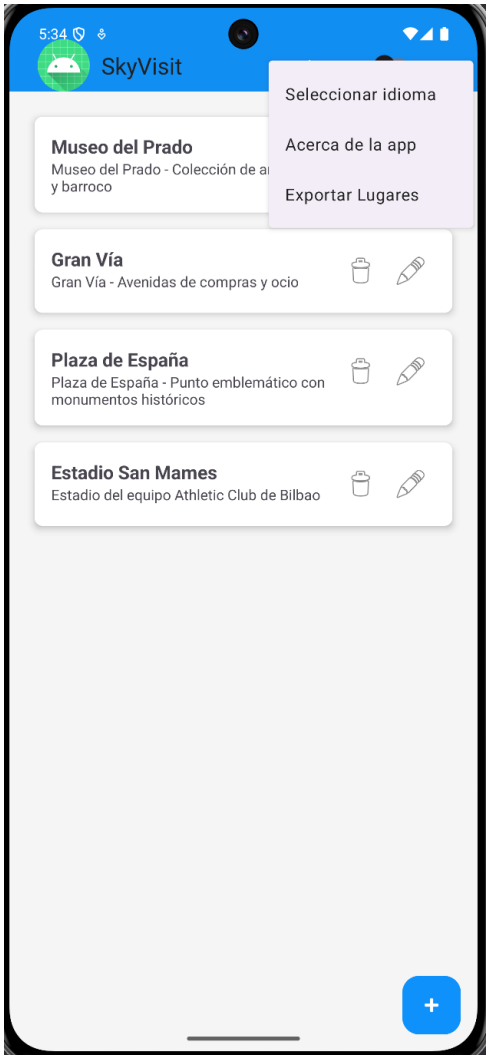

Figura 4: Botón desplegable ToolBar

Si se pincha en seleccionar idioma, sale un diálogo en el que se puede elegir entre inglés o español. Al pulsar cualquiera de estos dos idiomas, la app cambiará completamente de idioma, los lugares y su descripción se mantendrán en el idioma en el que han sido introducidos en la base de datos.

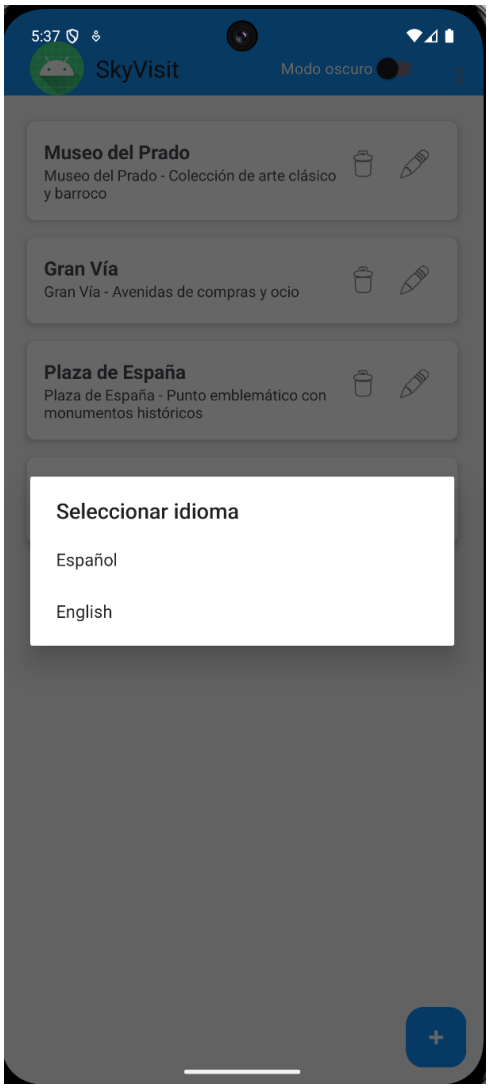

Figura 5: Seleccionar Idioma

Por otra parte, si se pulsa en Acerca de la app saldrá un mensaje informativo.

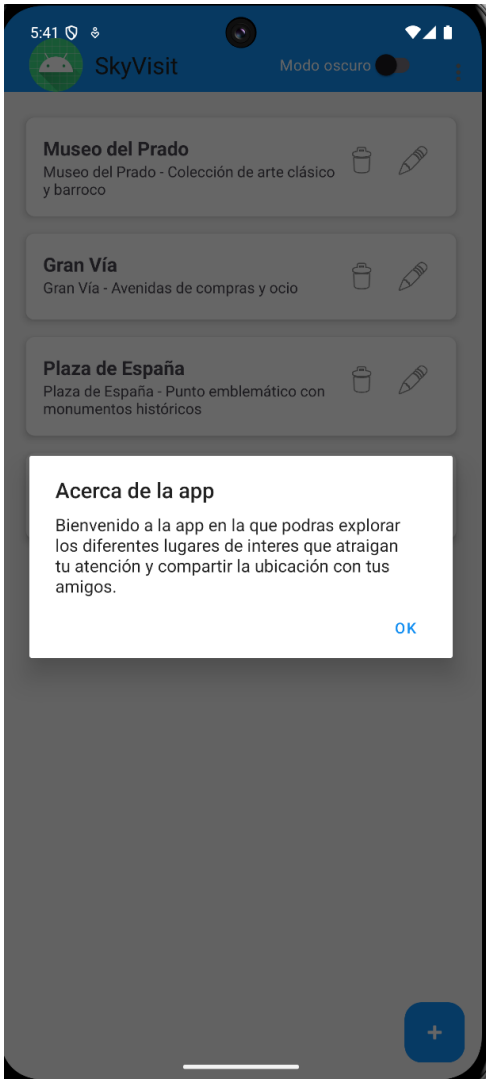

Figura 6: Información de la app

Por ultimo, si se pulsa en Exportar Lugares se abrirá por defecto la localización descargas del dispostivo donde podremos descargar los lugares de interes que tenemos actualmente y su descripción, para ello asignaremos el nombre deseado y pulsaremos "Save".

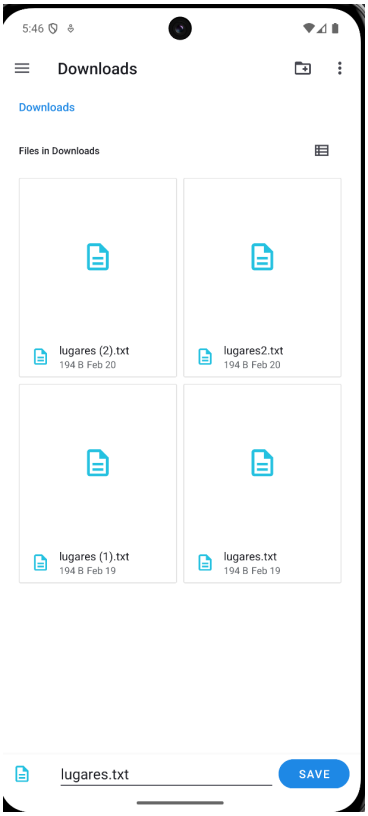

Figura 7: Descarga de lugares de interes

Volviendo a la ToolBar, si pulsamos el switch de modo oscuro, la app tendra el siguiente aspecto

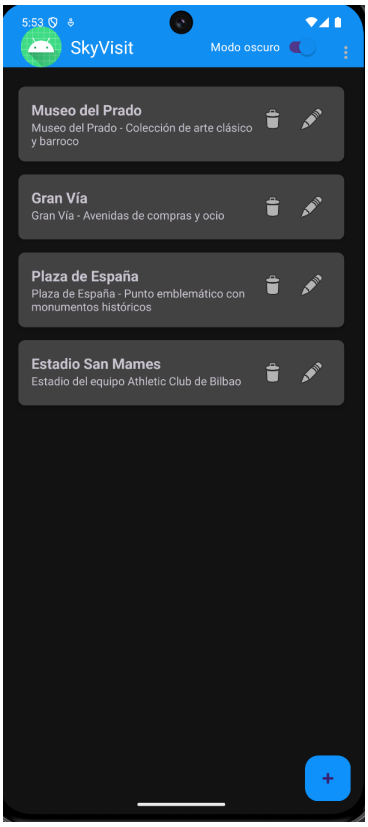

Figura 8: Aspecto modo oscuro

En la lista de lugares si mantenemos pulsado el lugar o pulsamos el icono de la basura aparecera la opción de borrar el lugar, y si se pulsa el lapiz, permitira editar el nombre o la descripción del lugar.

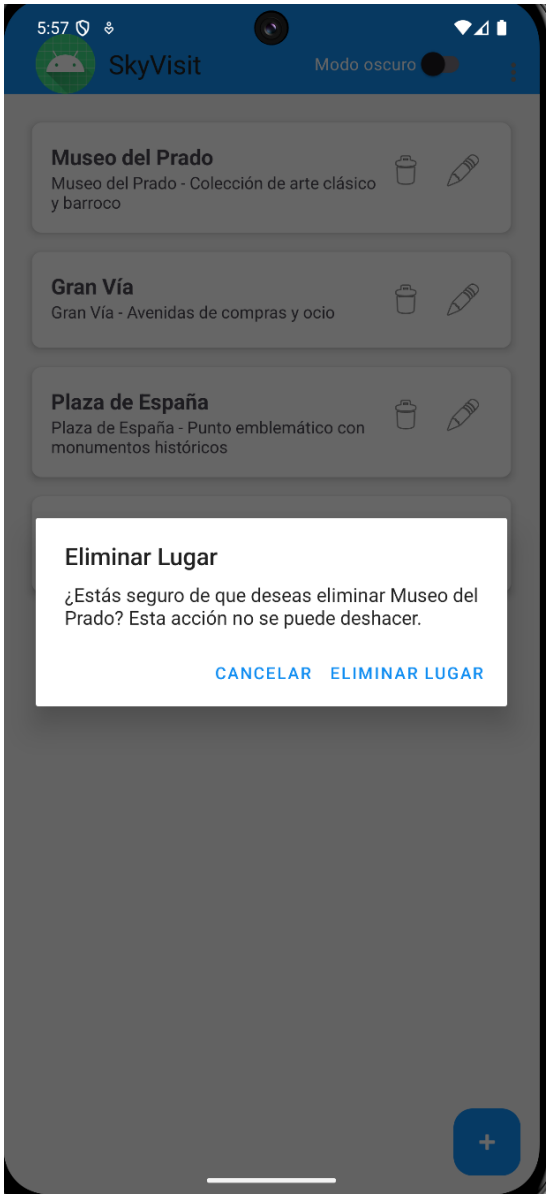

Figura 9: Eliminar Lugar de Interés

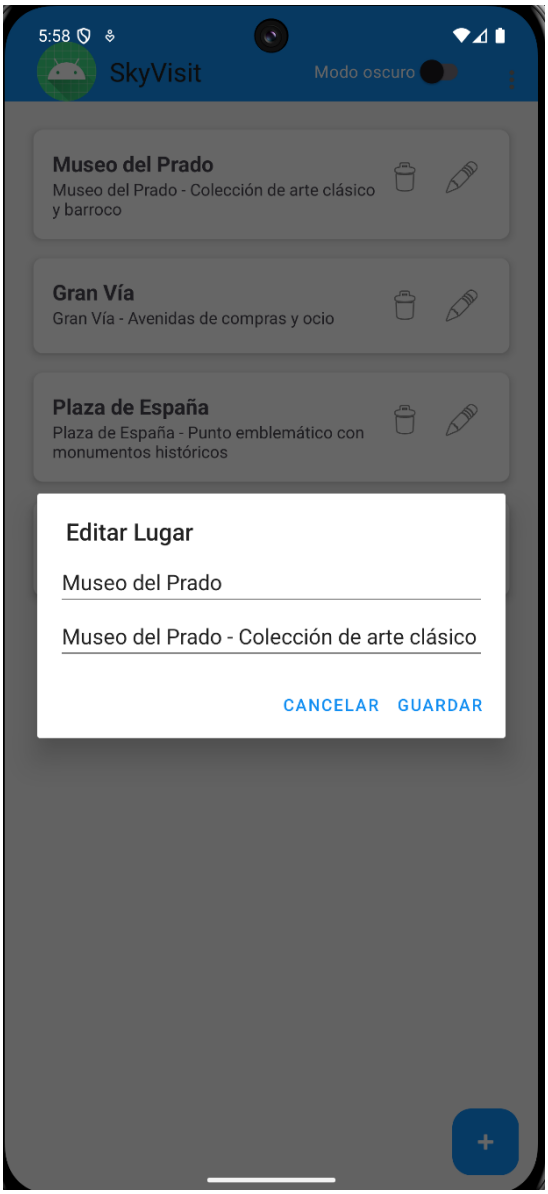

Figura 10: Editar Lugar de Interés

Al pulsar el botón flotante, permite añadir un nuevo lugar de interes, y al añadirlo llega una notificación al movil de dicha acción.

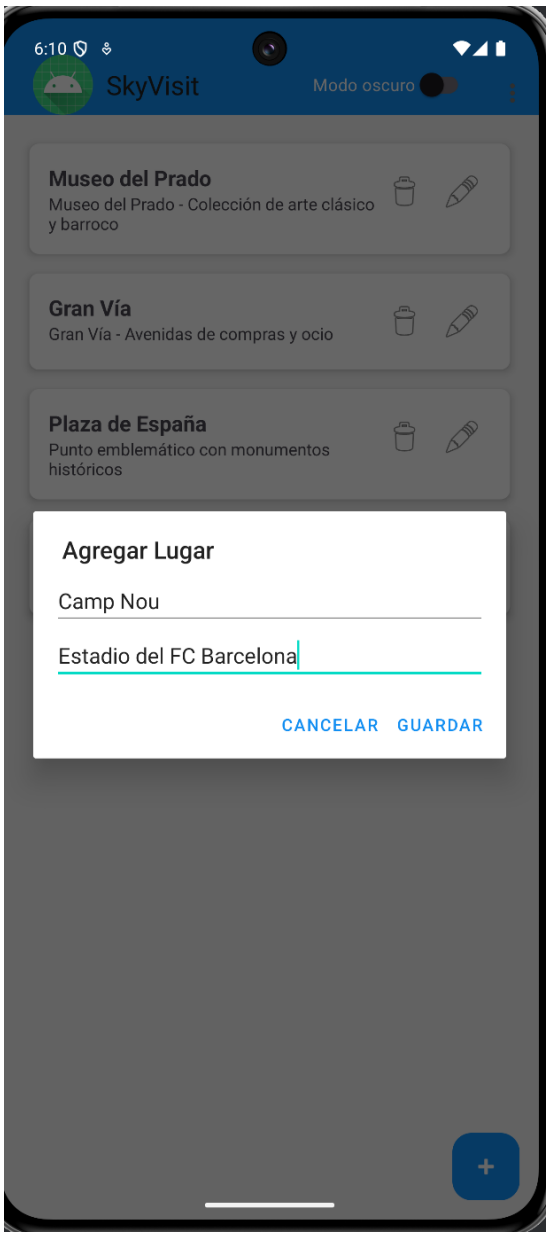

Figura 11: Añadir Lugar de Interes

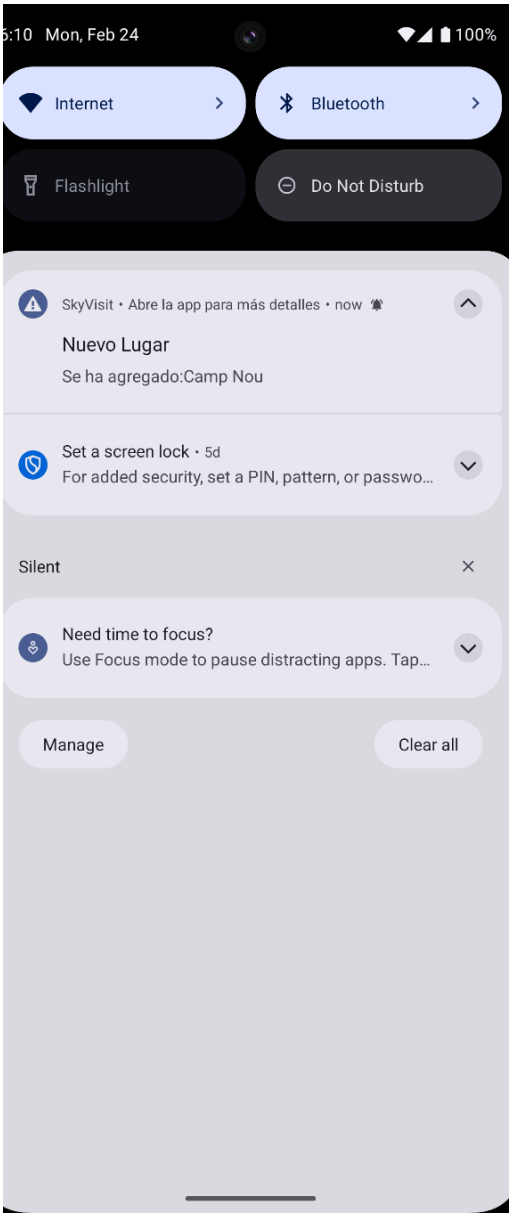

Figura 12: Notificación Local

Por ultimo,al pulsar en un lugar de interes especifico de la lista, se mostraran sus detalles, dependiendo de la horientación del movil de una forma u otra gracias a los fragments. En esta vista se puede acceder al google maps o también compartir la ubicación con quien queramos.

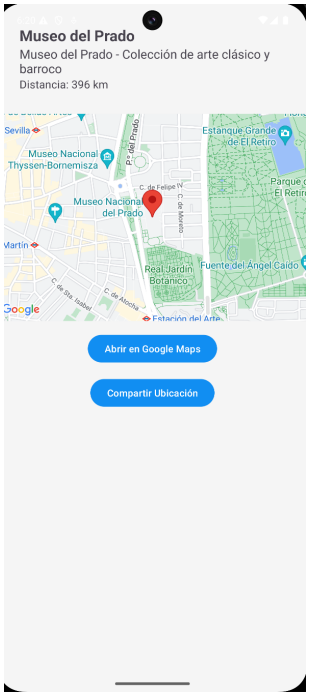

Figura 13: Detalles Lugar Vertical

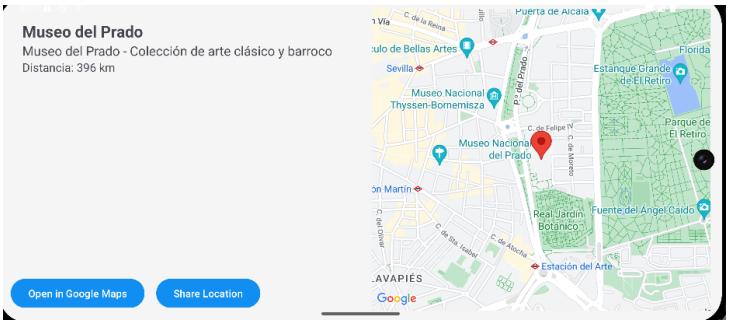

Figura 14: Detalles Lugar Horizontal

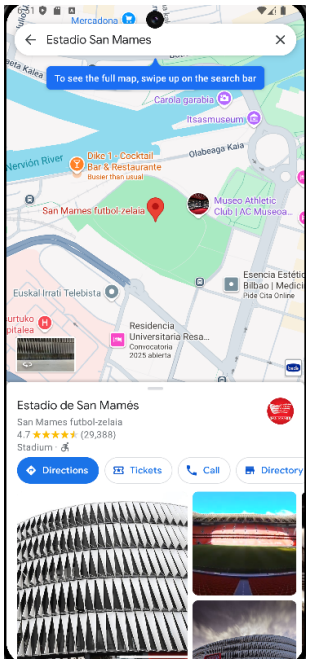

Figura 15: Google Maps

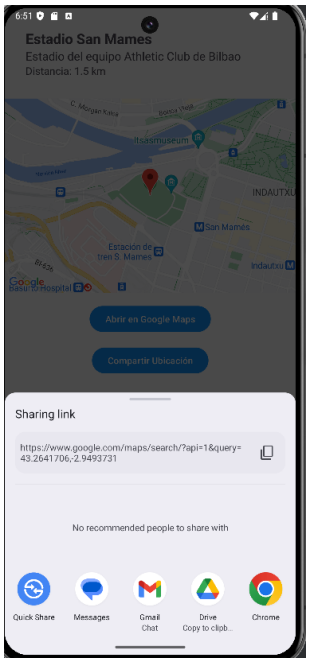

Figura 16: Compartir Ubicación

6. Dificultades Encontradas

Durante el desarrollo del trabajo, una de las principales dificultades fue el uso de Android Studio, ya que era una herramienta nueva para mí. En particular, la obtención de la localización del usuario presentó varios problemas. Al probar en el emulador, aunque modificara la ubicación manualmente, este seguía tomando la localización por defecto de Estados Unidos.

Para verificar el correcto funcionamiento del código, tuve que pedirle a un amigo con un dispositivo Android real que lo probara. Afortunadamente, en su dispositivo funcionó correctamente.

Sin embargo, dado que no sé cómo el profesor va a evaluar esta funcionalidad si en dispositivo real o en emulador, he decidido dejar la ubicación de Bilbao hardcodeada por defecto. No obstante, el método para obtener la localización sigue implementado en el código, aunque comentado.

7. Fuentes Utilizadas

Las siguientes fuentes han sido de gran ayuda durante el desarrollo del proyecto:

- **Documentación Oficial de Android:** [3]
- **Tutoriales y Blogs Especializados:** Recursos en línea sobre RecyclerView, SQLite, Notificaciones y más [2].
- **Foros de Desarrollo:** Stack Overflow, GitHub y otras comunidades que han aportado soluciones a problemas encontrados [4].
- **Egela:** Se han usado los apuntes del contenido visto en clase asi como los guiones de laboratorios y recursos de ayuda para la documentación [1]

8. Conclusiones

El desarrollo de esta aplicación me ha permitido adentrarme de forma práctica en el mundo de Android y la programación de aplicaciones móviles. A lo largo del proyecto, he aprendido a gestionar el ciclo de vida de las actividades, a trabajar con fragmentos y recursos XML, y a manejar bases de datos locales junto con notificaciones. Además, he profundizado en temas avanzados como el cambio de idioma, la exportación de datos y la personalización de la interfaz mediante estilos y temas propios.

Sin duda, este trabajo no solo ha cumplido con los requisitos establecidos, sino que también ha sido fundamental para ampliar mis conocimientos y habilidades en el desarrollo de software móvil, sentando una base sólida para futuros proyectos en este ámbito.

Referencias

[1] Apuntes. https://egela.ehu.eus/course/view.php?id=96452&section=0#tabs-tree-start. Consultado el 23 de febrero de 2025.

[2] Crear un recyclerview + cardview. https://www.youtube.com/watch?v=McOXT58A1Z4&t=529s. Consultado el 23 de febrero de 2025.

[3] Documentación oficial de android. https://developer.android.com/. Consultado el 23 de febrero de 2025.

[4] Stack overflow. https://stackoverflow.com/. Consultado el 23 de febrero de 2025.