Desarrollo Avanzado de Software

Trabajo Final

Titulación:

Grado en Informática de Gestión y Sistemas de Información

4º Curso

Fecha: 19-06-2025

Rocha Astutiani, Eduardo

Contenido

1. Introducción

Unigo es una aplicación móvil Android diseñada para facilitar la movilidad y el acceso al campus de Álava de la UPV/EHU. El objetivo principal del proyecto es ofrecer una solución integral que permita a los usuarios planificar sus rutas a la universidad desde la ubicación actual, incluyendo Bilbao, utilizando diferentes medios de transporte: a pie, en bicicleta y en autobús.

La aplicación cuenta con un sistema de registro e inicio de sesión de usuarios, gestión del perfil personalizado con foto, y una interfaz bilingüe (español e inglés) para maximizar su accesibilidad. El núcleo de la aplicación reside en sus módulos de mapas interactivos, que no solo visualizan rutas, sino que también integran datos de transporte público (urbano e interurbano) y puntos de interés relevantes como carriles bici.

Enlace al repositorio de código Git:

https://github.com/EdrochaA/Trabajo_Final_Unigo.git

2. Elementos de valoración utilizados

La aplicación Unigo cumple y excede los requisitos de desarrollo propuestos, implementando una arquitectura robusta y funcionalidades complejas. A continuación, se detallan los elementos clave:

Características Fundamentales Implementadas:

- **Diseños Personalizados y Componentes de Material Design:** Se ha hecho un uso extensivo de MaterialCardView para crear una interfaz de menú principal moderna y accesible. Se han diseñado "Drawables" personalizados mediante selectores XML para crear botones interactivos con diferentes estados (normal, pulsado).
- **Base de Datos Remota y Arquitectura Cliente-Servidor:** En lugar de una base de datos local, se ha implementado una arquitectura cliente-servidor completa. La aplicación Android (cliente) se comunica mediante la librería Retrofit con una API RESTful desarrollada en PHP.
- **Uso de Diálogos Interactivos:** Se utilizan diálogos (AlertDialog) para mejorar la interacción con el usuario, como en la edición del perfil de usuario, donde se muestra un formulario personalizado, y para la notificación de errores de login o de red.
- **Gestión del Ciclo de Vida y Navegación:** Se ha controlado la pila de actividades de forma precisa. Por ejemplo, al cerrar sesión, se utilizan Intent Flags (FLAG_ACTIVITY_CLEAR_TASK | FLAG_ACTIVITY_NEW_TASK) para limpiar el historial de pantallas y asegurar que el usuario no pueda volver atrás a una pantalla privada.
- **Soporte para Múltiples Idiomas (Internacionalización):** La aplicación es completamente bilingüe (español e inglés). Todo el texto visible por el usuario se gestiona a través de los ficheros de recursos strings.xml, y la selección de idioma se guarda y se carga para ofrecer una experiencia consistente.

Características Adicionales Avanzadas:

- **Integración Avanzada de Mapas (osmdroid):** Se utiliza osmdroid no solo para mostrar mapas, sino para crear una experiencia rica, incluyendo:
    - Dibujo de rutas dinámicas (Polyline) que siguen el trazado de las carreteras.
    - Carga y visualización de capas de datos geográficos desde un fichero GeoJSON para mostrar los carriles bici (bidegorris).
    - Inclusión de múltiples marcadores personalizados para puntos de interés (paradas de bus, parkings, etc.).
- **Planificación de Rutas Multimodales:** El módulo de autobús implementa una lógica compleja que:
    - Carga y gestiona dos fuentes de datos GTFS diferentes (autobuses urbanos e interurbanos).

    - Determina la ubicación del usuario y decide si necesita una ruta local o una ruta con transbordo.
    - Planifica y visualiza el viaje completo, incluyendo los tramos a pie y los diferentes trayectos en autobús.
- **Interacción con Hardware (GPS y Cámara):** La aplicación interactúa con el hardware del dispositivo. Utiliza el FusedLocationProviderClient para obtener la ubicación GPS precisa del usuario y lanza un Intent para usar la cámara y capturar una foto de perfil.
- **Uso de Preferencias Compartidas (SharedPreferences):** Se utilizan para gestionar la sesión del usuario (estado de login, ID, datos personales) y para guardar la preferencia de idioma del usuario entre diferentes usos de la aplicación.
- **Uso de Intents Implícitos:** Se usan para interactuar con otras aplicaciones del sistema, como abrir una página web en el navegador (ACTION_VIEW) o llevar al usuario a los ajustes de ubicación del dispositivo.

Las versiones de herramientas y dependencias utilizadas son las siguientes:

Tecnologías y Dependencias

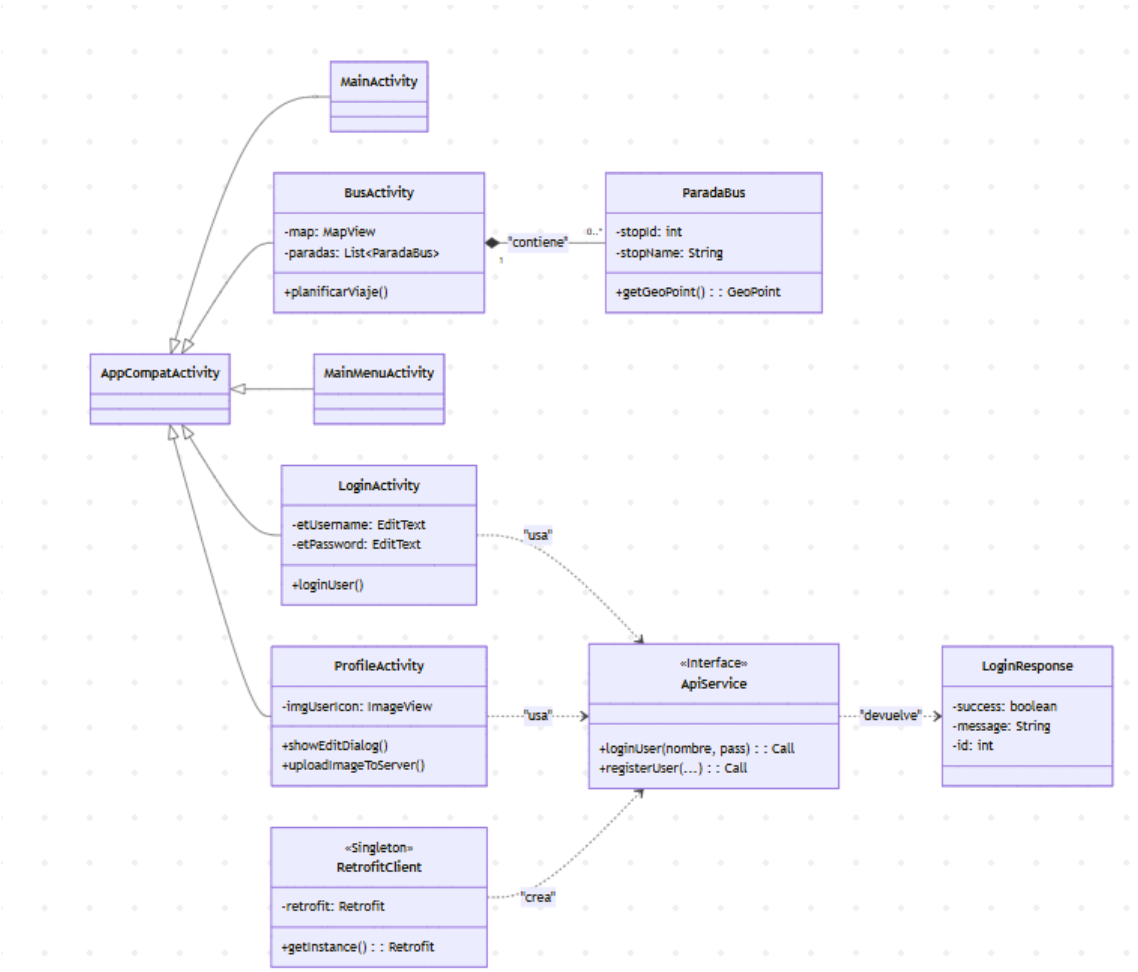
- **SDK de Android:** API Level 33 (Android 13).
- **Dispositivo de Pruebas:** Emulador de Android Studio (Pixel 8).
- **Librerías y Tecnologías Utilizadas:**
    - **Backend:** PHP y MySQL.
    - **Networking:** com.squareup.retrofit2:retrofit y converter-gson para la comunicación con la API.
    - **Mapas:** org.osmdroid:osmdroid-android y org.osmdroid.bonuspack:osmdroid-bonuspack para mapas y rutas.
    - **Localización:** com.google.android.gms:play-services-location para el acceso al GPS.
    - **Carga de Imágenes:** com.github.bumptech.glide:glide para mostrar la foto de perfil.
    - **Componentes de UI:** com.google.android.material:material (para CardView, TextInputEditText, etc.) y androidx.constraintlayout:constraintlayout.
    - **Persistencia Local:** SharedPreferences (nativa de Android).
    - **Componentes Nativos:** AlertDialog, Intents explícitos e implícitos.

3. Descripción de las clases y base de datos

Diagrama de la base de datos:

usuario			
INT	id	PK	Clave Primaria, Autoincremental
VARCHAR(50)	nombre		Único
VARCHAR(100)	email		Único
VARCHAR(255)	contraseña		Hash de la contraseña
VARCHAR(20)	telefono		Opcional
VARCHAR(255)	foto		URL de la imagen de perfil

Diagrama de clases:



Arquitectura del Proyecto

La aplicación Unigo sigue una arquitectura Cliente-Servidor. Esta estructura separa claramente las responsabilidades:

- **El Cliente (Aplicación Android):** Se encarga de toda la lógica de la interfaz de usuario, la presentación de los mapas, la interacción con el hardware del dispositivo (GPS, cámara) y de realizar peticiones de red al servidor.
- **El Servidor (Backend PHP):** Gestiona la lógica de negocio, la autenticación de usuarios y la persistencia de los datos en una base de datos MySQL remota.

Este enfoque modular permite que la lógica de la aplicación y la gestión de datos estén desacopladas, facilitando el mantenimiento y la escalabilidad.

Paquetes y Clases

El código fuente del cliente Android está organizado en los siguientes paquetes principales:

3.1. ui (Interfaz de Usuario) Contiene todas las Activity de la aplicación, que representan cada una de las pantallas visibles para el usuario.

- **MainActivity**:
    - Pantalla de bienvenida y punto de entrada de la aplicación.
    - Gestiona la selección de idioma (español/inglés) y persiste la elección.
    - Ofrece los puntos de acceso al Login y al Registro.
- **LoginActivity**:
    - Permite a los usuarios registrados iniciar sesión con su nombre de usuario y contraseña.
    - Se comunica con el script login_unigo.php a través de Retrofit para validar las credenciales.
    - Al tener éxito, guarda los datos de la sesión en SharedPreferences.
- **RegisterActivity**:
    - Permite a nuevos usuarios crear una cuenta proporcionando nombre, email, teléfono y contraseña.
    - Se comunica con el script register_unigo.php para guardar el nuevo usuario en la base de datos remota.
- **MainMenuActivity**:
    - Actúa como el menú principal tras un login exitoso.
    - Presenta una interfaz con MaterialCardView para navegar a las diferentes funcionalidades (A pie, Bici, Bus, Perfil, etc.).
- **ProfileActivity**:
    - Muestra los datos del usuario que ha iniciado sesión (nombre, email, teléfono y foto).

    - Permite editar estos datos, que se actualizan en la base de datos remota a través de update_profile_unigo.php.
    - Permite al usuario tomar una foto con la cámara y subirla al servidor a través de upload_profile_image_unigo.php.
- **WalkActivity**, **BikeActivity** y **BusActivity**:
    - Son las actividades principales de mapas, basadas en la librería osmdroid.
    - Obtienen la ubicación real del usuario mediante FusedLocationProviderClient.
    - Calculan y dibujan rutas en el mapa utilizando OSRMRoadManager.
    - BikeActivity carga una capa GeoJSON para visualizar los carriles bici.
    - BusActivity implementa la lógica más compleja, cargando dos fuentes de datos GTFS distintas (urbano e interurbano) para planificar rutas con posibles transbordos.
- **UniversityActivity**:
    - Muestra información sobre el campus, incluyendo una lista de grados disponibles en un Spinner.
    - Contiene un Intent implícito para abrir el sitio web de la universidad.
- **TramActivity**:
    - Actualmente es una clase vacía, preparada para una futura implementación de la funcionalidad del tranvía.

3.2. model (Modelos de Datos) Define las clases POJO (Plain Old Java Object) que representan las entidades de datos de la aplicación.

- **ParadaBus**:
    - Representa una parada de autobús.
    - Sus atributos son: stopId (identificador único), stopName (nombre de la parada), y las coordenadas lat y lon.

3.3. network (Gestión de Red) Este paquete es fundamental y contiene todas las clases relacionadas con la comunicación con el backend mediante Retrofit.

- **RetrofitClient**:
    - Clase Singleton que configura y provee una única instancia de Retrofit para toda la aplicación, apuntando a la URL base del servidor.
- **ApiService**:
    - Interfaz de Retrofit que define los endpoints de la API (ej. @POST("login_unigo.php")) y los métodos Java para realizar las peticiones de red.
- **LoginResponse**, **RegisterResponse**, **GenericResponse**:
    - Clases que modelan las respuestas JSON del servidor.

3.4. Backend (Scripts PHP) Aunque no forman parte del proyecto Android, son una pieza esencial de la arquitectura global.

- **db_config_unigo.php**: Almacena las credenciales de conexión a la base de datos MySQL.
- **login_unigo.php**: Valida las credenciales del usuario contra la base de datos y devuelve sus datos si son correctas.
- **register_unigo.php**: Inserta un nuevo usuario en la tabla usuario, hasheando la contraseña.
- **update_profile_unigo.php**: Actualiza los datos de un usuario existente.
- **upload_profile_image_unigo.php**: Recibe una imagen en Base64, la decodifica, la guarda en el servidor y actualiza la URL en la base de datos.

Base de Datos (MySQL Remota)

La aplicación no utiliza una base de datos local como Room. Toda la información persistente de los usuarios se almacena en una base de datos MySQL alojada en el mismo servidor que el backend PHP. La entidad principal es:

- **Tabla usuario**:
    - Almacena la información de cada cuenta de usuario.
    - Columnas:
        - id: INT, Clave Primaria, Autoincremental.
        - nombre: VARCHAR, nombre único del usuario.
        - email: VARCHAR.
        - telefono: VARCHAR.
        - contraseña: VARCHAR, almacena el hash de la contraseña.
        - foto: VARCHAR, almacena la URL completa de la imagen de perfil.

4. Manual de usuario

Pantalla principal.

Inicio (MainActivity):

- Permite acceder al inicio de sesión o registro de usuario.
- Se puede elegir idioma entre castellano o inglés.

Inicio de sesión (LoginActivity):

- Inicio de sesión con nombre de usuario y contraseña.
- Opción de poder acceder al registro o volver a la interfaz inicial.

Registro (RegisterActivity):

- Registro con nombre, contraseña, correo y teléfono.
- Redirige al inicio de sesión una vez registrado. Permite volver a la interfaz inicial o al inicio de sesión.

Menú principal (MainMenuActivity):

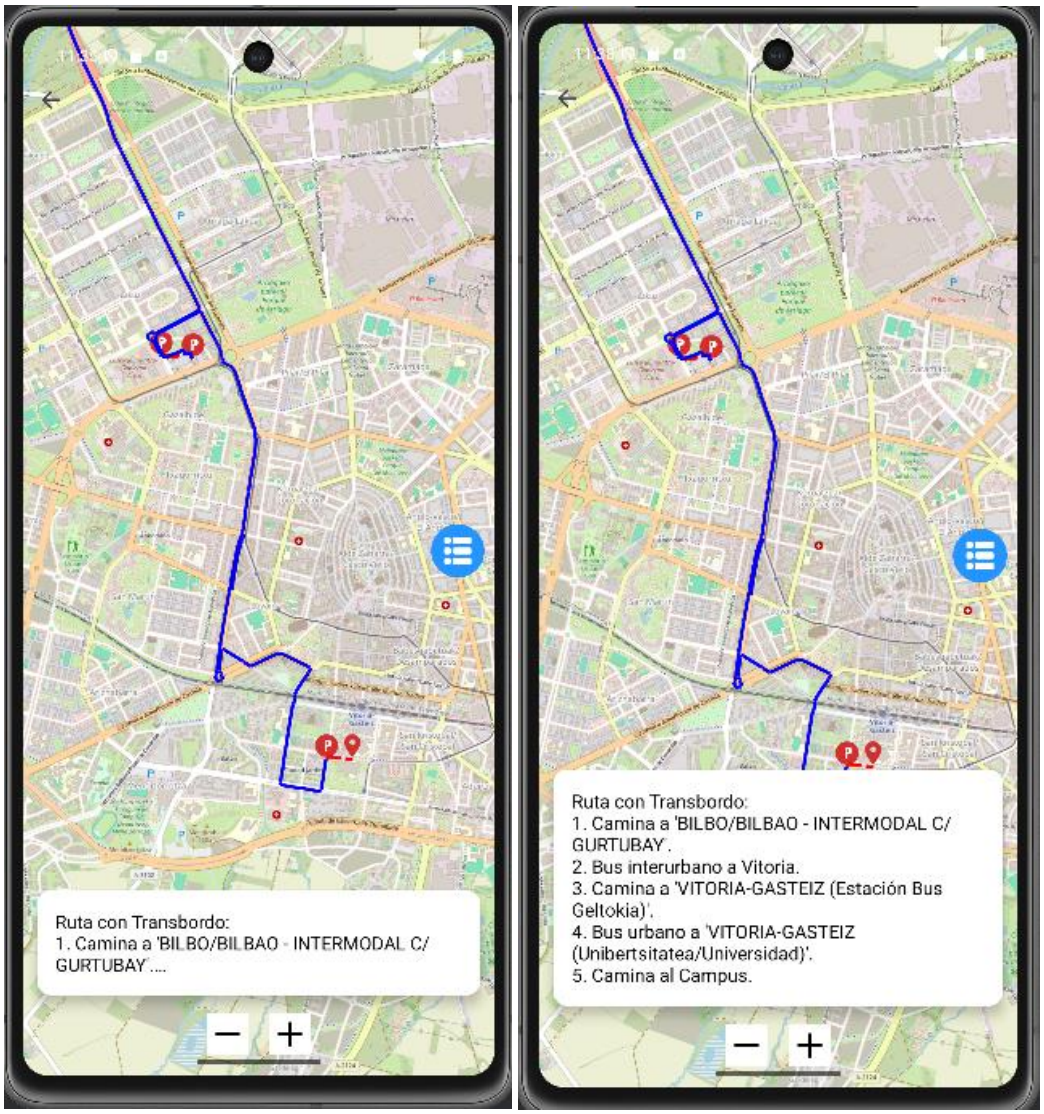
- Opciones de llegada al campus de Álava tanto en bici, bus o a pie. El tranvía está en proceso de desarrollo.
- Opción de visualizar los datos del perfil de usuario, modificarlos y subir una foto.
- Opción de visualización de los grados disponibles en el campus.

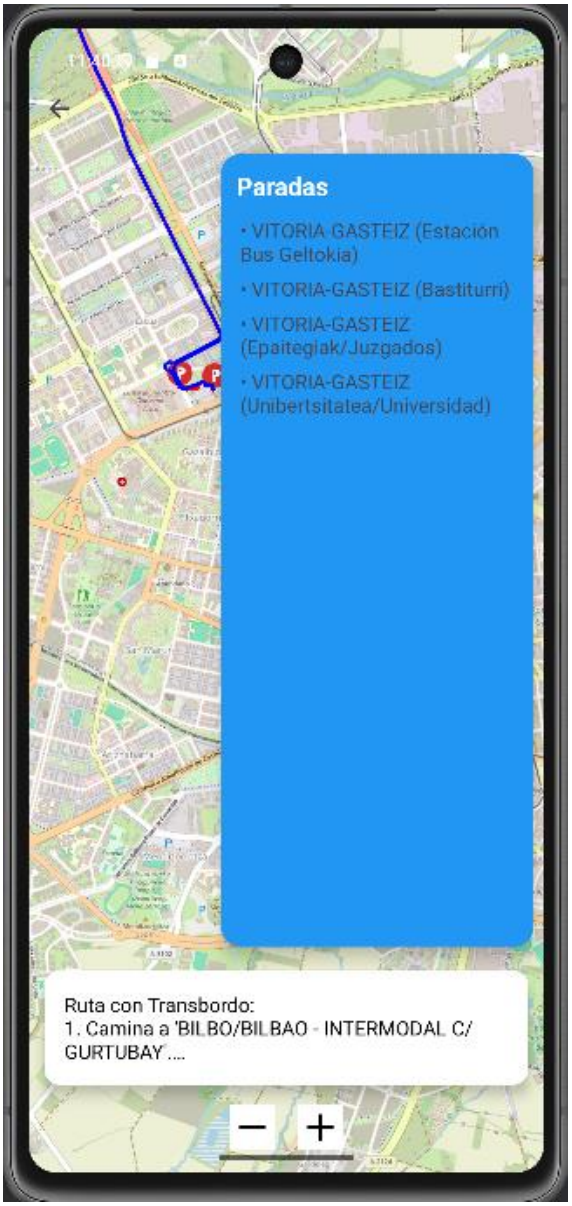
Mapa de recorrido en bicicleta (BikeActivity):

- Observamos la distancia de recorrido hasta el campus.
- Se dibujan todos los carriles disponibles.

Mapa de recorrido en autobús (BusActivity):

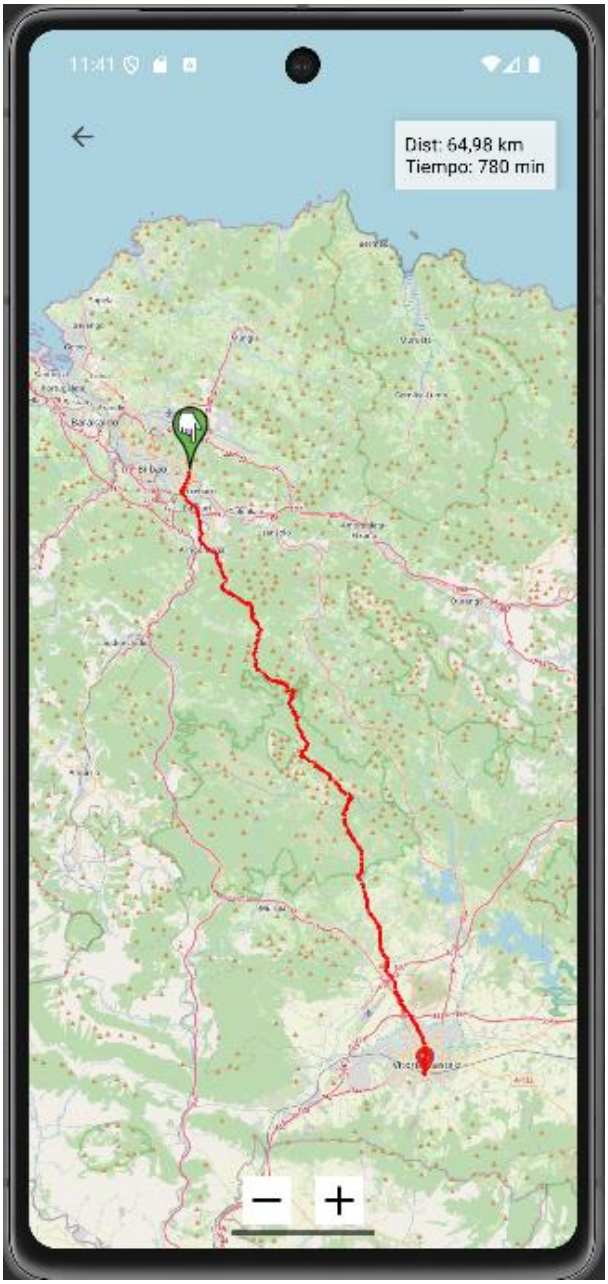
- Observamos la distancia de recorrido hasta el campus y las paradas de buses.
- Se En la barra lateral vemos las paradas de bus y si hacemos click en la información de abajo, se desplega los pasos a seguir para llegar.




Paradas
• VITORIA GASTEIZ (Estación Bus Geltokia)
• VITORIA-GASTEIZ (Bastiturri)
• VITORIA-GASTEIZ (Epaitegiak/Juzgados)
• VITORIA-GASTEIZ (Unibertsitatea/Universidad)
Ruta con Transbordo:
1. Camina a 'BILBO/BILBAO - INTERMODAL C/ GURTUBAY'....

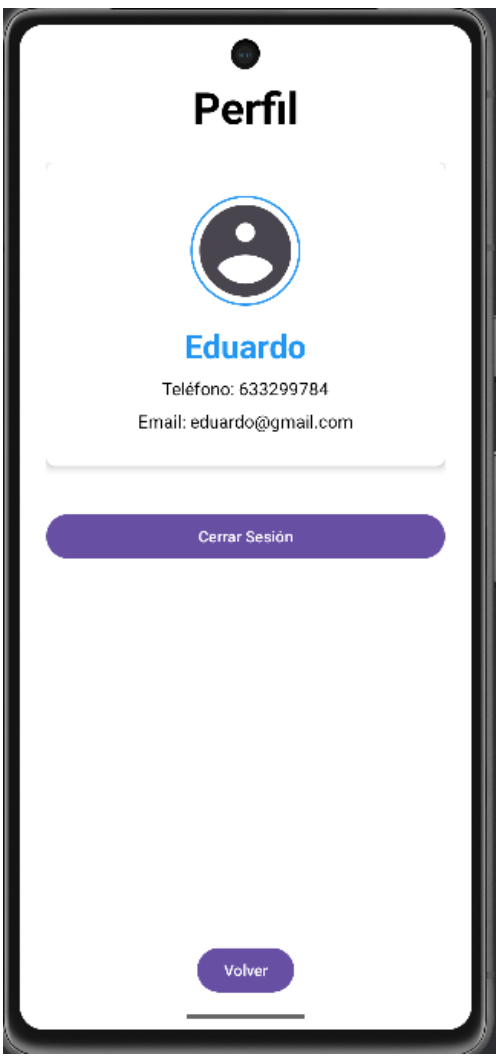
Mapa de recorrido a pie (WalkActivity):

- Observamos la distancia de recorrido hasta el campus y el tiempo que tardaría.



Perfil de usuario (ProfileActivity):

- Vemos la información del usuario.
- Opción de volver al menú principal o cerrar sesión, que nos llevaría al inicio de sesión.
- Al hacer click en los datos se abre un desplegable donde podremos modificar los datos de la base de datos.



Información de los grados (UniversityActivity):

- Desplegable con los grados.
- Enlace a la página oficial de la UPV/EHU del campus de Álava.
- Opción de volver al menú principal.

4. Dificultades encontradas

Durante el desarrollo de Unigo, surgieron varios desafíos técnicos que requirieron una depuración exhaustiva y una adaptación de la planificación inicial. A continuación, se detallan los principales problemas encontrados:

4.1. Implementación de la Ruta de Autobús Multimodal

La funcionalidad más compleja y desafiante fue la planificación de rutas en autobús, que debía combinar datos de diferentes redes de transporte.

- **Integración de Múltiples Fuentes de Datos:** Inicialmente, se planeó usar una única fuente de datos de transporte. Sin embargo, para ofrecer rutas desde Bilbao hasta el campus de Álava, fue necesario integrar dos conjuntos de datos GTFS distintos: los autobuses urbanos de Vitoria-Gasteiz y los interurbanos de La Unión. Estos ficheros tenían estructuras de columnas diferentes, lo que provocó errores de parsing (NumberFormatException) y requirió la creación de lógicas de lectura de datos separadas y robustas para cada fichero.
- **Lógica de Transbordo:** El plan original de "encontrar la parada más cercana" resultó ser insuficiente, ya que la parada más próxima al usuario no siempre pertenecía a una línea que conectara con el destino final. Fue necesario desarrollar un algoritmo más inteligente que, al detectar que el usuario está fuera de Vitoria, planifica la ruta asumiendo un transbordo en un punto clave (la estación de autobuses de Vitoria).
- **Formato de Ficheros (CSV):** Uno de los ficheros de paradas (stopsUnion.txt) contenía comas dentro de los nombres de las paradas, lo que rompía la lógica de división de cadenas (split). Fue necesario implementar un método de parsing más avanzado con expresiones regulares para leer correctamente estas líneas.

4.2. Internacionalización y Contenido Dinámico

Aunque la aplicación se diseñó para ser bilingüe, la traducción de los textos generados dinámicamente desde el código Java presentó un desafío.

- **Textos en Toast y TextView:** Inicialmente, solo se tradujeron los textos de los layouts XML. Sin embargo, todos los mensajes de notificación (Toast) y la información de las rutas que se generan en tiempo de ejecución (ej. "Distancia: 5.2 km") estaban "hardcodeados" en español en los ficheros Java.

- **Solución:** Fue un trabajo metódico externalizar todas estas cadenas de texto a los ficheros strings.xml, utilizando placeholders de formato (ej. %1$.2f) para insertar los valores numéricos dinámicos. Esto se aplicó en ProfileActivity, WalkActivity, BikeActivity y BusActivity.

4.3. Experiencia de Usuario en los Mapas

Mejorar la experiencia visual y de interacción en las pantallas de mapas requirió varios ajustes no previstos.

- **Pantalla de Carga del Mapa:** La librería osmdroid mostraba una pantalla de carga azul con un globo terráqueo por defecto mientras se descargaban los datos del mapa. Para ofrecer una experiencia más profesional, se implementó un ProgressBar (rueda de carga) que se muestra al usuario durante este proceso, ocultando el fondo por defecto.
- **Visualización de Rutas:** Las rutas de autobús se dibujaban inicialmente como líneas rectas entre paradas. Se mejoró la lógica para que también estas rutas consultaran el servidor de OSRMRoadManager y siguieran el trazado real de las carreteras, ofreciendo una visualización mucho más realista.
- **Ajustes de Layout:** Al implementar un modo de "pantalla completa" (edge-to-edge), los botones y textos en la parte superior de la pantalla quedaban ocultos por la barra de estado del sistema. Fue necesario reajustar los layouts, utilizando contenedores y la propiedad fitsSystemWindows para posicionar los controles correctamente sin perder el efecto de mapa a pantalla completa.

5. Bibliografía

Desarrollo General de Android

- **Guías Oficiales de Desarrollo de Android (Actividades, Intents, Vistas, etc.)**
  - https://developer.android.com/guide
- **Uso de Intents para la Navegación y Comunicación entre Apps**
  - https://developer.android.com/training/basics/intents
- **Gestión de Preferencias con SharedPreferences**
  - https://developer.android.com/training/data-storage/shared-preferences

Interfaz de Usuario (UI/UX)

- **Guías de Diseño de Material Design**
  - https://material.io/design
- **Temas y Estilos en Android (themes.xml, styles.xml)**
  - https://developer.android.com/guide/topics/ui/look-and-feel/themes
- **Uso de Diálogos con AlertDialog**
  - https://developer.android.com/develop/ui/views/components/dialogs
- **Notificaciones emergentes con Toast**
  - https://developer.android.com/guide/topics/ui/notifiers/toasts
- **Componente CardView para Contenedores**
  - https://developer.android.com/develop/ui/views/layout/cardview

Mapas y Localización

- **Librería de Mapas osmdroid (Alternativa a Google Maps)**
  - https://github.com/osmdroid/osmdroid
- **Librería osmdroid-bonuspack (Para Rutas y Puntos de Interés)**
  - https://github.com/MKergall/osmbonuspack
- **Obtención de la Ubicación del Dispositivo (FusedLocationProviderClient)**

- https://developer.android.com/training/location/retrieve-current

Comunicación de Red y Datos

- **Cliente HTTP con Retrofit (Para consumir la API PHP)**
  - https://square.github.io/retrofit/
- **Conversión de JSON con Gson**
  - https://github.com/google/gson
- **Carga de Imágenes desde URL con Glide**
  - https://github.com/bumptech/glide
- **Especificación del Formato de Datos de Transporte (GTFS)**
  - https://gtfs.org/
- **Parseo de GeoJSON para Capas de Mapa**
  - https://geojson.org/

Backend y Seguridad

- **Documentación Oficial de PHP**
  - https://www.php.net/manual/es/
- **Documentación Oficial de MySQL**
  - https://dev.mysql.com/doc/
- **Cifrado Seguro de Contraseñas en PHP (password_hash)**
  - https://www.php.net/manual/es/function.password-hash.php