Harmony

Design Document

Ali Doğan Çamur 231101017

Savaş Solak 231101025

Ege Azca 231101057

Yankı Yağız Ozan 231101059

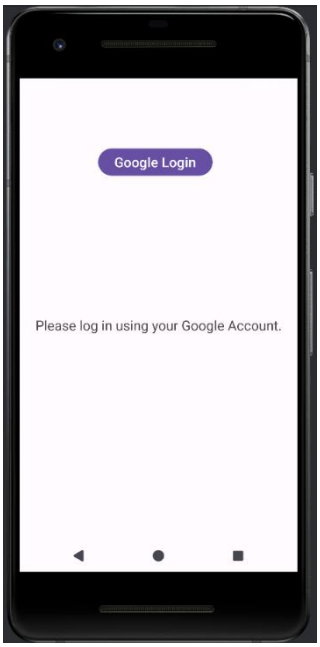
1- System Overview:

- **Brief Project Description:** An app that matches you based on your music taste; it lets you discover the songs playing around you or what your friends are listening to and interact with people who share a similar taste in music.
- **System architecture:** We will use layered approach for our project. The first layer (User Interface) will consist of login screen and main interface. The second one (Business Logic Layer) will consist of background tasks like displaying other users and social interactions. And lastly, the last layer (Data Access Layer) will be the one where is users' datas like messages and other important information's are stored.
- **Technology Stack:** As a development environment we preferred Android Studio and will use Java and Kotlin as the main programming languages. For front end, we are going to use Android Studio's built in UI framework. For database we preferred Firebase. Also, we will use API services like Spotify to track what users are listening to. And lastly, for backend we will use NodeJS.

2- Implementation Details:

- **Codebase Structure:** We will have 5 base modules on client. Those are: UI Module, Network Module, Database Module, Main Module, API Module; also, on backend side we will have those: Sign in/out Module, Data Processing Module, Chat Module.
- **Key Implementations:**
    - **Google Maps Integration:**
        - Uses GoogleMap to display a map.
        - FusedLocationProviderClient to get users current location
        - Requests permissions
    - **User Login with Google Sign-in:**
        - Implements Google Sign-in using GoogleSignInClient
        - Requests an ID for authentication.
        - Redirects to user if successful.
- **Component Interfaces:**
    - **MainActivity:**
        - void onMapReady(): Initializes the map.
        - void getCurrentLocation()
        - void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults): Handles location permission results.
    - **LoginActivity:**
        - void signIn(): Initiates Google Sign-In.
        - void onActivityResult(): Handles the sign-in result.
        - void handleSignInResult(): Processes the authentication result.

- **Visual Interfaces:**



3- Use Case Support in Design:

- User should be able to login using their account information.
- Map functionality should be working.
- It should connect with platforms like Spotify.
- The app should be able to detect and identify songs playing around the user.

4- Design Decisions:

Firebase is great for real-time updates and easy Android integration but lacks advanced queries, while PostgreSQL offers powerful querying and structured storage but requires more setup. For the backend, Node.js handles real-time apps well with non-blocking I/O, making it ideal for Spotify tracking, whereas Django is better for structured data but less suited for real-time performance. Given your needs, Firebase and Node.js are the best fit for real-time interactions, while PostgreSQL and Django excel in structured data management.

Firebase was chosen for real-time sync and easy Android integration. PostgreSQL was considered but requires more setup. Node.js is ideal for real-time APIs and high concurrency. Django is robust but less suited for real-time operations. The client-server architecture ensures scalability and smooth performance.