**Music App Documentation**

**Introduction**

The Music App is a platform designed to provide users with seamless access to music streaming, playlists, and personalized recommendations. It offers a user-friendly interface with features like search, favorites, and offline playback.

**Features**

* **User Authentication**: Sign-up and login functionality with secure authentication.
* **Music Library**: Access to a vast collection of songs across various genres.
* **Search & Discovery**: Users can search for songs, artists, and albums.
* **Playlists**: Create, edit, and manage custom playlists.
* **Favorites**: Users can like and save their favorite songs.
* **Offline Mode**: Download songs for offline listening.
* **Streaming**: High-quality music streaming.
* **Recommendations**: AI-based music suggestions based on listening history.
* **Social Sharing**: Share playlists and songs with friends.

**Technology Stack**

* **Frontend**: React Native / Flutter (for cross-platform compatibility)
* **Backend**: Node.js / Django (for handling API requests and authentication)
* **Database**: Firebase / PostgreSQL (for storing user data and music details)
* **Streaming Service**: AWS S3 / Cloudinary (for hosting music files)
* **Authentication**: Firebase Auth / OAuth
* **AI Recommendation Engine**: TensorFlow / Scikit-learn

**Architecture**

1. **Client-Side**: Handles UI/UX interactions, API calls, and caching.
2. **Server-Side**: Processes user requests, manages authentication, and handles music streaming.
3. **Database**: Stores user profiles, playlists, and song metadata.
4. **Third-Party Integrations**: Uses APIs for music licensing and streaming.

**API Endpoints**

**User Authentication**

* POST /register - Register a new user
* POST /login - Authenticate a user
* GET /profile - Retrieve user profile details

**Music Management**

* GET /songs - Fetch available songs
* GET /songs/{id} - Get details of a specific song
* POST /playlist - Create a new playlist
* PUT /playlist/{id} - Update a playlist
* DELETE /playlist/{id} - Remove a playlist

**Streaming**

* GET /stream/{song\_id} - Stream a specific song
* GET /download/{song\_id} - Download a song for offline mode

**User Interface Design**

The app consists of:

* **Home Screen**: Displays trending and recommended songs.
* **Search Screen**: Allows users to look for specific songs, artists, or albums.
* **Player Screen**: Features playback controls, lyrics, and song details.
* **Library Screen**: Stores playlists, favorites, and downloaded songs.

**Future Enhancements**

* Integration with voice assistants
* Lyrics synchronization
* Live streaming of concerts
* Collaborative playlists

**Conclusion**

The Music App provides an immersive music experience, catering to diverse user preferences with a robust feature set and scalable architecture.