Alpha Player

A Modern Web-Based Music Streaming Platform

1. Introduction

Alpha Player is a full-stack web application designed to provide users with a seamless music discovery and playback experience. The project serves as a digital jukebox where users can browse a curated list of songs, search for specific tracks, and manage their experience through a dedicated account system. The platform is built with a focus on a "glass-morphic" aesthetic, ensuring a modern user interface (UI) that is both visually appealing and functionally robust.

Core Features:

- **User Authentication**: Secure registration and login systems integrated with a SQL database.
- **Dynamic Music Library**: Automated population of song metadata via the Last.fm API and playable audio/covers via the Deezer API.
- **Persistent Media Player**: A fixed-position playback interface allowing users to browse while listening.
- **Administrator Interface**: A dedicated page for managing the product catalog (add/remove/modify functionality).

2. Motivation of Technologies Used

The technology stack was chosen to balance performance, scalability, and ease of deployment.

2.1 Backend: PHP & MySQL

- **PHP**: Chosen for its native compatibility with web servers and robust handling of server-side logic, such as session management and API communication.
- **MySQL**: Utilized for relational data storage. It reliably handles user credentials (user_data.sql) and song metadata (songs.sql), ensuring data integrity and quick search queries.

2.2 Frontend: HTML5, CSS3, and Vanilla JavaScript

- **Vanilla JS**: Instead of heavy frameworks like React, the project uses native JavaScript (ES6+). This ensures fast load times and demonstrates a core understanding of the Document Object Model (DOM) and asynchronous fetch requests.
- **CSS Grid & Flexbox**: Used to create a responsive, modern layout that adapts to different screen sizes, specifically handling the fixed-side player and scrollable music list.

2.3 DevOps: Docker & Docker Compose

- **Containerization**: By using Dockerfile and docker-compose.yml, the development environment is standardized. This eliminates "it works on my machine" issues and allows for easy deployment of the PHP app, MySQL database, and phpMyAdmin simultaneously.

3. Development Decisions

3.1 Dual-API Integration Strategy

One of the key technical decisions was the "hybrid" data approach found in loader.php.

- **Decision**: Use **Last.fm** for broad metadata search and **Deezer** for high-quality .mp3 previews and medium-sized album covers.
- **Motivation**: Last.fm provides superior search results, while Deezer offers more consistent access to playable audio files. Combining them ensures a high-quality library without manual data entry.

3.2 User Experience (UX) Enhancements

- **Fixed Navigation & Player**: The header and player remain visible at all times. This decision was made so that users never lose control of their music while scrolling through long lists of tracks.
- **Smooth Navigation**: The implementation of window.scrollTo({ top: 0, behavior: 'smooth' }) when selecting a song ensures the user is immediately focused on the active track's artwork and title.

3.3 Security & Session Management

- **Password Hashing**: User passwords are never stored in plain text. The project utilizes password_hash with BCRYPT in bridge.php to ensure that even if the database is compromised, user accounts remain secure.
- **State Management**: PHP Sessions are used to track the "logged-in" state, allowing for personalized links to the user's account page via the web.png profile icon.

4. Conclusion

Alpha Player successfully integrates a backend-oriented architecture with a modern frontend. The use of Docker ensures that the project meets professional deployment standards, while the integration of external APIs provides a dynamic experience that goes beyond a static database.