Frontend Development with React.js Project Documentation for Rhythmic Tunes

1. **Introduction**
   * **Project Title : RYTHMIC TUNES**
   * **Team Members**
   *  **P G Dharshini**  [Email Id: [cs2201111058239@lngovernmentcollege.com](mailto:cs2201111058239@lngovernmentcollege.com)]

 **J Anjali** [Email Id: [cs2201111058239@lngovernmentcollege.com](mailto:cs2201111058239@lngovernmentcollege.com)]

 **S Pinky kumari** [Email Id: [cs2201111058239@lngovernmentcollege.com](mailto:cs2201111058239@lngovernmentcollege.com)]

 **S Yuvaraj** [Email Id: [cs2201111058239@lngovernmentcollege.com](mailto:cs2201111058239@lngovernmentcollege.com)]

# Project Overview

* + **Purpose**:

The React Web Audio Player is a dynamic and feature-rich music player application built with React. It allows users to listen to a collection of 19 songs while offering various features that enhance the user experience. The app is designed with both functionality and customization in mind, providing support for auto-play, volume control, theme customization, and more. The app also supports a responsive design, ensuring that it functions smoothly on any device.

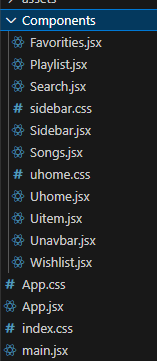
# Features:

* Music collections
* Auto -Play
* Current Time Indicator
* Media volume Control
* Responsive design
* Favroutes
* Music Playlist

# Architecture:

* + **Component Structure**:

The application is built using React.js with a component-based architecture. Major components include:

* **Header**: Contains the navigation bar and search bar.
  + - **Player**: Music player controls (play, pause, volume, etc.).
    - **HomePage**: Displays featured tracks, recommended playlists, and new releases.
    - **SearchPage**: Allows users to search for songs, albums, and artists.
    - **PlaylistPage**: Displays user-created playlists and allows playlist management.

# State Management:

The application uses **Redux** for global state management. The Redux store manages user authentication, current playing track, playlist data, and search results.

# Routing:

The application uses **React Router** for navigation. Routes include:

* + - /: Home page
    - /search: Search page
    - /playlist/:id: Playlist details page
    - /login: User login page

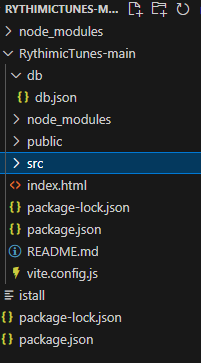
# SetupInstrutions:

* + **Prerequisites**:
    - Node.js (v16 or higher)
    - npm (v8 or higher)
    - Git

# Installation:

1. Clone the repository: git clone [https://github.com/unm12912137/rhythmic-](https://github.com/unm12912137/rhythmic-tunes.git) [tunes.git](https://github.com/unm12912137/rhythmic-tunes.git)
2. Navigate to the client directory: cd rhythmic-tunes/client
3. Install dependencies: npm install
4. Configure environment variables: Create a .env file in the client directory and add the necessary variables (e.g., API keys).
5. Start the development server: npm start

# Folder Structure:



# Running the Frontend:

* To start the frontend server, run the following command in the client directory: npm start
* npm install
* npx json-server ./db/db.json
* npm run dev
* The application will be available at [http://localhost:3000](http://localhost:3000/)

# Component Documentation

* + **Key Components**:
    - **Header**: Displays the navigation bar and search bar.
      * Props: onSearch (function to handle search queries).
    - **Player**: Controls the music playback.
      * Props: currentTrack (object containing track details), onPlay, onPause, onSkip.
    - **PlaylistCard**: Displays a playlist with its name and cover image.
      * Props: playlist (object containing playlist details), onClick (function to handle playlist selection).

# Reusable Components:

* + - **Button**: A customizable button component.
      * Props: text, onClick, disabled.
    - **Input**: A reusable input field for forms and search.
      * Props: type, placeholder, value, onChange.

# State Management

* + **Global State**:

The Redux store manages the following global states:

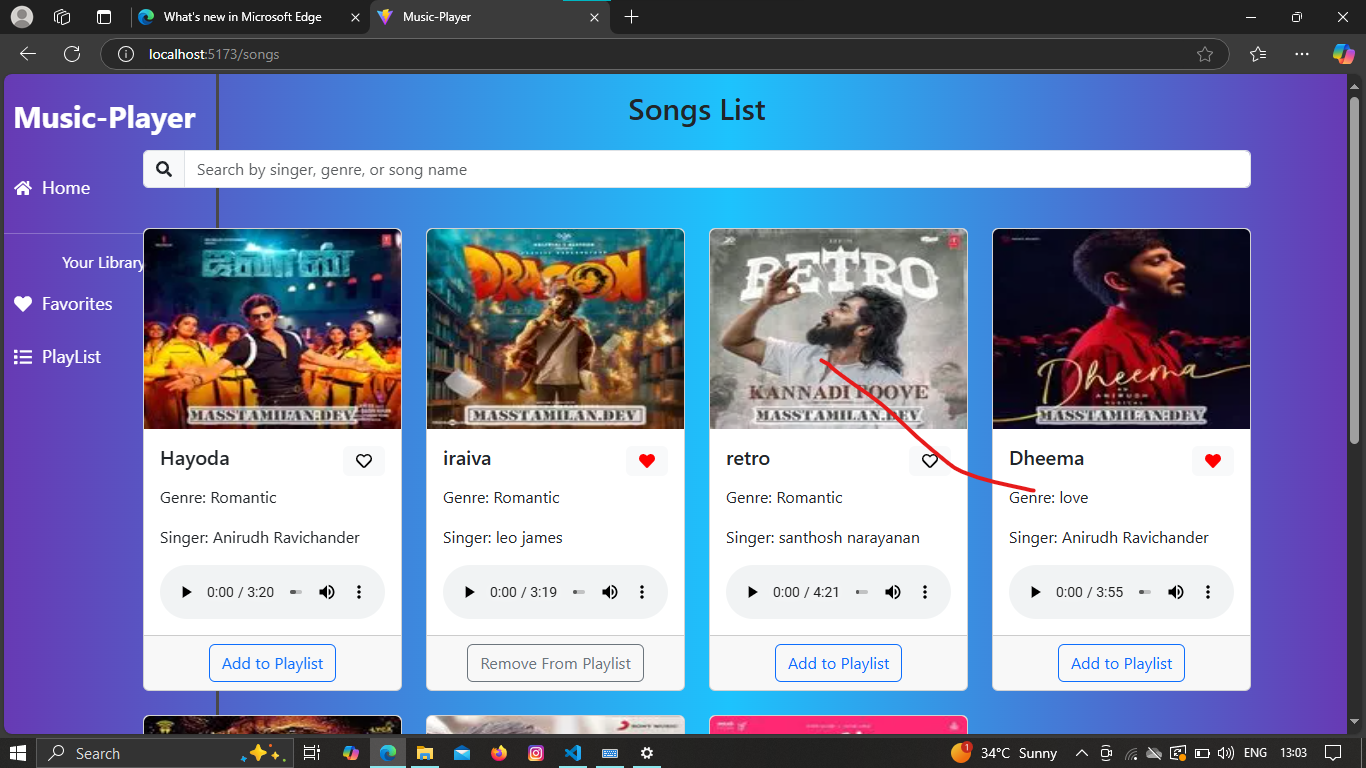
* + - **user:** Current authenticated user.
    - **player:** Current playing track, playback status (playing/paused), and volume.
    - **playlists:** User-created playlists.
    - **searchResults:** Results from the search functionality.

# Local State:

Local state is managed using React's useState hook within components. For example, the SearchPage component manages the search query input locally.

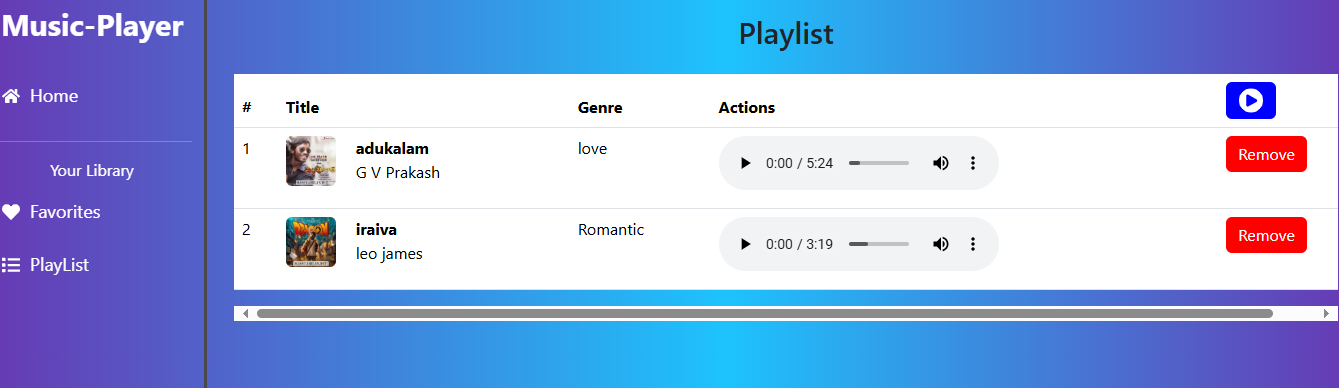
# User Interface

* + **Screenshots**
    - **Home Page:** Display featured tracks and recommended playlists.

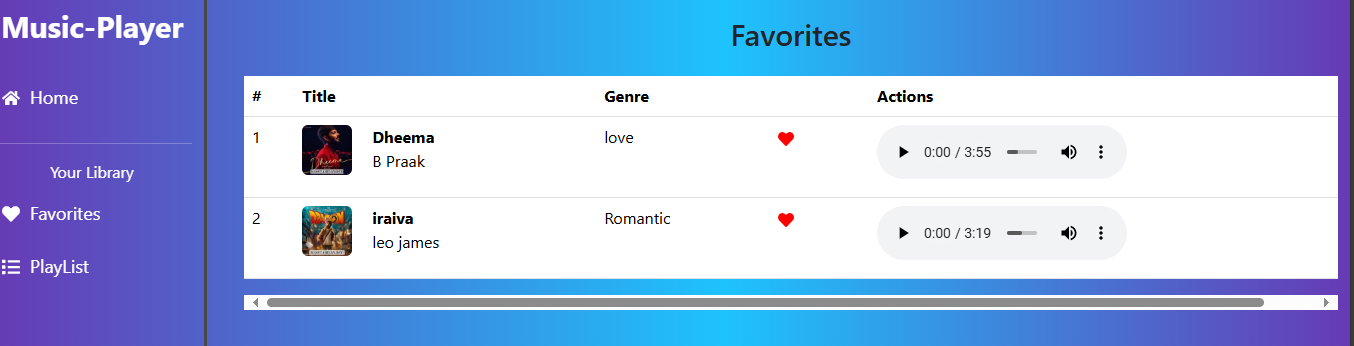


* + - **Search Page:** Allows users to search for songs, albums, and artists.

-4--\*+1



* + - **Playlist Page:** Displays user-created playlists and allows playlist management.



* Favorites Page : Users liked a songs that are stored in favorites.

# Styling

* + **CSS Frameworks/Libraries**:

The application uses **Styled-Components** for styling. This allows for modular and scoped CSS within components.

# How to use:

**Visit the site** : To access the app,go to react web Audio player in your browser.

**Start listening** : once on the site,you can immediately start playingsongs from the playlist.

# Screenshots or Demo

* + **Demo Link:**

https://drive.google.com/drive/folders/1-2dzv9XfsiZ6iZOmWyw\_WLiH5OuvbfmD

* + **Screenshots:** See section 9 for UI screenshots.

# Known Issues

* + **Issue 1**: The music player sometimes skips tracks unexpectedly.
  + **Issue 2**: The search functionality is slow with large datasets.

# Future Enhancements

* + **Future Features**:
    - Add support for user profiles and social sharing.
    - Implement a recommendation engine for personalized music suggestions.
    - Add animations and transitions for a smoother user experience.

14.CONCLUSION:

The React Web Audio Player is a modern, user-friendly music player application with a variety of customizable features. Whether you want to adjust the volume, change themes, or just enjoy your music, this app provides an enhanced listening experience. With its responsive design and ease of use, it is perfect for any music enthusiast looking for a customizable and interactive web player.

For any questions or feedback, feel free to visit the React Web Audio Player or reach out to the development team.