**Rhythmic Tunes (React)**

**1. Introduction**

**Project Title: Rhythmic Tunes (React)**

**Team Members:**

* **Team Leader:** K. Mangala Kowsalya
* **Team Members:**
  + V. Lakshmi
  + K. Kirthana
  + P. Monisha

**Team Collaboration: Building Rhythmic Tunes Together**

As we embark on this exciting journey to develop **Rhythmic Tunes**, I want to acknowledge the incredible team making it happen. Our dedication, creativity, and teamwork will shape this project into something truly special.

**Meet the Team**

* **K. Mangala Kowsalya** – Team Leader & Full-Stack Developer, ensuring smooth development and project coordination.
* V**. Lakshmi** – Frontend Developer & UI/UX Designer, crafting an intuitive and visually engaging user interface.
* K**. Kirthana** – Backend Developer & API Integration, managing data handling and backend functionality.
* P**. Monisha** – Testing & Quality Assurance, ensuring seamless performance and bug-free user experience.

**2. Project Overview**

**Purpose:**

Rhythmic Tunes is a modern music streaming application built with React.js, offering an immersive and interactive audio experience. It provides seamless access to a wide variety of music with a user-friendly interface that enables effortless exploration, playlist creation, and playback control.

**Features:**

* Song Listings with details such as title, artist, genre, and release date.
* Playlist Creation for personalized music collections.
* Playback Control including play, pause, skip, and volume adjustment.
* Offline Listening for convenient access to favorite tracks.
* Advanced Search Functionality for quick discovery of songs, artists, or albums.

**3. Architecture**

**Component Structure:**

* **Songs Component:** Displays song listings.
* **Playlist Component:** Manages user-created playlists.
* **Favourites Component:** Stores favorite songs.
* **Search Component:** Handles song search functionality.
* **Sidebar Component:** Provides navigation across the application.

**State Management:**

* **use State Hook** for managing local states like search terms and song lists.
* **use Effect Hook** for fetching data from APIs.
* **Axios** for handling HTTP requests.

**Routing:**

Implemented using react-router-dom:

* / - Displays the Songs component.
* /favourites - Displays the Favourites component.
* /playlist - Displays the Playlist component.

**4. Setup Instructions**

**Prerequisites:**

* Node.js and npm
* React.js
* Git for version control
* Code editor (e.g., Visual Studio Code, Sublime Text, or WebStorm)

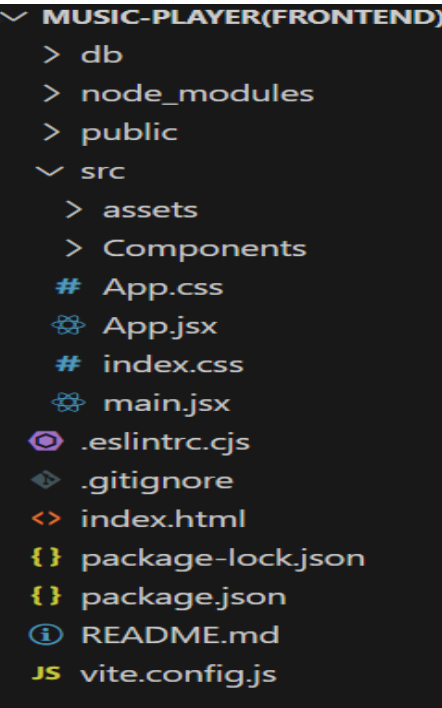
**Installation:**

1. Clone the repository:
2. git clone [repository-link]

<https://github.com/mangalakousalya/RYTHIMIC-TUNES>

1. Navigate to the project directory:
2. cd project-name
3. Install dependencies:
4. npm install
5. Start the development server:
6. npm run dev
7. Start the JSON server:
8. json-server --watch ./db/db.json
9. Open the application in a browser at http://localhost:5173.

**5. Folder Structure**



**Utilities:**

* **Helper functions** for handling API requests and playlist management.
* **Custom hooks** for fetching and managing global state.

**6. Running the Application**

* **Frontend:**
* npm start
* **JSON Server:**
* json-server --watch ./db/db.json

**7. Component Documentation**

**Key Components:**

* **Songs**: Displays a list of songs with play/pause functionality.
* **Playlist**: Manages user-added playlists.
* **Favourites**: Shows saved favorite songs.
* **Search**: Implements filtering by song title, artist, and genre.
* **Sidebar**: Provides easy navigation across different sections.

**Reusable Components:**

* **AudioPlayer**: Handles playback controls.
* **Button**: Customizable buttons for various actions.

**8. State Management**

**Global State:**

* Uses use State and use Effect to manage the global application state.
* Fetches data from local JSON server.

**Local State:**

* Maintains individual component states such as search terms and playlist contents.

**9. User Interface**

* Responsive design ensuring accessibility across devices.
* Smooth transitions and animations for an engaging experience.
* Screenshots and GIFs showcasing UI elements.

**10. Styling**

**CSS Frameworks/Libraries:**

* **Bootstrap/Tailwind CSS** for styling.
* **Custom CSS** for additional tweaks.

**Theming:**

* Light and dark mode options.
* Custom design system for consistency.

**11. Testing**

**Testing Strategy:**

* **Unit Testing:** Using Jest and React Testing Library.
* **Integration Testing:** Ensuring seamless component interaction.
* **End-to-End Testing:** Using Cypress for full application flow validation.

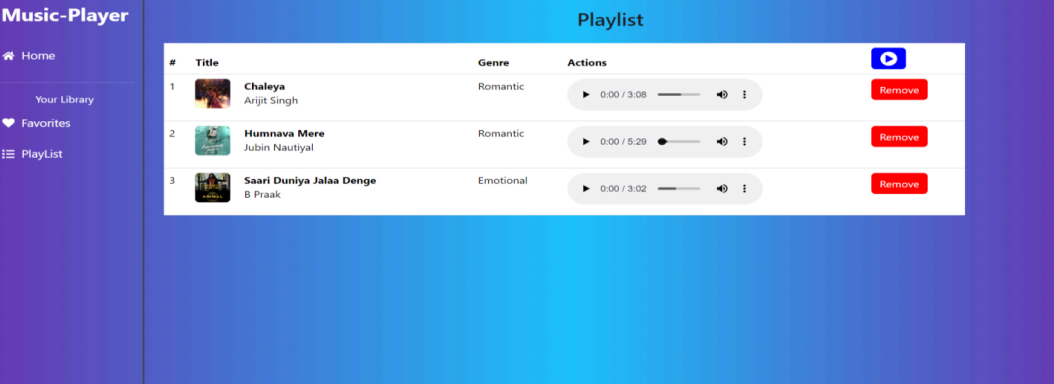
**Code Coverage:**

* Uses Jest coverage reports to ensure adequate test coverage.

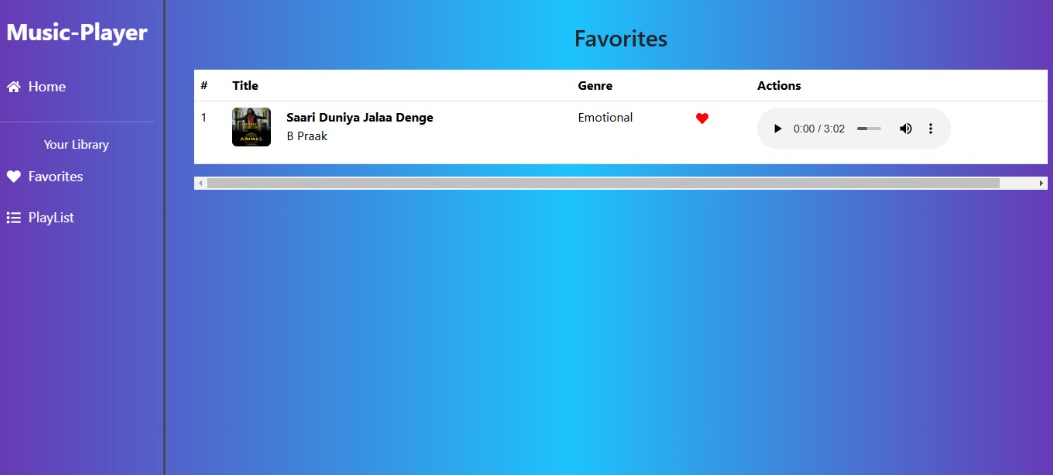
**12. Screenshots or Demo**

* Demo Link:
* Screenshots:

**Playlist**



**Favorites**



**13. Known Issues**

* Some features may have minor performance lags.
* Offline listening might require additional storage optimizations.

**14. Future Enhancements**

* Implementing AI-based music recommendations.
* Adding real-time lyrics display.
* Expanding offline playback options.
* Enhancing UI with modern animations and effects.

**15. Code Repository**

Use the source code link:

<https://drive.google.com/drive/folders/1boIbT2wQiih_w7YrvwQRUr1fGXzLJH0O>

Demo video-link: <https://drive.google.com/file/d/1W1FZYnNwoNVPl1pMCPEKZA83yBARwUwv/view?usp=sharing>