**FRONTEND DEVELOPMENT WITH REACT.JS**

1. **INTRODUCTION**

**Project Title**

**-** Rythmic Tunes Your Melodic Companion(Music Streaming )

**Team Members:**

* Vigneshwar J.K \_ (vigneshwarjk-cswithai@srmasc.ac.in)
* Keerthanapriya S – (s.keerthanapriya-cswithai@srmasc.ac.in)
* Avinash B – (avinashb-cswithai@srmasc.ac.in)
* Gowshicksriram S – (gowshicksrirams-cswithai@srmasc.ac.in)
* Vigneshwaran K \_ (vigneshwarank-cswithai@srmasc.ac.in)

1. **PROJECT OVERVIEW:**

**Purpose:**

RythmicTunes is an innovative and interactive web application aimed at delivering an engaging and immersive music experience to users. The platform is designed to cater to music enthusiasts by offering a seamless interface for song playback, allowing users to enjoy their favorite tracks effortlessly.

One of the key features of RythmicTunes is the ability to create and manage personalized playlists, enabling users to curate collections of songs based on their moods, preferences, and listening habits. This customization enhances the overall experience by providing a tailored selection of music for different occasions.

.

**Features:**

 **Song Playback:** Users can play a variety of songs directly on the platform for a seamless listening experience.

 **Favorites List:** Allows users to save and quickly access their favorite tracks without searching.

 **Personalized Playlists:** Users can create, customize, and manage their own playlists based on their preferences and moods.

1. **Architecture:**

**Component Structure:** The application is structured with a clear hierarchy of React components. The main components include:

* SongList: Displays a list of available songs.
* Playlists: Manages and displays user playlists.
* Favorites: Shows the user's favorite songs.
* These components interact to provide a seamless user experience.

**State Management:** We utilize React's Context API for efficient state management, allowing for easy prop drilling and maintaining a clean architecture.

**Routing:** React Router is employed for navigation, providing a smooth, single-page application feel.

1. **Setup Instructions:**

**Pre-requisites:**

Node.js, VScode.

**Installation:**

1. Install the Node.js, VScode to your local machine.
2. Navigate to the project directory in your terminal.
3. Run npm install npx to install all necessary dependencies.
4. Start the development server with npm start.
5. **Folder Structure:**

**Client:**

* Components: Contains all React components.
* Pages: Includes different pages of the application.
* Assets: Stores static files like images and stylesheets.

**Utilities:** Custom hooks and utility functions are located in the hooks directory to manage state and logic.

1. **Running the Application:**

To run the frontend server locally, use the command npm start in the client directory.

* Commands used:
* cd db
* npm install npx
* npx json-server –watch db.json
* npm i
* npm run dev

1. **Component Documentation:**

**Key Components:**

* SongList: Renders a list of songs and handles playback.
* Playlists: Manages playlist creation, deletion, and display.
* Favorites: Displays and manages the user's favorite songs.

**Reusable Components:** Components like SongCard and Button are designed to be reusable across the application.

1. **State Management:**

**Global State:** Context API is used to manage global state, such as the currently playing song and user preferences.

**Local State:** Local state within components is managed using the useState hook for simplicity and efficiency.

1. **User Interface:**

The UI is designed to be intuitive, with easy navigation and clear visual cues for playback and playlist management.

1. **CSS:**

**CSS Frameworks/Libraries:** We use Material-UI for consistent and responsive styling.

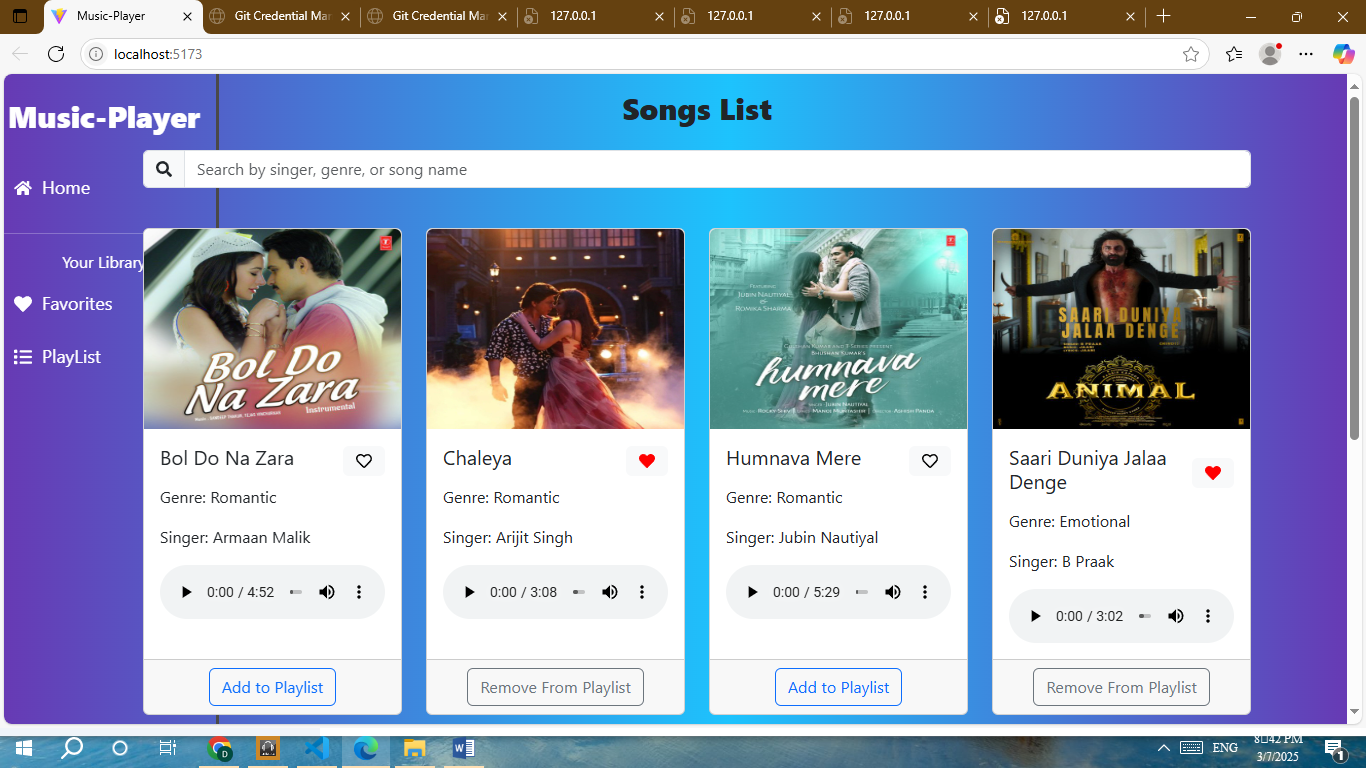
**Theming:** Custom themes are implemented to match the application's branding.

1. **Testing:**

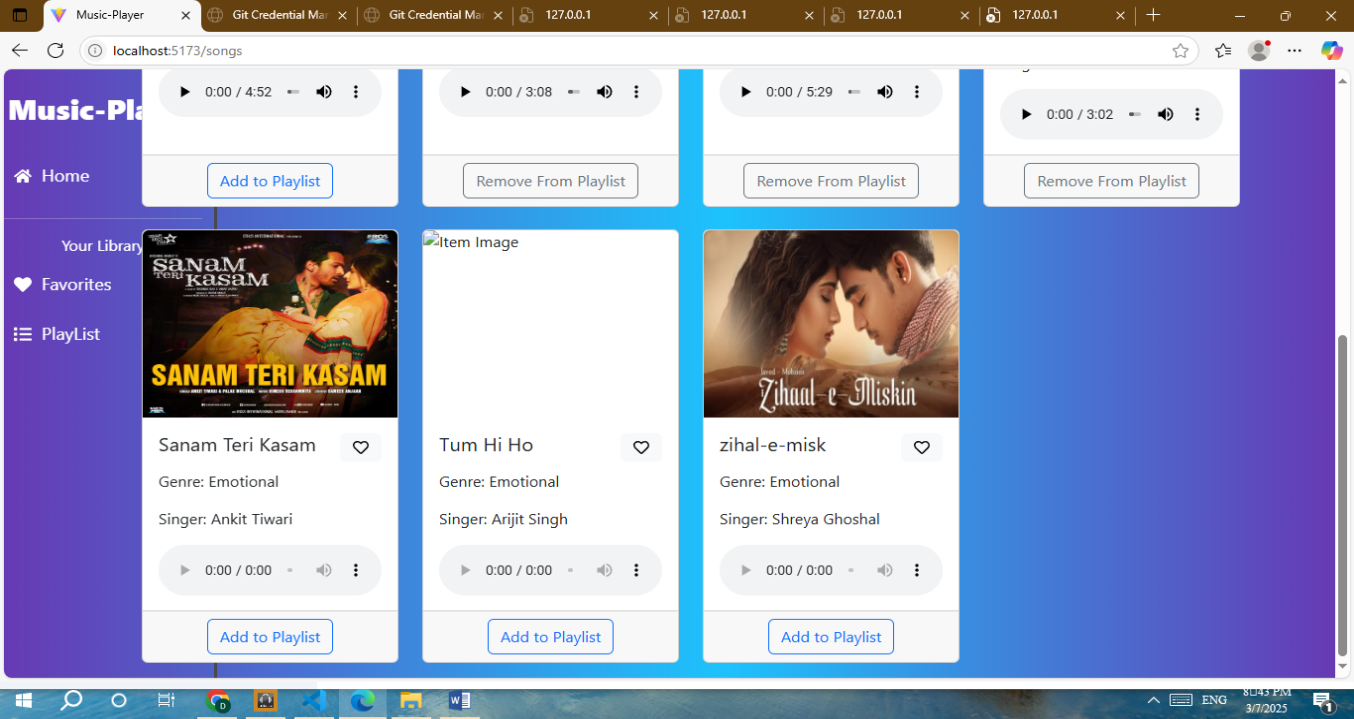
**Testing Strategy:** Jest and React Testing Library are used for unit and integration testing to ensure reliability.

**Code Coverage:** We aim for high test coverage to maintain code quality.

1. **Screenshots or Demo:**

****

**13**

****

1. **Known Issues:**

At present, there are no significant or major issues identified within the RythmicTunes application. The platform is functioning smoothly, with all core features—including song playback, personalized playlists, and the favorites list—operating as expected.

However, as with any web application, minor bugs or performance optimizations may arise over time. The development team remains proactive in monitoring system performance, gathering user feedback, and addressing any potential concerns that may emerge in future updates. Regular testing and quality assurance measures are in place to ensure a seamless and uninterrupted music experience for users.

1. **Future Enhancements:**

To further improve user experience and expand the capabilities of RythmicTunes, several enhancements are planned for future updates. These include:

* **Enhanced Recommendation System:** Implementing AI-driven music recommendations based on user listening habits and preferences.
* **Offline Mode:** Allowing users to download and listen to their favorite tracks without an internet connection.
* **Social Sharing Features:** Enabling users to share playlists and favorite songs with friends on social media platforms.
* **Multi-Device Synchronization:** Ensuring seamless music playback and playlist synchronization across multiple devices.