Rhythmic Tunes

**Introduction**

* + Project Title: Rhythmic Tunes
  + Project ID: SWTID1741152412
  + Team Leader: SIVA B
  + Team Member: SANTHOSH H
  + Team Member: SAMATHNATHAN P
  + Team Member: RAVIKRISHNAN C
  + Team Member: GOPINATH M

**Project Overview**

**Purpose:** Rhythmic Tunes is a music streaming application designed to provide users with an immersive and personalized audio experience. Built with React.js, it offers seamless music discovery, playlist creation, and playback controls.

**Features:**

* Song Listings (title, artist, genre, release date)
* Playlist Creation
* Playback Controls (play, pause, skip, volume adjustment)
* Offline Listening
* Search Functionality

**Architecture**

**Component Structure:**

* + - *App.js* - Root component, handles routing
    - *Songs.js* - Displays list of available songs
    - *Playlist.js* - Manages user-created playlists
    - *Favorites.js* - Handles favorited songs
    - *SearchBar.js* - Implements search functionality

**State Management:**

* Uses React's useState and useEffect hooks for local state management
* Data fetched and managed using axios
* useContext for global state (optional for future expansion)

**Routing:**

* Utilizes react-router-dom
* Routes include:
  + - * / - Song Listings
      * /playlist - User Playlists
      * /favorites - Favorited Songs

**Setup Instructions**

**Prerequistes:**

* + - Install [Node.js](https://nodejs.org/en/download/)
    - Install Git for version control ([Git Download](https://git-scm.com/downloads))
    - IDE: [VS Code](https://code.visualstudio.com/download)

**Installation:**

* git clone [repository\_url]
* cd rhythmic-tunes
* npm install

**Start the application:**

* npm run dev
* json-server --watch ./db/db.json
* # Start JSON server for backend simulation

**Folder Structure**

rhythmic-tunes/

│── src/

│ ├── components/ # Reusable UI components

│ ├── pages/ # Main pages (Songs, Playlist, Favorites)

│ ├── assets/ # Images, icons, etc.

│ ├── hooks/ # Custom React hooks

│ ├── context/ # State management (if applicable)

│── public/ # Static assets

│── package.json # Dependencies and scripts

│── README.md # Project documentation

**Running the Application**

* **Frontend:** npm run dev

**Component Documentation**

**Key Components:**

* + - *Songs.js:* Displays list of songs with search functionality
    - *Playlist.js:* Allows users to add/remove songs from playlists
    - *Favorites.js:* Manages favorite songs

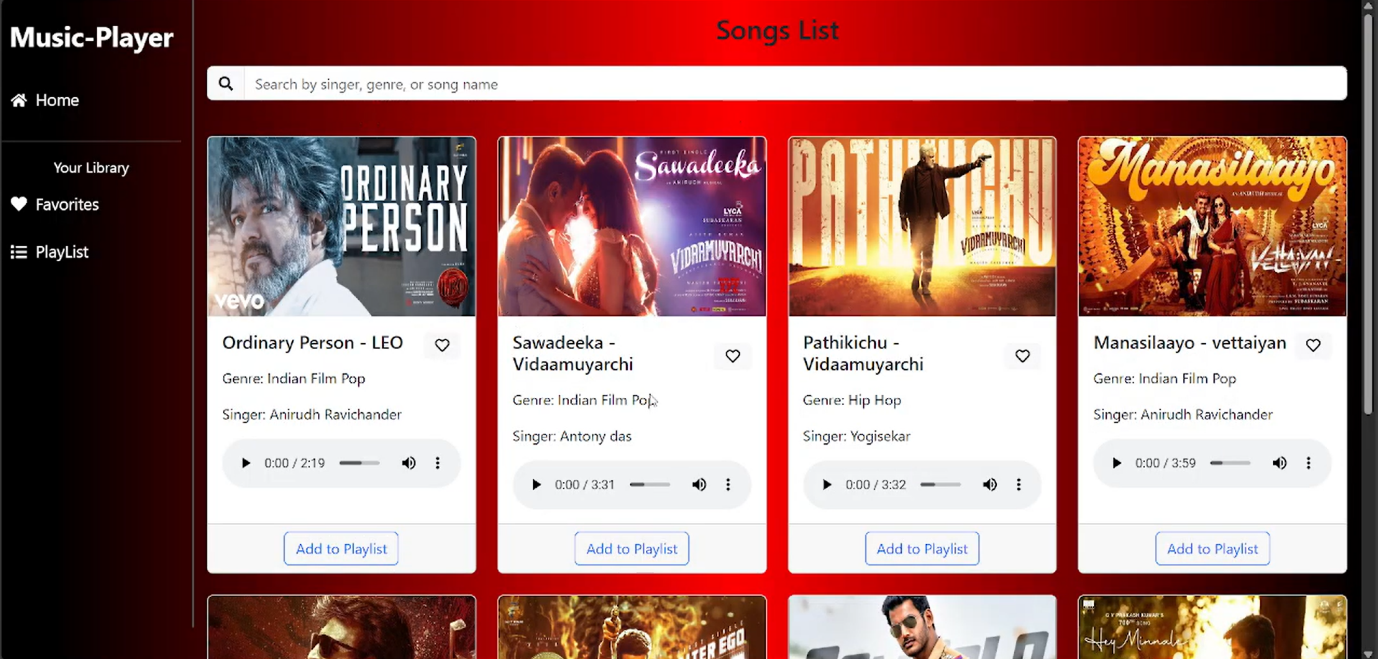
**Reusable Components:**

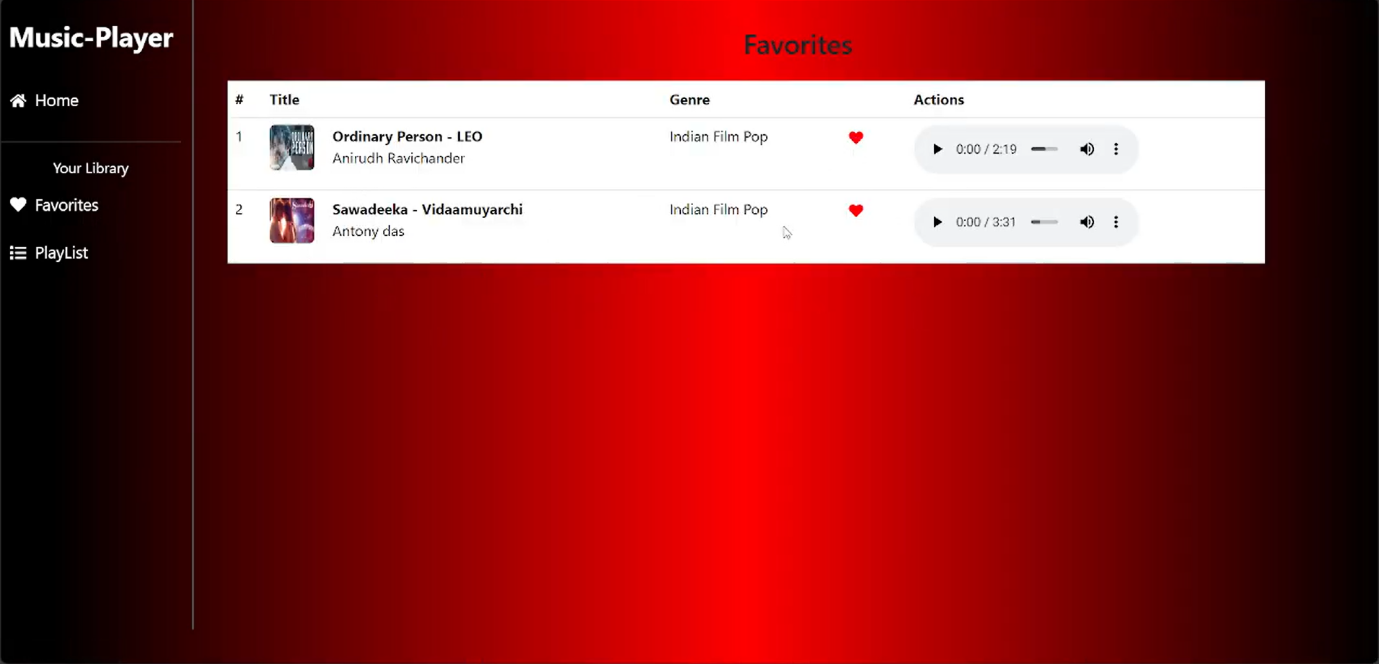
* *SearchBar.js:* Search input for filtering songs
* *AudioPlayer.js: Custom audio playback component*

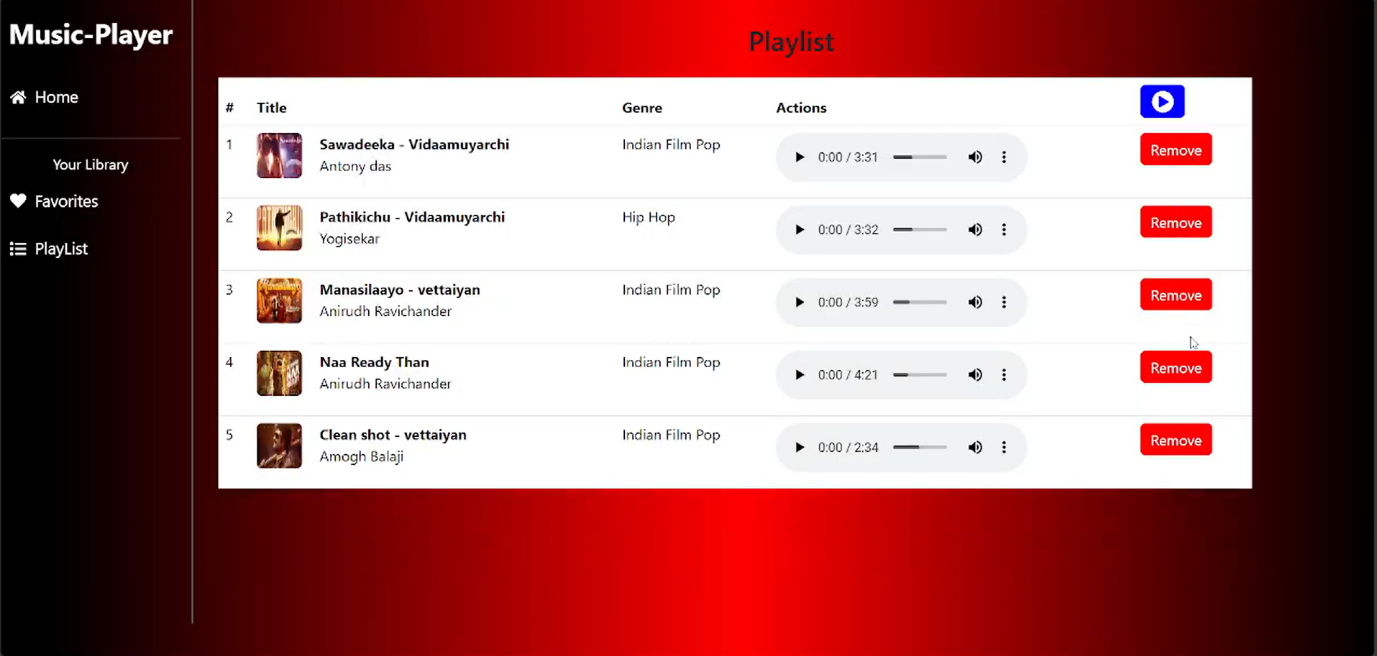
**State Management**

* **Global State:** Not yet implemented (potentially using Context API)
* **Local State:** Managed with useState and API calls with axios

**User Interface**

****

****

****

**Styling**

**CSS Frameworks/Libraries:**

* + - Bootstrap/Tailwind CSS
    - Custom CSS for additional styling

**Testing**

**Testing Strategy:**

* Unit tests with Jest & React Testing Library (future enhancement)

**Code Coverage:**

* No test coverage tool implemented yet

**Demo**

<https://drive.google.com/file/d/1tqY8mBg7Js8K3VCJvYELWf7UujlyotIR/view?usp=drivesdk>

**Known Issues**

* Playback might not work in some browsers due to CORS issues
* Limited offline listening support

**Future Enhancements**

* Implement user authentication
* Improve UI with animations
* Integrate a real music API for streaming