**RYTHIMIC TUNES**

**INTRODUCTION**

Welcome to **Rythimic Tunes** ,the future of musical indulgence awaits you with our cutting-edge music streaming application, meticulously crafted using the power of React.js. This platform seamlessly blends innovation with a user-centric design, redefining how you connect with music, with a heartfelt emphasis on the soulful melodies of Tamil songs. Designed for the modern music enthusiast, Rythimic Tunes offers a harmonious fusion of robust functionality and an intuitive interface, delivering a rich, immersive experience tailored to your unique taste.

**TEAM MEMBERS**

1 Team Leader –DEVAKUMAR S

Email –aruvaadeva420@gmail.com

1. Speaker About the Project -SURESH KUMAR R

Email –sureshsk2704@gmail.com

3 Document Creator – RAJA N

Email – rajadsouza22@yahoo.com

4 Notes Creator – SANJAY M

Email – sanjaysanjai2004@gmail.com

**SCENARIO-BASEDINTRO**

Picture yourself on a vibrant Chennai street: autorickshaws buzz past, vendors shout their wares, and faint Tamil melodies drift from a street performer’s flute. You’re heading to college, craving a mood lift. You pull out your phone, launch Rythimic Tunes, and with a few taps, you’re immersed in a personalized musical world.

The app’s smart curation kicks off with an upbeat Tamil hit like “Oh Penne” by Anirudh Ravichander, syncing perfectly with your stride. As you hop onto the metro, it transitions to the soothing “Vaseegara” by Bombay Jayashri, easing you into a calm commute. With Rythimic Tunes, every moment finds its perfect soundtrack, deeply rooted in Tamil culture.

**TARGET AUDIENCE**

**Rythimic Tunes caters to a vibrant and diverse audience, including:**

* **MUSIC ENTHUSIASTS:** Passionate listeners who relish music—especially Tamil songs—as a way to relax and connect with their preferences in their free time.
* **YOUNG ADULTS AND STUDENTS:** Tech-savvy individuals seeking an accessible platform to discover new Tamil hits and revisit classics during commutes, study breaks, or downtime.
* **GLOBAL TAMIL COMMUNITY:** Tamil speakers and fans worldwide eager to stay tethered to their cultural roots through a vast collection of Tamil music.

**PROJECT GOALS AND OBJECTIVES**

The primary goal of Rythimic Tunes is to deliver a seamless music streaming platform that empowers users to enjoy, discover, and share diverse musical experiences, spotlighting Tamil music. Our objectives are:

* **USER-FRIENDLY INTERFACE:** Craft an intuitive, responsive UI with dark theme support, enabling effortless exploration, saving, and sharing of tracks and playlists.
* **COMPREHENSIVE MUSIC STREAMING:** Offer robust features for music management, including advanced search by title or singer, YouTube-powered playback, and a curated Tamil-heavy song library (1300+ entries).
* **MODERN TECH STACK:** Utilize cutting-edge tools like React.js, React Router, Express.js, and Vite to ensure an efficient, enjoyable, and scalable user experience.

**KEY FEATURES**

* **SONG LISTINGS:** Browse a comprehensive song list (predominantly Tamil) with details like title, singer, and playback controls, fetched from http://localhost:5000/api/songs, styled in a responsive grid via SongCard.jsx.
* **PLAYLIST CREATION:** Create and manage personalized playlists through Playlist.jsx, adding/removing songs via POST /api/playlist and DELETE /api/playlist/:id, displayed in a clean list format.
* **FAVORITES MANAGEMENT:** Mark and access favorite tracks in Favorites.jsx, with add/remove functionality via POST /api/favorites and DELETE /api/favorites/:id, shown as a user-friendly list.
* **PLAYBACK CONTROL:** Enjoy seamless playback with play, pause, and progress tracking in SongCard.jsx and a fixed current song bar in App.jsx, powered by react-youtube.
* **SEARCH FUNCTIONALITY:** Find songs easily with a search bar in Header.jsx, filtering by title or singer, styled for clarity and responsiveness.
* **YOUTUBE AUDIO INTEGRATION:** Stream songs via YouTube URLs fetched from http://localhost:5000/api/youtube/:title/:singer using yt-search, with error handling in App.jsx.
* **RESPONSIVE NAVIGATION:** Navigate via a fixed, toggleable sidebar (Sidebar.jsx) linking to Songs, Favorites, and Playlist pages, styled with a dark theme in index.css.
* **DARK THEME SUPPORT:** Experience a sleek dark theme (with potential light theme expansion) using CSS variables in index.css, enhancing accessibility and comfort.

**PRE-REQUISITES**

**Node.js and npm**

* **Purpose:** Runs JavaScript locally and manages dependencies.
* **Install:** Download from [nodejs.org](https://nodejs.org/en/download/) and follow [installation instructions](https://nodejs.org/en/download/package-manager/).

**React.js**

* Purpose: Builds the interactive UI.
* Setup: Create a React app with Vite:
* npm create vite@latest rythimic-tunes -- --template react
* cd rythimic-tunes
* npm install
* npm run dev

**HTML, CSS, and JavaScript**

* Purpose: Core web technologies for structure, styling (with CSS variables), and interactivity.

**Version Control**

* Purpose: Track changes with Git.
* Install: Download from [git-scm.com](https://git-scm.com/downloads); host on GitHub or similar.

**Development Environment**

* Options: Use Visual Studio Code ([code.visualstudio.com](https://code.visualstudio.com/download)), Sublime Text ([sublimetext.com](https://www.sublimetext.com/download)), or WebStorm ([jetbrains.com/webstorm](https://www.jetbrains.com/webstorm/download)).

**YouTube Integration**

* Purpose: Fetches song URLs via yt-search in server.js; no API key needed (unlike the reference’s YouTube Data API suggestion).

**PROJECT STRUCTURE**

The project is structured for maintainability, separating frontend components, backend logic, styles, and configurations. Here’s the directory tree based on all provided files:

rythimic-tunes/

├── db/

│ └── db.json

├── public/

│ ├── vite.svg

│ └── db.json (server reference)

├── src/

│ ├── assets/

│ │ └── react.svg

│ ├── components/

│ │ ├── Favorites.jsx

│ │ ├── Header.jsx

│ │ ├── Playlist.jsx

│ │ ├── Sidebar.jsx

│ │ ├── SongCard.jsx

│ │ └── Songs.jsx

│ ├── App.jsx

│ ├── App.css

│ ├── index.css

│ └── main.jsx

├── .eslintrc.js

├── .gitignore

├── index.html

├── package-lock.json

├── package.json

├── README.md

├── server.js

└── vite.config.js

**FILE DESCRIPTIONS:**

* **DB/DB.JSON:** Stores song data (1300+ Tamil entries), favorites, and playlists, served via server.js at http://localhost:5000/api/.
* **PUBLIC/VITE.SVG:** Default Vite logo, used as favicon in index.html.
* **PUBLIC/DB.JSON:** Duplicate of db/db.json, read by server.js (assumed in public/).
* **SRC/ASSETS/REACT.SVG:** Placeholder image for UI elements like song cards.
* **SRC/COMPONENTS/FAVORITES.JSX:** 
  + Displays favorites from http://localhost:5000/api/favorites, with playlist toggling.
* **SRC/COMPONENTS/HEADER.JSX:** 
  + Search bar for filtering songs, styled with search-bar class.
* **SRC/COMPONENTS/PLAYLIST.JSX:**
  + Shows playlist songs from http://localhost:5000/api/playlist, with favorite toggling.
* **SRC/COMPONENTS/SIDEBAR.JSX:**
* Fixed, toggleable navigation (Songs, Favorites, Playlist), styled with sidebar class.
* **SRC/COMPONENTS/SONGCARD.JSX:**
* Displays song details with play, favorite, and playlist buttons, plus a progress bar.
* **SRC/COMPONENTS/SONGS.JSX:**
  + Lists songs fromhttp://localhost:5000/api/songs, with search and randomization (12 songs default).
* **SRC/APP.JSX:** Core component with routing, sidebar toggle, YouTube player, and error boundary.
* **SRC/APP.CSS:** Basic body styling (incomplete; complements index.css).
* **SRC/INDEX.CSS:** Global dark-themed styles for layout, responsiveness, and UI elements.
* **SRC/MAIN.JSX:** Entry point, rendering App with index.css.
* **.ESLINTRC.JS:** ESLint config with React hooks and custom rules.
* **.GITIGNORE:** Excludes node\_modules, dist, and editor files.
* **INDEX.HTML:** Base HTML with favicon and #root mount point.
* **PACKAGE-LOCK.JSON:** Locks dependency versions.
* **PACKAGE.JSON:** Defines scripts (start, dev, build, serve) and dependencies.
* **README.MD:** Minimal setup guide, suggesting TypeScript expansion.
* **SERVER.JS:** Express server with API endpoints and YouTube search via yt-search.
* **VITE.CONFIG.JS:** Vite config with React plugin.

**PROJECT FLOW**

**PROJECT DEMO**

* **Demo**:
* **GitHub Link**: <https://github.com/D83A9DAA8BED30D9540443D98D7617D5/music--SWTID1741261133149995>

**MILESTONE 1: PROJECT SETUP AND CONFIGURATION**

* **Install tools:** Node.js, npm, Git, and a code editor.
* Create React app with Vite (package.json, vite.config.js, index.html).
* Set up Express server (server.js) for db.json and YouTube integration.
* **Define API endpoints in server.js:** 
  + GET /api/songs
  + GET /api/favorites
  + GET /api/playlist
  + POST /api/favorites
  + POST /api/playlist
  + DELETE /api/favorites/:id
  + DELETE /api/playlist/:id
  + GET /api/youtube/:title/:singer
* **Run backend:** npm start (http://localhost:5000).
* **Run frontend:** npm run dev (http://localhost:5173).

**MILESTONE 2: PROJECT DEVELOPMENT**

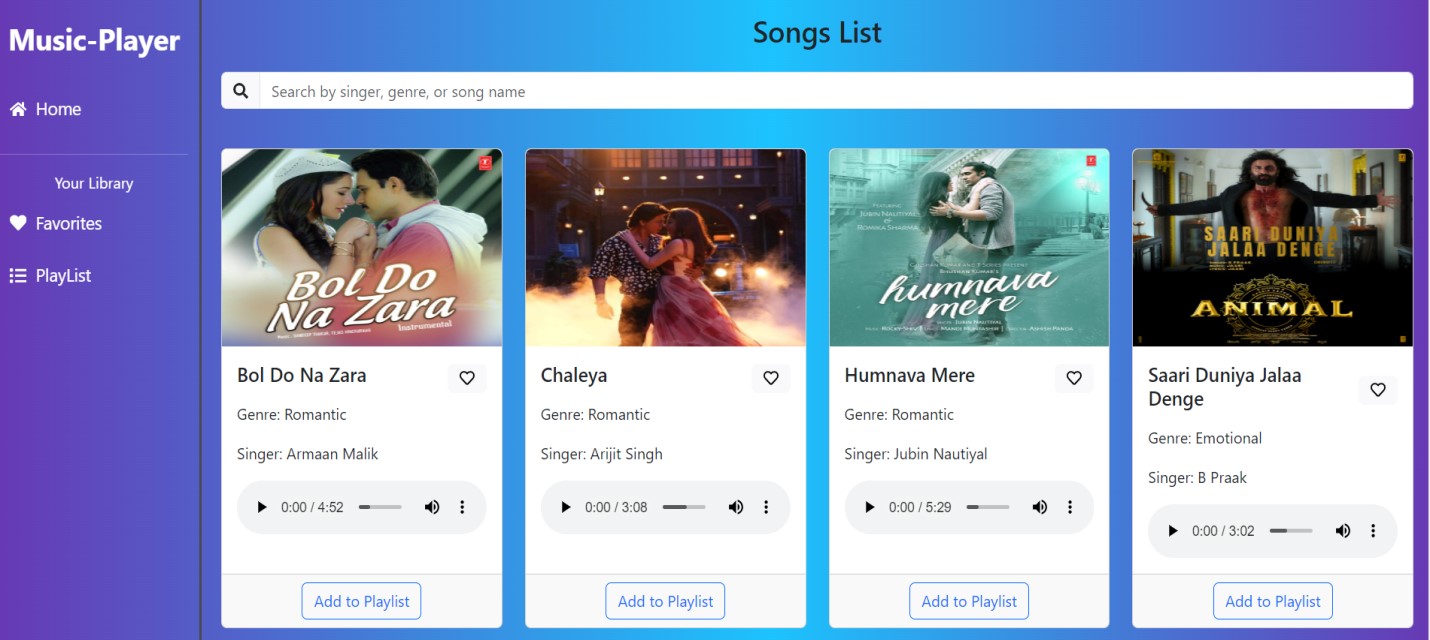
* Configure routing in App.jsx with react-router-dom.
* Install dependencies (package.json): react, axios, react-youtube, yt-search, etc.
* Develop components:
  + **Header.jsx**: Search bar.
  + **Sidebar.jsx:** Navigation.
  + **SongCard.jsx:** Song display and controls.
  + **Songs.jsx, Favorites.jsx, Playlist.jsx:** Core pages.
* Style with index.css (dark theme) and App.css (layout).

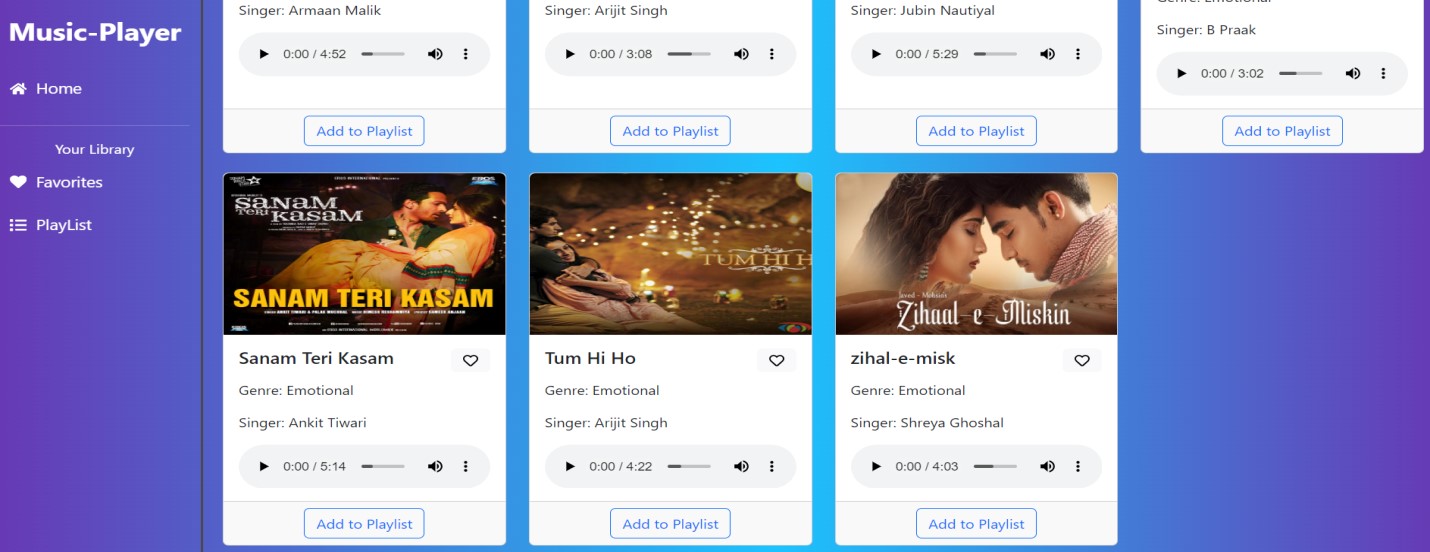
**PROJECT EXECUTION**

* **Start backend:** npm start.
* **Start frontend:** npm run dev (separate terminal).
* **Access at http:**//localhost:5173, ensuring backend is running for API calls.
* **Build for production:** npm run build, then npm run serve.

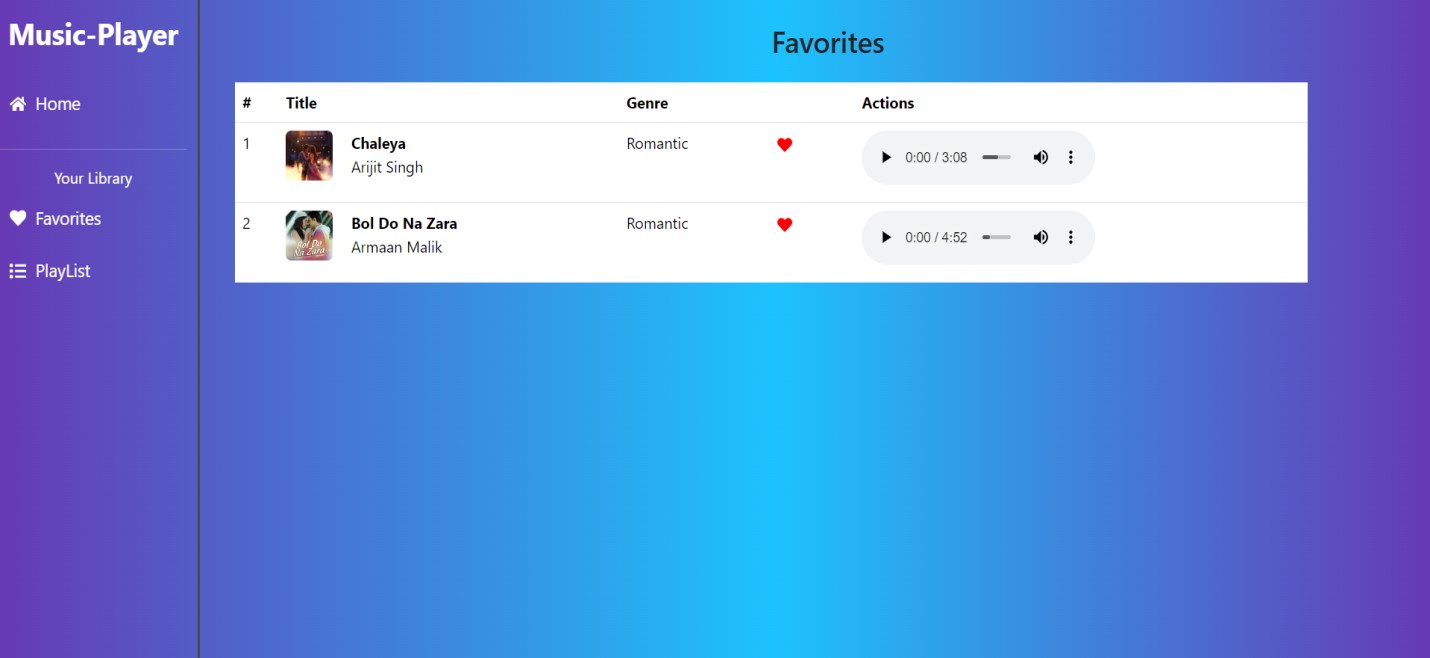
**SCREENSHOTS**

* **SONG LISTING:** Main page (Songs.jsx) with song cards and search bar.

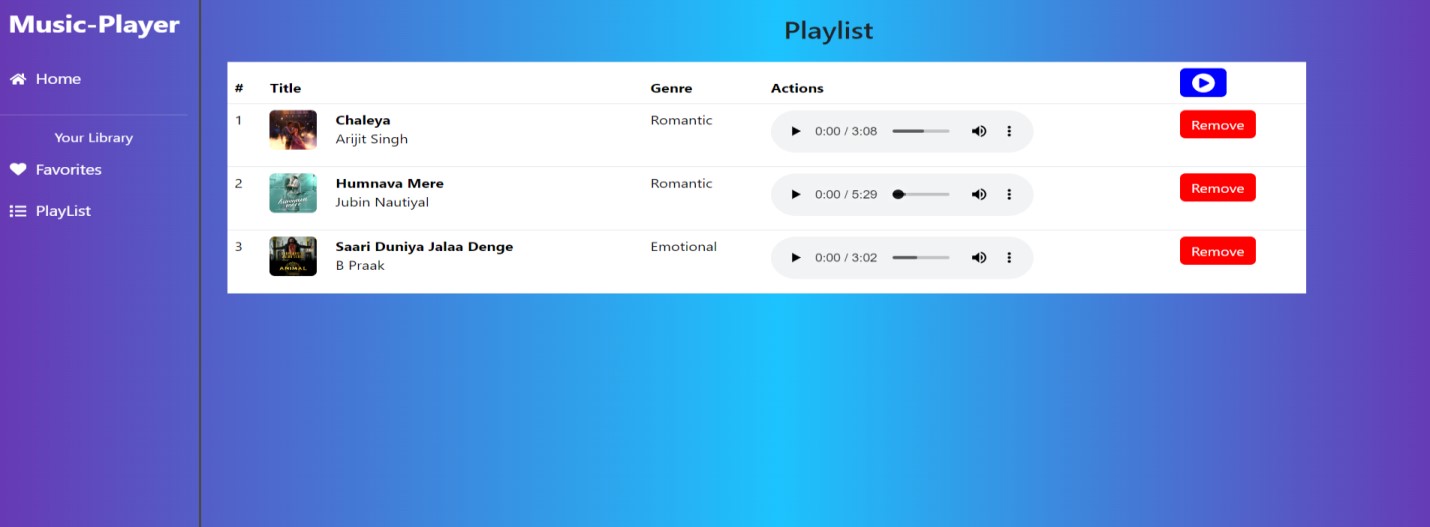




* **FAVORITES:** Favorite songs list (Favorites.jsx).



* **PLAYLIST:** Playlist songs list (Playlist.jsx).



**HAPPY CODING!!**

This document captures the essence of Rythimic Tunes—a Tamil music haven powered by modern tech. It’s built from your reference vision and updated files, ready to guide your next steps. Let me know if you want tweaks, buddy!