Frontend Development with React.js Project Documentation format

# Introduction

**Project Title:** [RhythmicTunes: Your Melodic Companion]

**Team Members*:***

***1)VIGNESH P:*** [Project Manager; Frontend Developer]

***2)KAMALSEH N K***:[Backend Developer]

***3)CHANDRAPRAKASH B***:[UI Desingner]

***4)JAGADHESAN S :***[Project video creater ]

***5)VIJAY A***

# Project Overview

## Purpose:

RhythmicTunes is a music streaming web application designed to provide users with a smooth, enjoyable, and personalized music experience. It allows users to explore, search, and play their favorite songs with an easy-to-use interface.

## Features:

\*User-friendly music player

\*Search and filter songs by artist, genre, or album

\*Playlist creation and management

\*Responsive UI for mobile and desktop

\*Dark/Light theme toggle

# Architecture

## Component Structure:

* + Navbar → Navigation menu (Home, Explore, Playlist, Now Playing)
  + SongCard → Shows song image, title, artist
  + MusicPlayer → Play, Pause, Next, Previous, Volume control
  + PlaylistManager → Create, view, and manage playlists
  + Footer → Bottom information (copyright, controls)
  + Tree-like structure:

App.js

├── Navbar

├── Pages

├── Home

├── Explore

├── Playlist

└── NowPlaying

├── Components

├── SongCard

├── MusicPlayer

└── PlaylistManager

└── Footer

## State Management:

\*Global State → Data needed everywhere (e.g., current playing song, playlists, theme mode).

\*Managed using Context API (or Redux if required).

\*Local State → Data only inside one component (e.g., text inside the Search bar).

\*Managed using useState hook.

## Routing:

\*<BrowserRouter>

<Routes>

<Route path="/" element={<Home />} />

<Route path="/explore" element={<Explore />} />

<Route path="/playlist" element={<Playlist />} />

<Route path="/now-playing" element={<NowPlaying />} />

</Routes>

</BrowserRouter>

# Setup Instructions

## Prerequisites:

* + Node.js,html,npm

## Installation:

git clone

https://github.com/sanjaysakthivel986-ux/Rythmic-tuness.git

Project Folder Name : rythmic tune npm install

npm start

# Folder Structure

## Client:

* + github Folder (Rythmic tunes):

\*Folder(db)

\*Folder(node\_modules)

\*Folder(public)

\*Folder(src)

\*File (.eslintrc)

\*File (.gitignore)

\*File (Index)

\*File (Package)

\*File (Package-lock)

\*File (README)

\*File (Vite.comfig)

## Utilities:

\* api.js

\*formatPrice.js

\*useLocalStorage.js(custom hook)

\*Helper functions for API calls, hooks

# Running the Application

## Frontend:

* + cd client

\*npm start

\*The application will run at [http://localhost:300**0**](http://localhost:3000/)

# Component Documentation

## Key Components:

* + MusicPlayer, PlaylistManager, SearchBar, Header

## Reusable Components:

* + Button, InputField, Card, Loader

# State Management

## Global State:

* + Managed using Context API (for user login, playlist data, etc.)

## Local State:

* + For UI-specific states like volume control, play/pause toggle.

# User Interface

## Screenshots/GIFs:

* + Home page

\*play list

\*Cart page

# Styling

## CSS Frameworks/Libraries:

* + Tailwind CSS / Styled Components

## Theming:

* + Dark mode & Light mode

# Testing

## Testing Strategy

* + Unit Testing → Testing small parts like SongCard or MusicPlayer component

individually.

\*Integration Testing → Checking if different components work together.

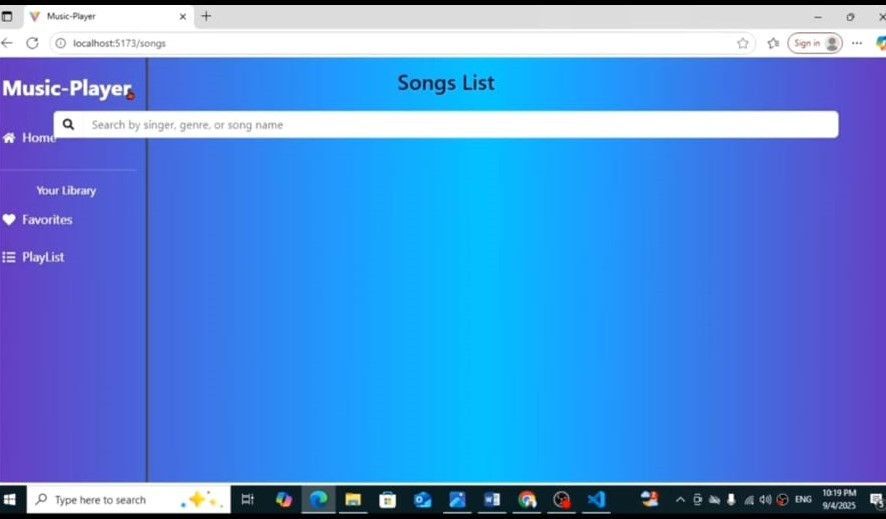
\*UI Testing → Ensuring the user interface looks and behaves correctly.

## Code Coverage:

\*Code coverage means checking how much of your project code is tested.

\*With Jest, you can generate a coverage report: npm test coverage

# Screenshots or Demo



1. **Known Issues**
   * Button component click delay
     + Form validation inconsistency
     + Dark mode theme flickering issue
     + API Data Load Delay

# Future Enhancements

\*Component Reusability Improvements

* + Advanced Theming System
  + Performance Optimization
  + Enhanced Testing
  + PWA Support
  + Mobile Responsive Enhancements Mobile-specific UI improvements.

Touch gestures support (Swipe, Pinch zoom).

\*API Error Handling Improvements.