**Music Streaming App – Project Documentation**

**1. Introduction :**

//Project Title: Rhythmic Tunes – Music Streaming Application

* Team Leader : DEEPIKA R
* Team member : BOOMA V
* Team member : PRIYADHARSHINI A S
* Team member : SIVANI K

**2. Project Objective :**

The primary goal of Music Streaming is to provide a seamless platform for music enthusiasts to enjoy and share diverse musical experiences.

**Objectives include**

1. User-Friendly Interface – Develop an intuitive interface that allows users to effortlessly explore, save, and share their favourite music tracks and playlists.

2. Comprehensive Music Streaming – Provide robust features for organizing and managing music content, including advanced search options for easy discovery.

3. Modern Tech Stack – Harness cutting-edge web development technologies, such as React.js, to ensure an efficient and enjoyable user experience.

**3. Pre-Requisites :**

To develop and run the project successfully, the following are required:

* Node.js and npm

Required to run React app and install dependencies.

* Download Node.js
* React.js

Library for building the frontend interface.

Create app with:

* npx create-react-app my-react-app
* cd my-react-app
* npm start

HTML, CSS, JavaScript – Basic knowledge for structure, styling, and interactivity.

Version Control (Git/GitHub)

* For collaboration and code tracking.

Download Git

* IDE / Code Editor

Recommended: Visual Studio Code

**4. Installation Steps:**

* Navigate into project

cd tunes

* Install dependencies

npm instastar

* Start React development server

npm start

* Access app at http://localhost:3000

**5. Project Structure :**

rhythmic-tunes/

│── public/ # Static assets

│── src/

│ ├── assets/ # Images, icons

│ ├── components/ # Navbar, Sidebar, Buttons, Modals

├── pages/ # Songs, Favorites, Playlist

│ ├── context/ # State management (if Context API used)

│ ├── utils/ # Helper functions

│ ├── App.js # Root component

│ ├── index.js # Entry point

│── db/ # Mock backend JSON server

│── package.json # Dependencies

* App.js – Root component with routes.
* Components – Navbar, Sidebar, Feature cards, Audio players.
* Pages – Songs list, Favorites, Playlist.

**6. Project Flow :**

* Demo Video: https://drive.google.com/file/d/1yU5sP5unhbVsPkyy98avuruTWsRmRgDv/view?usp=sharing
* Project Reference: https://drive.google.com/drive/folders/1daKg\_iUZeYy0C42k0wtURBz1ukg-wYE0?usp=sharing

**7. Project Setup and Configuration :**

Tools Used:

* React.js
* React Router DOM (navigation)
* React Icons (UI icons)
* Bootstrap / Tailwind CSS (styling)
* Axios (API calls)

Reference Docs:

\*React Installation

\*React Bootstrap

\*Axios

**8. Project Development :**

1. Setup React Application

Installed dependencies.

Configured routing with react-router-dom.

Imported Bootstrap CSS & custom styles.

Defined routes:

/ → Songs

/favorites → Favorites

/playlist → Playlist

2. Fetching Songs (Data Handling)

* Used axios + useEffect to fetch data from JSON Server:
* /items → All songs
* /favorites → Favorite songs
* /playlist → Playlist songs

State Management:

* items → all songs
* wishlist → favorites
* playlist → playlist items
* currentlyPlaying → active audio element
* searchTerm → live search

Audio Playback Management

* Ensures only one song plays at a time.

Wishlist / Playlist

* Add/remove songs via API requests (POST/DELETE).

Search Functionality

* Filters songs by title, genre, or singer.

3. Frontend Code (UI & Components)

* Songs Page
* Responsive cards with song image, title, genre, singer.
* Built with Bootstrap grid.
* Audio player included.
* Buttons for Add to Favorites / Add to Playlist.
* Search Bar
* Implemented with React Bootstrap InputGroup.
* Playlist & Favorites Pages
* Separate lists with add/remove functionality.

**9. Project Implementation & Execution :**

Run Application:

* npm start

Run JSON Server:

* json-server --watch ./db/db.json
* Open app at http://localhost:3000.

Launches Rhythmic Tunes frontend + backend mock API.