**Music Player Frontend Documentation**

**1. Introduction**

Project Title: Music Player Web App

Team Members:

**[Sarathi K]** – **[Frontend Developer]**

**[Bharath Guru V]** – **[Project Manager/Team Lead]**

**[Aravinth R ]** – **[Tester/QA]**

**[Kabilan B] ]** – **[UI/UX Designer]**

**2. Project Overview**

Purpose:

The Music Player Web App allows users to browse, search, and play songs, manage playlists, and

mark favorites. It provides a modern, responsive interface for music management and playback.

Features:

Song listing and playback

Playlist management

Favorites functionality

Sidebar navigation

Responsive design

**3. Architecture**

Component Structure:

App.jsx: Main entry, sets up routing and layout.

Sidebar.jsx: Navigation sidebar.

Songs.jsx: Displays and manages songs.

Favorities.jsx: Shows favorite songs.

Playlist.jsx: Manages user playlists.

Other components: Uhome.jsx, Uitem.jsx, Unavbar.jsx, Wishlist.jsx.

State Management:

Local state is managed using React's useState and useEffect hooks within components.

No global state management library (e.g., Redux) is used.

Routing:

Uses react-router-dom for client-side routing.

Main routes: /songs, /favorities, /playlist.

**4. Setup Instructions**

Prerequisites:

Node.js (v14 or higher)

npm (v6 or higher)

**Installation:**

**1.** Clone the repository:

git clone <repository-url>

**2**. Navigate to the project directory:

cd code

**3**. Install dependencies:

npm install

**4**. (Optional) Configure environment variables if needed.

**5**. Folder Structure

Client:

src/: Main source code

Components/: React components (Sidebar, Songs, Playlist, etc.)

assets/: Static assets (e.g., images, SVGs)

App.jsx, main.jsx: Entry points

App.css, index.css: Stylesheets

public/: Public assets (e.g., songs, images)

db/: Mock database (db.json)

Utilities:

Any helper functions or custom hooks are placed within the src/Components/ or src/ directory

as needed.

**6**. Running the Application

Frontend:

Start the development server:

npm start

The app will be available at http://localhost:5173 (default Vite port).

**7.** Component Documentation

Key Components:

Sidebar: Navigation between app sections.

Songs: Displays list of songs and playback controls.

Favorities: Shows user's favorite songs.

Playlist: Manages user playlists.

Reusable Components:

Uitem, Unavbar, Wishlist, etc. are used for UI consistency and reusability.

Props are passed as needed for data and event handling.

**8.** State Management

Global State:

Not implemented; state is managed locally within components.

Local State:

Managed using React hooks (useState, useEffect).

State flows from parent to child via props.

**9.** User Interface

The UI features a sidebar for navigation, main content area for songs/playlists, and responsive design

for different devices.

(Add screenshots or GIFs here)

**10**. Styling

CSS Frameworks/Libraries:

Bootstrap for layout and components.

Custom CSS in App.css, sidebar.css, uhome.css, etc.

Theming:

No custom theming system implemented.

**11.** Testing

Testing Strategy:

Manual testing of components and flows.

(Add details if using Jest or React Testing Library)

Code Coverage:

No automated code coverage tools configured.

**12.** Screenshots or Demo

(Insert screenshots or provide a link to a live demo here)

**13**. Known Issues

Some features may not be fully implemented.

No authentication or backend integration.

Limited error handling.

**14**. Future Enhancements

Add user authentication and backend integration.

Implement global state management (e.g., Redux).

Add more animations and enhanced styling.

Improve accessibility and mobile responsiveness.

Add automated testing and code coverage tools.