**Frontend Development with React.js – Project Documentation**

1. **Introduction:**

1.Project Title:

* Rhythmic Tunes: Your Melodic Companion

2.Team Members:

* Team Members:

1. YASMIN R

2. ELAMATHI M

3. CHITRA S

4. VINODHINI K

5. CHAITHRA R

* Team members Roles:

1. YASMIN R: Coding
2. ELAMATHI: M Coding Development
3. CHITRA S: Document
4. VINODHINI K: Document
5. CHAITHRA R: Demo Video
6. **Project Overview:**
7. Purpose:

* Enhance Melodic Creativity - Provide melodic patterns, harmonies, and inspiration for composing or practicing music.
* Support All Levels - Be equally useful for beginners building fundamentals and advancement musicians refining their craft.

1. **Features:**

* Rhythm Trainer – Provides guided exercises to build timing, tempo control, and rhythmic accuracy.
* Melody Companion – Suggests harmonies, scales, and melodic lines to support your compositions.

1. **Architecture:**
2. Component Structure:
3. Sidebar Component
4. Home
5. Your Library
6. Favourites
7. Playlist
8. State Management:
9. Current Track State
10. Playback State
11. Playlist State
12. Favourites states
13. Library States
14. UI State
15. Routing:
16. Application Initialization
17. Navigation initialization
18. Playlist & Song Loading Routine
19. Music Playback Routine
20. Favourite Routine
21. UI Interaction Routine
22. Exit Routine
23. **Setup Instruction**:
24. Prerequisites:
25. Windows 10/11
26. Node.js
27. npm (node manager)
28. React.js
29. Code Editor
30. Browser
31. Installation
32. npm create-vite my-app
33. cd my-app
34. npm install
35. npm run dev
36. **Folder Structure:**
37. Client:

* Open the application

(http://local host:5173)

1. Utilities:

* Format Time
* Music Player

1. **Running the Application:**

* Frontend: npm start in the client directory

1. **Component Documentation:**
2. Key Components: Document major components, their purpose, and any props they receive.
3. Reusable Components: The project uses several modular and reusable React components to ensure scalability, maintainability, and consistency in design.
4. **State Management:**
5. Global State:
6. Current Track State
7. Playback State
8. Playlist State
9. Favourites States
10. Library State
11. Local State:
12. Search input state
13. Volume Control State
14. Play button State
15. **User Interface:**

Designed to be simple, intuitive, and responsive, ensuring users can easily browse, play, and manage.

1. **Styling:**

* CSS Framework/Libraries

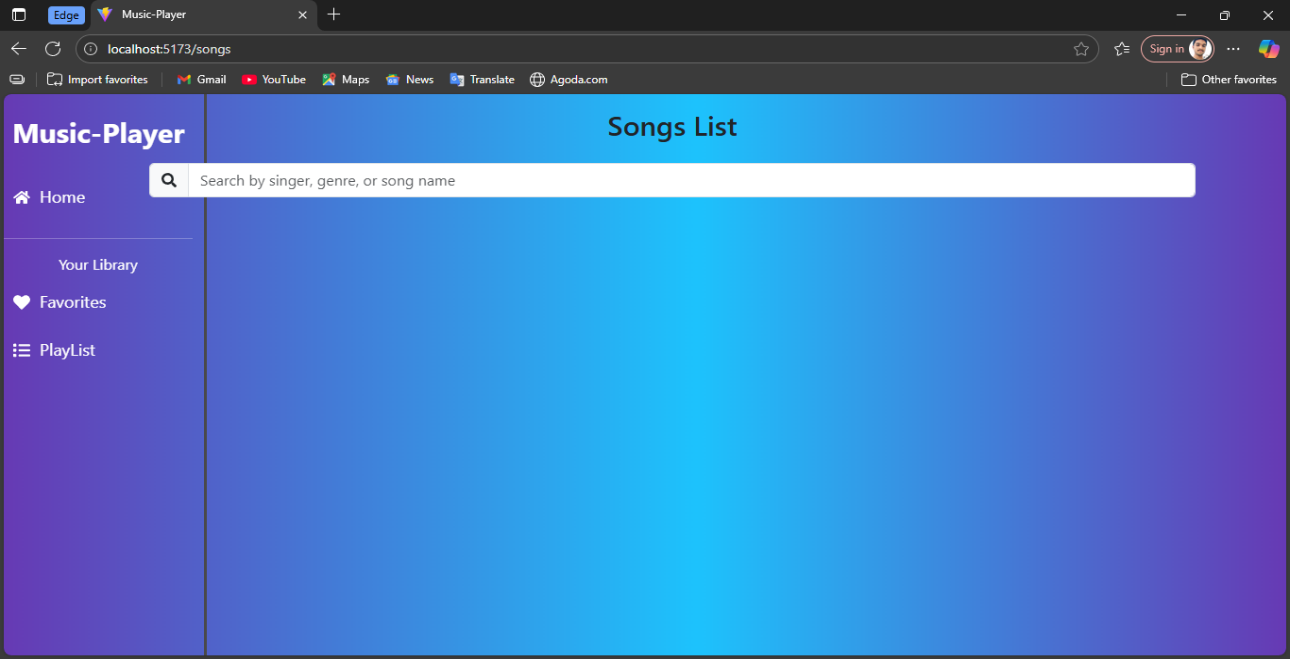
1. Introduction
2. Core Modules
3. State Management
4. **Testing:**
5. Testing Strategy

Utility function (shuffle playlist)

Custom hooks

1. Code Coverage:

* Unit Tests
* Integration tests
* E2E Test



1. **Screenshot or demo**
2. **Known issues:**

* Testing Coverage:

Some utility functions are not fully

Covered by automated tests.

* Performance Limitations

Slight lag may occur when handling very large playlists or demo.

1. **Future Enhancements**:
2. Equalizer support for custom sound
3. Personalization & Recommendation
4. Improved User Interface & Experience