Rhythmic Tunes - Project Documentation

**Introduction**  
**Project Title:** Rhythmic Tunes  
**Team Members:**

* **J. Ajay – (Team Leader)**
* **A.Abdul Rahman**
* **M.Abishek**
* **K.Anbarasan**

**Project Overview**  
**Purpose:**  
Rhythmic Tunes is a dynamic music platform designed to allow users to explore, play, and manage a collection of rhythmic tunes. The goal is to provide a seamless, interactive music experience with modern web technologies.

**Features:**

* Browse and search rhythmic tunes.
* Play audio tracks with an interactive player.
* User-friendly interface for adding and managing playlists.
* Responsive design for desktop and mobile devices.
* Authentication for user-specific playlists.

**Architecture**  
**Component Structure:**

* App.js: Root component that initializes routing and global context.
* Header.js: Navigation bar with links and user profile.
* HomePage.js: Displays featured tunes and search bar.
* TuneList.js: Displays a list of tunes with play buttons.
* TunePlayer.js: Audio player component with play, pause, and progress controls.
* PlaylistManager.js: Allows users to create and manage playlists.

**State Management:**

* Global State: Managed using React Context API to share state like current user, playlists, and currently playing track.
* Local State: Component-level state for UI toggles and form inputs.

**Routing:**  
Using react-router-dom for navigation:

* / → HomePage
* /tunes → TuneList
* /playlist → PlaylistManager
* /login → LoginPage

**Setup Instructions**

**Prerequisites:**

* Node.js (version >= 14)
* npm (version >= 6)

**Installation Steps:**

* Clone the repository: git clone https://github.com/your-repo/rhythmic-tunes.git
* Navigate to the project folder: cd rhythmic-tunes/client
* Install dependencies: npm install
* Configure environment variables in a .env file.

**Folder Structure**

/client

/src

/components

Header.js

TunePlayer.js

PlaylistManager.js

/pages

HomePage.js

LoginPage.js

TuneList.js

/assets

/images

/audio

/utils

api.js

auth.js

useCustomHook.js

App.js

index.js

**Utilities:**

* api.js: Handles API calls (GET, POST).
* auth.js: Authentication helper functions.
* useCustomHook.js: Custom hooks for reusable logic.

**Running the Application**  
From the client directory:  
npm start

**Component Documentation**  
**Key Components:**

* Header.js - Purpose: Navigation menu and user profile.  
  Props: user (object): User information.
* TunePlayer.js - Purpose: Plays selected audio track.  
  Props: track (object): { title, artist, url }

**Reusable Components:**

* Button.js - Configuration: Accepts onClick, label, disabled.
* Modal.js - Configuration: Displays custom content via children.

**State Management**  
Global State:  
Managed via React Context API. State includes: currentUser, playlists, currentlyPlayingTrack.

Local State:  
Form inputs in LoginPage and PlaylistManager use useState for controlled inputs.

**User Interface**  
Screenshots/GIFs will be provided showing: Home page layout, Tune list display, Audio player interactions, Playlist management UI.

**Styling**  
CSS Frameworks/Libraries:

* Tailwind CSS for utility-first styling.
* Styled-Components for component-level styles.

**Theming:**  
Light and Dark mode implemented using context and CSS variables.

**Testing**  
**Testing Strategy:**

* Unit Tests: Jest + React Testing Library.
* Integration Tests: Test API interaction and state flow.
* E2E Tests: Cypress for end-to-end user flows.

**Code Coverage:**  
Configured using Jest with coverage reports generated after tests.  
Command: npm run test -- --coverage

**Screenshots or Demo**  
Live demo available at: [https://rhythmic-tunes-demo.example.com](https://rhythmic-tunes-demo.example.com/)  
Screenshots to be included of Home, Tune List, Player, Playlist Manager.

**Known Issues**

* Minor delay in loading large playlists.
* Occasional player stutter on mobile devices.

**Future Enhancements**

* Add animations for smoother transitions.
* Implement advanced filtering options in TuneList.
* Add collaborative playlist functionality.
* Improve accessibility (ARIA compliance).