Frontend Development with React.js

Project Documentation format

1. **Introduction**
   - **Project Title**: Rhythmic tunes
   - **Team Members**: Grahitha k
   - Harshini G
   - Girija K
   - Divya A
   - Anju v

   .

2. **Project Overview**
   - **Purpose**:

     This project introduces an innovative music application designed to explore the intersection of rhythm, sound, and visual abstraction. The app allows users to create and manipulate complex rhythmic patterns, experiment with polyrhythms, and visualize their compositions in dynamic, fluid designs. By integrating customizable soundscapes and real-time audio-visual feedback, the app provides an immersive experience where users can engage with music beyond traditional boundaries. Whether composing original pieces, remixing beats, or collaborating with others, the app transforms rhythmic exploration into a multi-sensory journey. The use of abstract visualizations responds to changes in tempo, pitch, and structure, offering an intuitive and creative environment for both amateur and seasoned musicians alike. Through this project, we aim to redefine how people experience music, blending auditory elements with abstract art to create a truly interactive and experimental platform.

Features: 1. Wishlist & Favorites Management

- **Create a Wishlist**: Users can add specific sounds, beats, loops, or compositions to a personalized wishlist for easy access later. Perfect for collecting favorite samples or instruments they want to use in future projects.
- **Favorites**: Quickly mark specific tracks, rhythms, or soundscapes as favorites, allowing them to easily revisit their best-loved creations.
- **Track Collection**: Organize favorite compositions and sounds in a dedicated section for efficient project curation.

2. Like & Share System

- **Like Content**: Users can "like" tracks, rhythms, or soundscapes within the app to show appreciation or to save them in a simple, accessible way.
- **Social Integration**: Liked content can be shared with friends or the community through the app's built-in social platform, helping foster engagement and feedback.

3. Collaboration and Community Engagement

- **Shared Favorites**: Easily share your favorite beats or compositions with other users. It fosters a sense of community by allowing others to remix or collaborate on your favorite tracks.
- **Follow Creators**: Users can follow creators whose music or rhythm styles they enjoy, getting updates on new projects, rhythms, and sound packs they share.

4. Personalized Recommendations

- **Smart Recommendations**: The app can suggest new rhythms, sounds, and tracks based on your likes, favorites, and wishlist. This helps users discover new music and inspiration that aligns with their personal preferences.
- **Trending Content**: Showcase popular or highly liked tracks, loops, and soundscapes to spark creativity and help users stay up-to-date with the latest trends.

5. Advanced Search & Filtering

- **Filter by Favorites/Liked Content**: Users can easily search through their liked content, favorites, and wishlist to quickly find previously saved tracks or sounds.
- **Explore Section**: A dynamic section that updates based on the user's activity, showing content that aligns with their interests, likes, and wishlist items.

3. Architecture

The solution architecture for **HarmonyStream**, the Rhythmic Tunes Application, ensures a **scalable, high-performance, and immersive platform** for discovering, streaming, and organizing music across genres, artists, and personalized playlists. The architecture prioritizes seamless audio delivery, real-time recommendations, and cross-device synchronization to enhance user engagement.

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

4. Setup Instructions

- **Prerequisites**: Node.js,Bootstrap, axios,react js
- **Installation**: clone the repository, install dependencies, and configure environment variables.

5. Folder Structure

- **Client**: Describe the organization of the React application, including folders like components, pages, assets, etc.

    - **Utilities**: Explain any helper functions, utility classes, or custom hooks used in the project.
6. **Running the Application**
    - Provide commands to start the frontend server locally.
        - **Frontend**: npm start in the client directory.
7. **Component Documentation**
    - **Key Components**: Document major components, their purpose, and any props they receive.
    - **Reusable Components**: Detail any reusable components and their configurations.
8. **State Management**
    - **Global State**: Describe global state management and how state flows across the application.
    - **Local State**: Explain the handling of local states within components.
9. **User Interface**
    - Provide screenshots or GIFs showcasing different UI features, such as pages, forms, or interactions.
10. **Styling**

- **CSS Frameworks/Libraries**: Describe any CSS frameworks, libraries, or pre-processors (e.g., Sass, Styled-Components) used.
- **Theming**: Explain if theming or custom design systems are implemented.

11. **Testing**

- **Testing Strategy**: Describe the testing approach for components, including unit, integration, and end-to-end testing (e.g., using Jest, React Testing Library).
- **Code Coverage**: Explain any tools or techniques used for ensuring adequate test coverage.

12. **Screenshots or Demo**

    https://drive.google.com/file/d/1mg7233Uwhm_GelQOaPzy-I2QUugQZLFu/view?usp=drivesdk

13. **Known Issues**

- Downloading issue of npm installer.

14. **Future Enhancements**

    Automatic suggestion for user in music streaming app.