**Rhythmic Tunes**

**Introduction:-**

We believe that music is more than just sound; it’s an experience. That’s why we’ve created an app that combines the best of modern web development with a passion for music. Built using **HTML**, **CSS**, **JavaScript**, **React.js**, and **Node.js**, our app provides an intuitive and immersive music streaming experience designed for every listener. Whether you're jamming to your favorite tunes or exploring new genres, [App Name] offers a seamless, engaging platform to enjoy music anytime, anywhere.

**A Seamless, User-Centric Experience**

With a clean, responsive design powered by **React.js**, our app ensures smooth navigation across all devices. Whether you’re using your laptop, tablet, or phone, you’ll find that the interface adapts beautifully to your screen size. Say goodbye to clunky, outdated layouts and hello to a modern, intuitive design that’s easy to use and fun to explore.

**Fast and Reliable Performance**

Behind the scenes, we’ve harnessed the power of **Node.js** to ensure fast load times and real-time streaming. Our app’s back-end is optimized for minimal latency, delivering a smooth listening experience even when your playlist is filled with high-quality audio. Whether you’re listening to the latest chart-toppers or rediscovering older tracks, our app ensures that you spend more time enjoying music and less time waiting.

**Personalized Music at Your Fingertips**

we put the power in your hands. Easily create personalized playlists, discover new artists, and enjoy tailored recommendations based on your listening habits. Whether you're in the mood for upbeat tracks or relaxing instrumental sounds, our app learns your preferences and curates suggestions just for you. Plus, with offline listening capabilities, you can take your music anywhere without needing a Wi-Fi connection.

**Target Audience:-**

Music Streaming is designed for a diverse audience, including:

● Music Enthusiasts: People passionate about enjoying and listening Music Through out there free time to relax themselves.

Project Goals and Objectives:-

The primary goal of Music Streaming is to provide a seamless platform for music enthusiasts, enjoying, and sharing diverse musical experiences. Our objectives include:

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

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

Modern Tech Stack: Harness cutting-edge web development technologies, such as React.js, to ensure an efficient and enjoyable user experience while navigating and interacting with the music streaming application.

Key Features:-

Song Listings: Display a comprehensive list of available songs with details such as title, artist, genre, and release date.

Playlist Creation: Empower users to create personalized playlists, adding and organizing songs based on their preferences.

Playback Control: Implement seamless playback control features, allowing users to play, pause, skip, and adjust volume during music playback.

Offline Listening: Allow users to download songs for offline listening, enhancing the app's accessibility and convenience.

Search Functionality: Implement a robust search feature for users to easily find specific songs, artists, or albums within the app.

REQUIREMENTS:

Here are the key prerequisites for developing a frontend application using React.js:

✔ Node.js and npm:

Node.js is a powerful JavaScript runtime environment that allows you to run JavaScript code on the local environment. It provides a scalable and efficient platform for building network applications.

Install Node.js and npm on your development machine, as they are required to run JavaScript on the server-side.

● Download: https://nodejs.org/en/download/

✔ React.js:

React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications. Install React.js, a JavaScript library for building user interfaces.

● Navigate to the project directory: cd my-react-app

● Running the React App:

With the React app created, you can now start the development server and see your React application in action.

● Start the development server: npm start

✔ HTML, CSS, and JavaScript:

Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

✔ Development Environment:

Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

\* Visual Studio Code:

Download from https://code.visualstudio.com/download

To get the Application project from drive:

Follow below steps:

✔ Get the code:

• Download the code from the drive link given below:

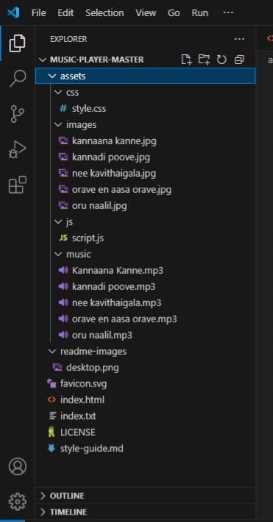
Install Dependencies:

\* Navigate into the cloned repository directory and install libraries: cd fitness-app-react npm install

✔ Start the Development Server:

• To start the development server, execute the following command: npm start

PROJECT STRUCTURE:

******

The project structure may vary depending on the specific library, framework, programming language, or development approach used. It's essential to organize the files and directories in a logical and consistent manner to improve code maintainability and collaboration among developers.

app/app.component.css, src/app/app.component: These files are part of the main AppComponent, which serves as the root component for the React app. The component handles the overall layout and includes the router outlet for loading different components based on the current route

PROJECT FLOW:-

Project demo:

Before starting to work on this project, let’s see the demo.

Demolink:

https://drive.google.com/file/d/1zZuq62lyYNV\_k5uu0SFjoWa35UgQ4LA9/view?usp=driv e\_link

Use the code in: https://drive.google.com/drive/folders/1BkYWfW\_K3ek\_UgtXNTAsDqlhdCuqz6nT?usp= drive\_link

Milestone 1: Project Setup and Configuration:

1. Install required tools and software:

● Installation of required tools:

1. Open the project folder to install necessary tools In this project, we use:

* React Js
* React Router Dom
* React Icons
* Bootstrap/tailwind css
* Axios

For further reference, use the following resources

o <https://react.dev/learn/installation>

o <https://react-bootstrap-v4.netlify.app/getting-started/introduction/>

o <https://axios-http.com/docs/intro>

o <https://reactrouter.com/en/main/start/tutorial>

Milestone 2: Project Development:

1. Setup React Application:
   * Create React application.
   * Configure Routing.
   * Install required libraries.
   * Setting Up Routes:-

Code Description:-

● Imports Bootstrap CSS (bootstrap/dist/css/bootstrap.min.css) for styling components.

● Imports custom CSS (./App.css) for additional styling.

● Imports BrowserRouter, Routes, and Route from react-router-dom for setting up client-side routing in the application.

● Defines the App functional component that serves as the root component of the application.

● Uses BrowserRouter as the router container to enable routing functionality.

● Includes a div as the root container for the application.

● Within BrowserRouter, wraps components inside two div containers:

* The first div contains the Sidebar component, likely serving navigation or additional content.
* The second div contains the Routes component from React Router, which handles rendering components based on the current route.
* Inside Routes, defines several Route components:
* Route with path='/' renders the Songs component when the root path is accessed (/).
* Route with path='/favorities' renders the Favorities component when the /favorities path is accessed. o Route with path='/playlist' renders the Playlist component when the /playlist path is accessed.

● Exports the App component as the default export, making it available for use in other parts of the application.

Code Description:

- ● useState:

o items: Holds an array of all items fetched from <http://localhost:3000/items>.

o wishlist: Stores items marked as favorites fetched from <http://localhost:3000/favorities>.

o playlist: Stores items added to the playlist fetched from <http://localhost:3000/playlist>.