***Frontend Development with React.js***

***Project Documentation for fitflex (fitness tracker)***

**1. Introduction**

**Project Title**: **FITFLEX (FITNESS TRACKER)**

• **Team Members**:

**KARTHIK RAJA G** (**Team Leader**) [Email Id: karthikrajaganesan9@gmail.com]



**JUBBAIR SHERIFF M**  [Email Id: jubbair8@gmail.com]

**ANTONY CHRISTOPHER J** [Email Id: stevenchristo0110@gmail.com ]

**GOPINATH R**  [Email Id: r.gopinath.bca@gmail.com ]

**KARTHIK L** [Email Id: deepalaks617@gmail.com]

**2. Project Overview**

• **Purpose:** Fitness Tracker is a web application designed to help users monitor and improve their health and fitness levels. The application allows users to track their physical activities, set fitness goals, monitor progress, and stay motivated on their journey towards a healthier lifestyle.

**3. Architecture**

The **Fitness Tracker** application is built using **React.js** with a component-based architecture, ensuring modularity, reusability, and scalability. The core components of the application include:

1. **Header:**
   * Contains the navigation bar with links to different sections of the app (e.g., Home, Profile, Settings).
   * Includes a search bar for quick access to specific workouts, progress, or features.
2. **Tracker:**
   * Centralized component for tracking activities.
   * Displays the real-time data like duration, calories burned, heart rate, distance covered, etc.
   * Provides controls to start, pause, or reset the workout session.
3. **Sidebar:**
   * Displays links to different sections like the user’s workout history, goals, progress charts, and personal settings.
   * Shows recent activities, goals, and quick links to workout types (running, cycling, etc.).
4. **HomePage:**
   * **Featured Workouts:** Highlights trending or recommended workouts. o **Progress Overview:** A quick snapshot of daily/weekly progress towards

goals.

* + **Motivational Tips:** Provides fitness tips, news, or challenges to engage users.

1. **SearchPage:**
   * Allows users to search for exercises, workout routines, or fitness articles.
   * Provides filters like difficulty level, type of workout, or duration.
2. **ProfilePage:**
   * Displays the user’s personal information and fitness stats (weight, height, BMI, etc.).
   * Allows users to track their progress and update their goals and preferences.
3. **GoalPage:**
   * Displays user-defined fitness goals (e.g., steps per day, calories burned).
   * Allows users to adjust goals, track progress, and receive motivational feedback.

**4. Setup Instructions**

**Prerequisites**:

o Node.js (v16 or higher) o npm (v8 or higher) o Git

• **Installation**:

1. Clone the repository: git clone https://github.com/Karthik14023 /FITNESS-APP
2. Navigate to the client directory: cd fitflex/client
3. Install dependencies: npm install
4. Configure environment variables: Create a .env file in the client directory and add the necessary variables (e.g., API keys).
5. Start the development server: npm start

1. **Folder Structure** 
   * **Client**:

o  **src/components:** # Reusable components (Header, Player, etc.) o  **src/pages:** # Page components (HomePage, SearchPage, etc.) o **src/assets:** # Images, icons, and other static files o **src/redux:** # Redux store, actions, and reducers o **src/utils:** # Utility functions and helpers o **App.js:** # Main application component o **index.js:** # Entry point

* + **Utilities**:

o **api.js**: Handles API requests to the backend. o **auth.js**: Manages user authentication and token storage. o **hooks/usePlayer.js**:

Custom hook for managing the music player state.

1. **Running the Application**

**Frontend**:

* + - To start the frontend server, run the following command in the client directory: npm start
    - npm install o npx json-server ./db/db.json o npm run dev o The application will be available at http://localhost:3000

1. **Component Documentation**

**Key Components:** These are the major components of the **Fitness Tracker** application, designed to handle different aspects of user interaction and app functionality.

1. **Header**

* + - **Description:** Displays the navigation bar and search bar, allowing users to navigate through the app and search for specific exercises, routines, or articles.
    - **Props:**

▪ onSearch: A function that handles the search queries and performs filtering on the available exercises, workouts, or fitness content based on the user input.

2. **Tracker (Activity Player)**

* + - **Description:** Controls the activity tracking, including starting, pausing, and stopping a workout session.
    - **Props:**
      * + currentActivity: An object containing details of the current workout (e.g., type of workout, duration, calories burned).
        + onStart: A function that starts the workout session.
        + onPause: A function that pauses the workout session.
        + onReset: A function to reset the workout session.
        + onStop: A function to end the session and log the activity details.

3. **WorkoutCard** o **Description:** Displays a specific workout card showing the workout name, image, and some key information (e.g., duration, calories burned, etc.). o **Props:**

* + - * + workout: An object containing workout details such as name, description, image, and duration.
        + onClick: A function that handles workout selection, which might open a detailed workout view or log the workout.

**Reusable Components:** These components are designed to be flexible and used throughout the application.

1. **Button**

* + - **Description:** A customizable button component that can be used for different interactions such as submitting forms, starting workouts, or navigating through the app.
    - **Props:**
      * + text: The text that appears on the button.
        + onClick: A function to handle the button's click event.
        + disabled: A boolean value to disable the button when necessary (e.g., during a loading state or invalid form input).

2. **Input**

* + - **Description:** A reusable input field component that can be used for forms or the search bar. It handles dynamic form data and search queries.
    - **Props:**
      * + type: The type of the input (e.g., text, number, password, etc.).
        + placeholder: Placeholder text to guide the user on what to input (e.g., "Search for exercises").
        + value: The current value of the input field.
        + onChange: A function that updates the input's value in the state as the user types.

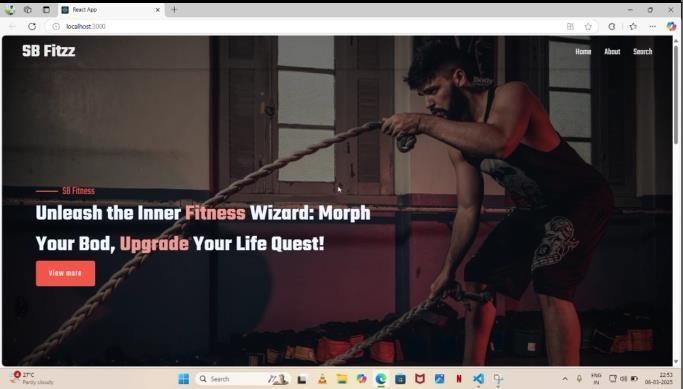
1. **State Management** 
   * **Global State**:

The Redux store in this app manages the following global states:

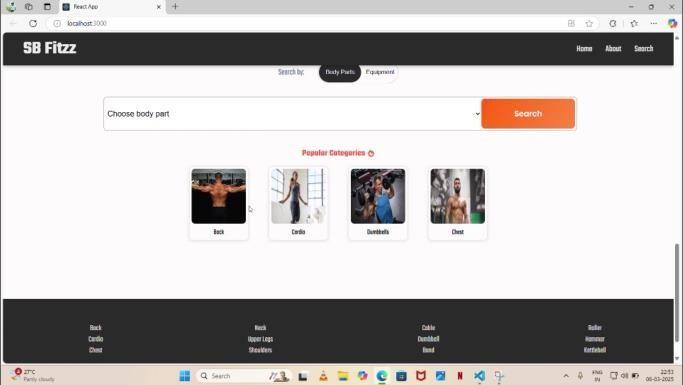
* + - **user:** Stores the information of the current authenticated user.
    - **tracker:** Tracks the current workout session or activity, including the workout being performed, its status, and the metrics like calories burned, duration, and distance.
    - **workouts:** Manages the user's workout routines or playlists, including their creation, modification, and deletion.
    - **searchResults:** Stores the results of a search query, such as exercises, workouts, or fitness articles.
  + **Local State**:

**React's useState hook** is used to manage **local state** within individual components. This is particularly useful for managing temporary data that doesn’t need to be shared across components.

1. **User Interface** 
   * **Screenshots**

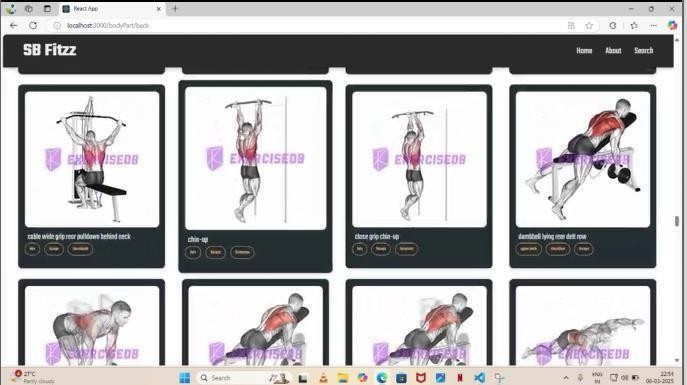


* + - * **Home Page:** Display featured tracks and recommended playlists.



**Search Page:** Allows users to search for songs, albums, and artists.

o



* + - * **Playlist Page:** Displays user-required fitness methods or workouts

1. **Styling:**

**CSS Frameworks/Libraries**:

The application uses **Styled-Components** for styling. This allows for modular and scoped CSS within components.

* + **Theming**:

A custom theme is implemented using Styled-Components, with support for light and dark modes.

1. **Testing** 
   * **Testing Strategy**:
     + **Unit Testing:** Using **Jest** and **React Testing Library**.
     + **Integration Testing**: Is performed to ensure that components work together as expected.
     + **End-to-End Testing:** **Cypress** is used for end-to-end testing of user flows.
   * **Code Coverage**:

o Code coverage is monitored using Jest’s built in coverage tool. The current coverage is 85%.

1. **Screenshots or Demo** 
   * **Demo Link:** [**https://github.com/Karthik14023/FITNESS-APP/blob/main/vid-20250311-wa0018\_7UCt4XsF.mp4**](https://github.com/Karthik14023/FITNESS-APP/blob/main/vid-20250311-wa0018_7UCt4XsF.mp4)
   * **Screenshots:** See section 9 for UI screenshots.

1. **Known Issues** 
   * + **Track Change Logic:** There may be an issue in the logic responsible for switching tracks, such as multiple triggers occurring in rapid succession, which can force a skip.
     + **Async Issues:** If the track change involves asynchronous calls (e.g., fetching track details), delays in response might cause skips.

1. **Future Enhancements** 
   * **Future Features**:

Allow users to create and update their profiles with personal details (name, profile picture, workout preferences, fitness goals, etc.).

o Add animations and transitions for a smoother user experience.

This documentation provides a comperhensive overview of the fitness tracker ( fitflex) project, including its architecture , setup instructions, and future plans.