**Frontend Development with React.js**

**1. Introduction**

* **Project Title**: **FitFlex (Fitness App)**
* **Team Leader**: DHANALAKSHMI M
* **Team Members**:
  + DEEPIKA M
  + LAVANYA A
  + MAHALAKSHMI J
  + MALATHI M

**2. Project Overview**

* **Purpose**:  
  FitFlex is a fitness app designed to help users track their workouts, set fitness goals, monitor progress, and maintain healthy habits. It aims to provide personalized workout plans, nutrition tracking, and a motivating community experience.
* **Features**:
  + Personalized workout plans based on user preferences.
  + Real-time progress tracking and data visualization.
  + Goal setting for fitness achievements (e.g., weight loss, strength gain).
  + Integration with wearable devices for activity tracking.
  + Community features (workout challenges, leaderboards).
  + Nutrition tracking with calorie and macro breakdowns.
  + Daily fitness tips and motivational content.

**3. Architecture**

* **Component Structure**:  
  The FitFlex app is divided into several major components:
  + **HomePage**: Displays daily workout recommendations, progress, and motivational content.
  + **WorkoutPage**: Shows a list of exercises with detailed instructions and videos.
  + **ProgressPage**: Visualizes user progress with charts and statistics.
  + **ProfilePage**: Allows users to update personal information, goals, and preferences.
  + **CommunityPage**: Displays leaderboards, challenges, and social interactions.
* **State Management**:  
  We use **Redux** for managing global application state, ensuring a smooth data flow across the app. Local state is managed with React's useState and useEffect hooks.
* **Routing**:  
  We use **React Router** for navigation, providing smooth transitions between pages such as Home, Workout, Progress, and Profile.

**4. Setup Instructions**

* **Prerequisites**:
  + Node.js (v16 or higher)
  + npm (v8 or higher)
* **Installation**:  
  Follow the steps below to get the project running locally:
  + Clone the repository:
  + git clone https://github.com/your-repo/fitflex.git
  + Navigate to the project directory:
  + cd fitflex
  + Install dependencies:
  + npm install
  + Set up environment variables (e.g., API keys, database URL) in a .env file at the root of the project.
  + Start the development server:
  + npm start

**5. Folder Structure**

* **Client**: The main React application is inside the client folder, structured as follows:
  + **components/**: Contains reusable UI components like buttons, cards, etc.
  + **pages/**: Contains page-level components like HomePage, WorkoutPage, etc.
  + **assets/**: Stores images, icons, and other static assets.
  + **hooks/**: Custom React hooks for common functionality (e.g., useAuth, useFetch).
* **Utilities**: Includes helper functions for date formatting, API requests, etc. Example:
  + **api.js**: Helper functions to interact with backend APIs (e.g., fetchWorkoutPlans).
  + **helpers.js**: Utility functions for data manipulation.

**6. Running the Application**

* To start the frontend server locally, run:
* npm start

**7. Component Documentation**

* **Key Components**:
  + **HomePage**: Displays daily recommendations and progress overview.
    - Props: userData, workoutPlans
  + **WorkoutPage**: Displays detailed workout information with video demonstrations.
    - Props: workoutDetails, onCompleteWorkout
  + **ProfilePage**: Allows users to view and update personal data and goals.
    - Props: userProfileData, onUpdateProfile
* **Reusable Components**:
  + **Button**: A customizable button used throughout the app.
    - Props: label, onClick, style

**8. State Management**

* **Global State**:
  + Redux is used for managing the application state, such as user data, workout plans, and progress statistics.
  + Actions like SET\_USER, FETCH\_WORKOUTS, and UPDATE\_PROGRESS are dispatched across the app to modify the global state.
* **Local State**:
  + Local states like form inputs and temporary data are managed using React's useState hook in individual components.

**9. User Interface**

* Screenshots or GIFs of UI features:
  + HomePage: Displaying daily workout and progress.
  + WorkoutPage: Visuals of different exercises with instructions.
  + ProgressPage: Charts showing calories burned, workout performance, etc.

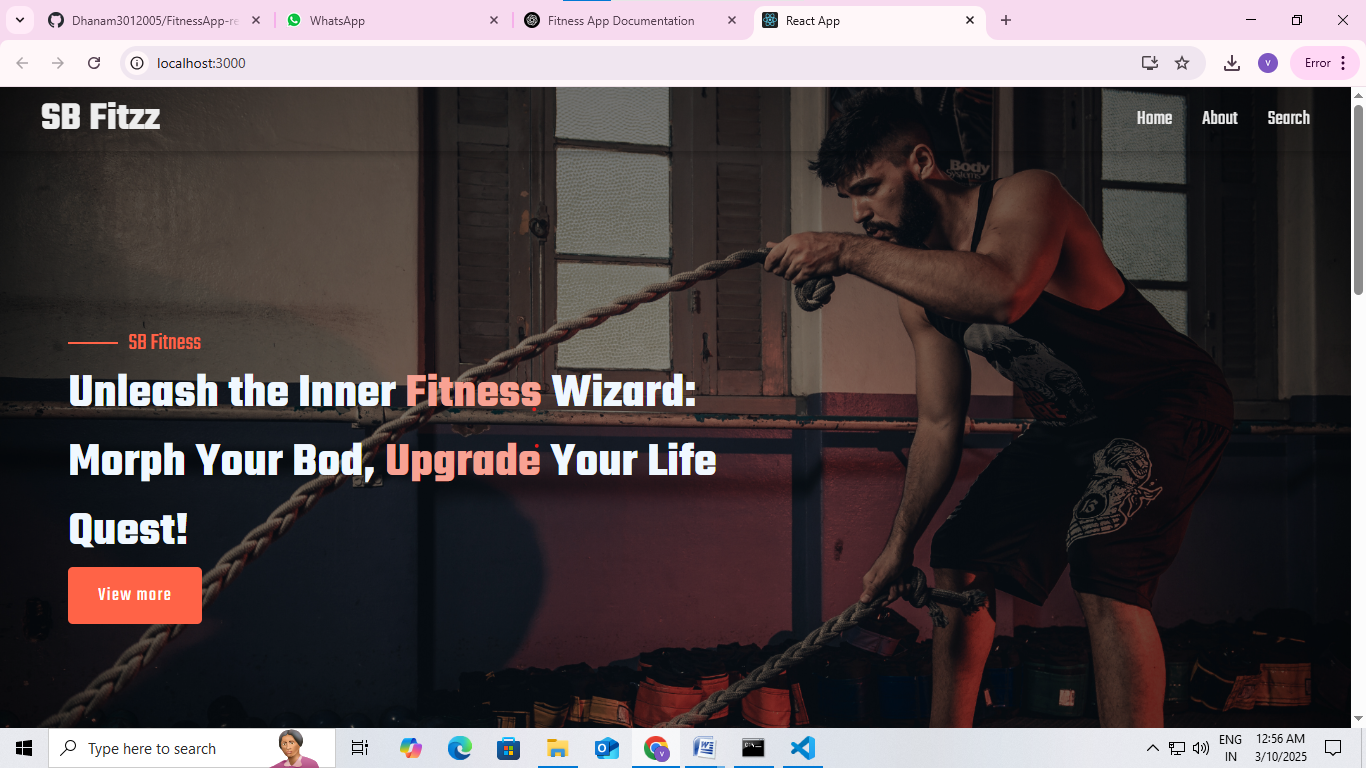
**10. Styling**

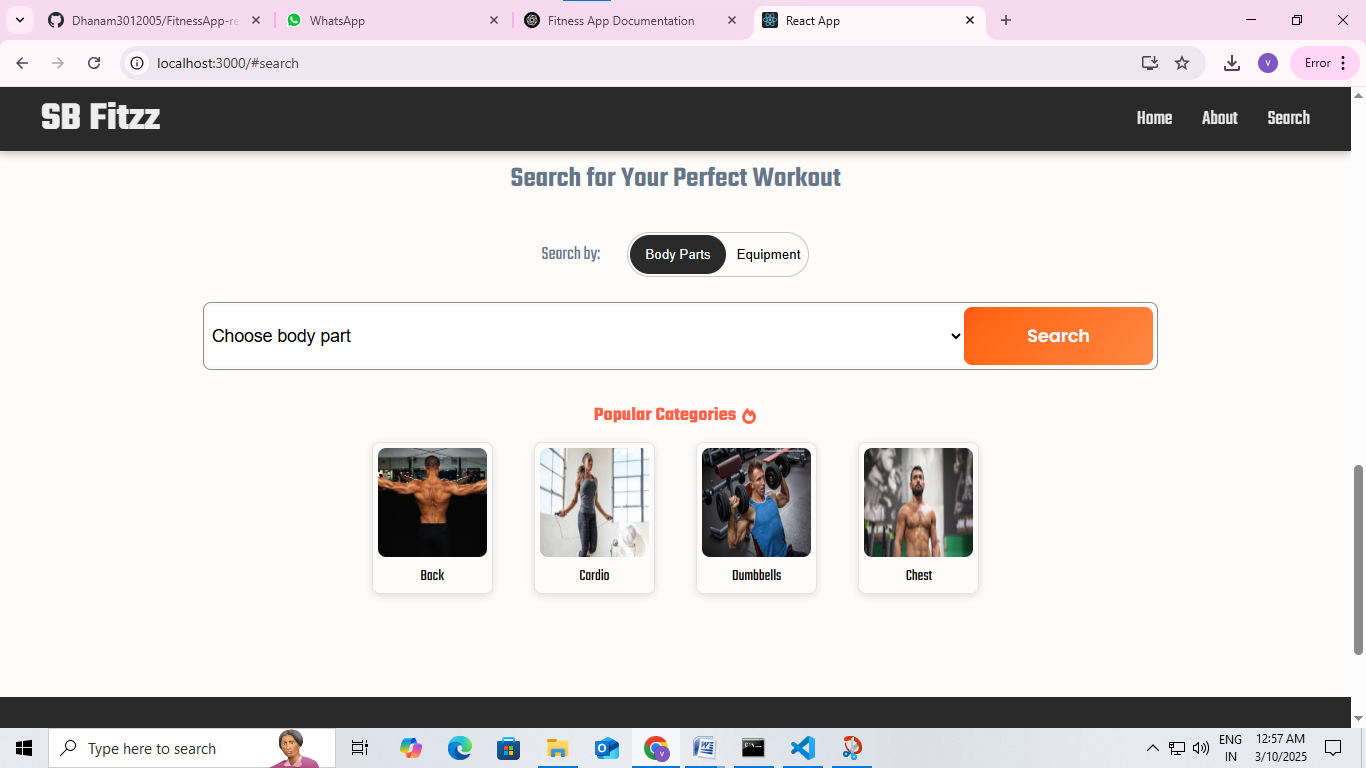
* **CSS Frameworks/Libraries**:
  + We use **Styled-Components** for component-level styling, allowing for more modular and dynamic styles.
  + **CSS Grid** and **Flexbox** are used to create responsive layouts.
* **Theming**:
  + The app has a custom theme with light and dark modes, allowing users to toggle between the two based on their preference.

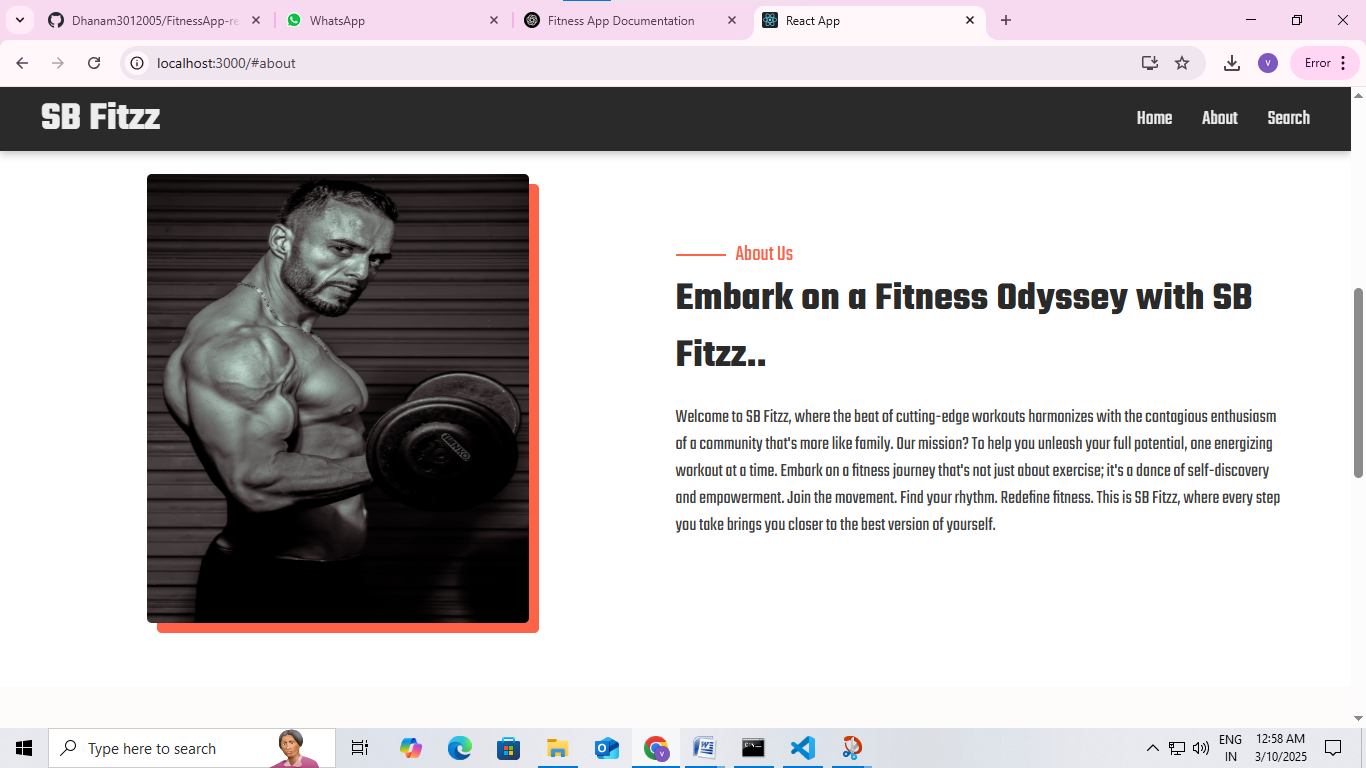
**11. Testing**

* **Testing Strategy**:  
  We use **Jest** and **React Testing Library** to test the components and ensure correctness.
  + **Unit tests** are written for small, isolated components.
  + **Integration tests** ensure components work together properly (e.g., Form submission).
  + **End-to-End (E2E) tests** ensure the app works as expected from the user's perspective.
* **Code Coverage**:  
  We use **Jest** with coverage reporting enabled to ensure all critical paths are tested.

**12. Screenshots or Demo**

* **Demo Link**: [Link to live demo](https://chatgpt.com/)
* Screenshots of the app interface showcasing key features such as the home screen, workout tracking, and progress charts.





**13. Known Issues**

* **Bug 1**: The workout plan sometimes fails to load after app restart (being investigated).
* **Bug 2**: Notifications for workout reminders are delayed.

**14. Future Enhancements**

* Integration with more wearable devices (e.g., Fitbit, Garmin).
* Enhanced workout animations and interactive exercise tutorials.
* Social features for users to share progress and compete in challenges.