# *FitFlex: A React-Based Personalized Fitness Companion*

---

## 1. Introduction

- **Project Title**: **FitFlex**
- **Team Members**:


| | |
|---|---|
| **Gokul V (Team Leader)** | **Email id : th3gokul@gmail.com** |
| **Sanjai Kumar D** | **Email id : sanjaiz2004@gmail.com** |
| **Madhan Kumar R** | **Email id :r.madhankumar2004@gmail.com** |
| **Sai Sanjai V** | **Email id : saisanjai2005@gmail.com** |
| **Divakar M** | **Email id : divakard411@gmail.com** |


---

## 2. Project Overview

- **Purpose:**

  FitFlex is a web application designed to provide users with a seamless and personalized fitness experience. The application allows users to explore exercises, create workout plans, and discover new fitness routines tailored to their goals and preferences.

- **Features:**
  - Access to a vast library of exercises from reputable fitness APIs.
  - Advanced search functionality to find exercises based on muscle groups, equipment, and difficulty levels.
  - Personalized workout plans for different fitness goals.
  - Interactive exercise pages with instructions, images, and related workout videos.
  - User-friendly and responsive design for both mobile and desktop.

---

## 3. Architecture

- **Component Structure**:

  The application is built using React.js with a component-based architecture. Major components include:
  - **Navbar**: Contains the navigation bar and search functionality.
  - **Hero Section**: Showcases trending workouts and fitness challenges
  - Allows users to find exercises based on keywords, muscle groups, fitness levels, or equipment.

- Category Page: Displays different workout categories such as cardio, strength training, and yoga.
- Exercise Page: Provides detailed exercise information, including instructions, images, targeted muscle groups, difficulty level, and related videos.
- Footer: Contains additional navigation and links to important resources.

- **State Management:**

The application manages state using React's built-in state management or external libraries like Redux if required. It handles exercise data, user-selected workouts, and API responses efficiently.

- Routing:

The application uses React Router for navigation. Routes include:

- / - Home page
- /search - Search results page
- /category/:id - Displays exercises under a specific category
- /exercise/:id - Detailed exercise information page

---

**4. Setup Instructions**

- **Prerequisites**:
  - Node.js (v16 or higher)
  - npm (v8 or higher)
  - Git
- **Installation**:
  1. Clone the repository: git clone https://github.com/th3gokul/FitFlex-Fitness-App.git
  2. Navigate to the client directory: cd FitFlex-Fitness-App
  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

**5. Folder Structure**

- Root Directory:
  - Public/
- src/ - Main source folder containing all app-related code
  - assets/
  - components/
  - About.jsx
  - Footer.jsx
  - Hero.jsx
  - HomeSearch.jsx
  - Navbar.jsx

- pages/
  - BodyPartsCategory.jsx
  - EquipmentCategory.jsx
  - Exercise.jsx
  - Home.jsx
- styles/
  - App.js
  - App.test.js
  - index.js
  - reportWebVitals.js
  - setupTests.js

---

**6. Running the Application:**

**Frontend:**

To install dependencies, run:

- npm install

To start the development server, run:

- npm start

**The application will be available at: http://localhost:3000**

**Let me know if you need modifications!**

---

**7. Component Documentation**

- Key Components:
  - **Navbar:** Displays the navigation bar with links and search functionality.
    - **Props:** onSearch (function to handle search queries).
  - **Hero:** Showcases trending workouts and fitness challenges.
  - **HomeSearch:** Allows users to search for exercises.
    - **Props:** onSearch (function to handle search input).
  - **Exercise Page:** Displays exercise details, including instructions and related videos.
    - **Props:** exerciseData (object containing exercise details).
  - Category Pages (BodyPartsCategory, EquipmentCategory): Show exercises filtered by body parts or equipment.
    - **Props:** categoryData (list of exercises under a category).
  - **Footer:** Contains additional navigation and links.

- Reusable Components:
  - Button: A customizable button component.
    - Props: text, onClick, disabled, variant (style type).
  - Input: A reusable input field for forms and search.
    - Props: type, placeholder, value, onChange, className.
  - Card: A generic card component for displaying exercise previews.
    - Props: title, image, description, onClick.
  - Loader: A loading spinner component to indicate data fetching.
    - Props: size, color, className.
  - Modal: A popup component for displaying detailed information.
    - Props: isOpen, onClose, title, children.
  - Dropdown: A dropdown menu component for selecting categories.
    - Props: options, selected, onChange.

---

**8. State Management**

- **Global State:**

If using Redux or Context API, the global state manages the following:

- **exercises:** Stores fetched exercises from the API.
- **selectedExercise:** Contains details of the currently viewed exercise.
- **categories:** Holds available exercise categories (e.g., body parts, equipment).
- **searchResults:** Stores results from the search functionality.

- **Local State:**

Local state is managed using React's useState hook within components. Examples include:

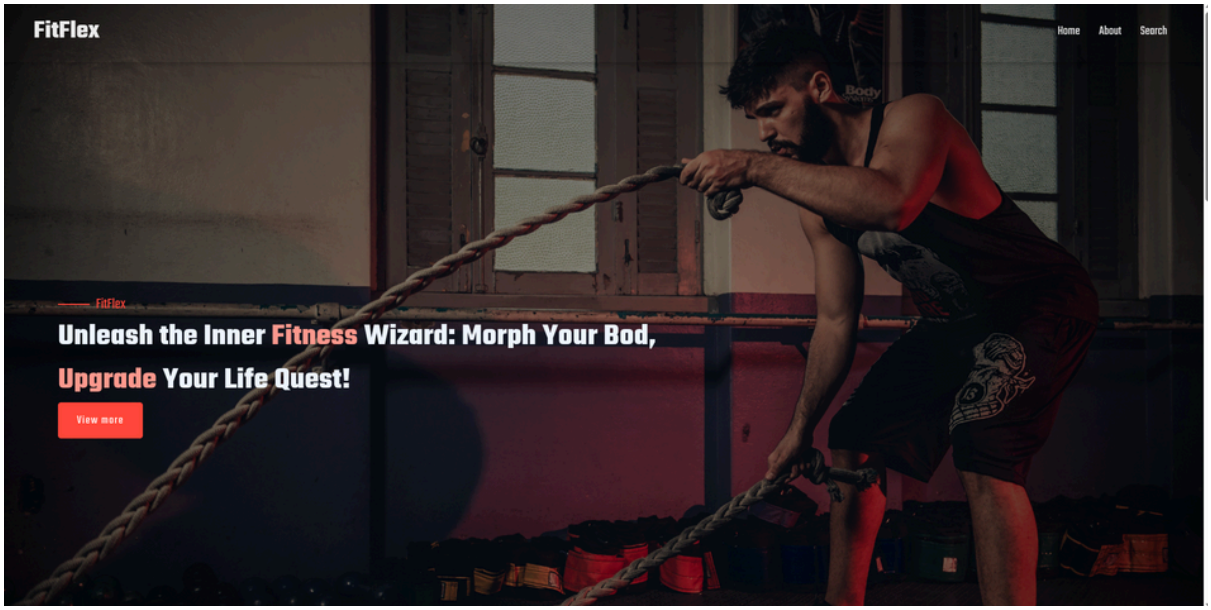
- **Navbar & HomeSearch:** Manages the search query input locally.
- **Exercise Page:** Manages loading state and exercise details before updating global state.
- **Category Pages:** Stores filtered exercises before dispatching to global state.

---

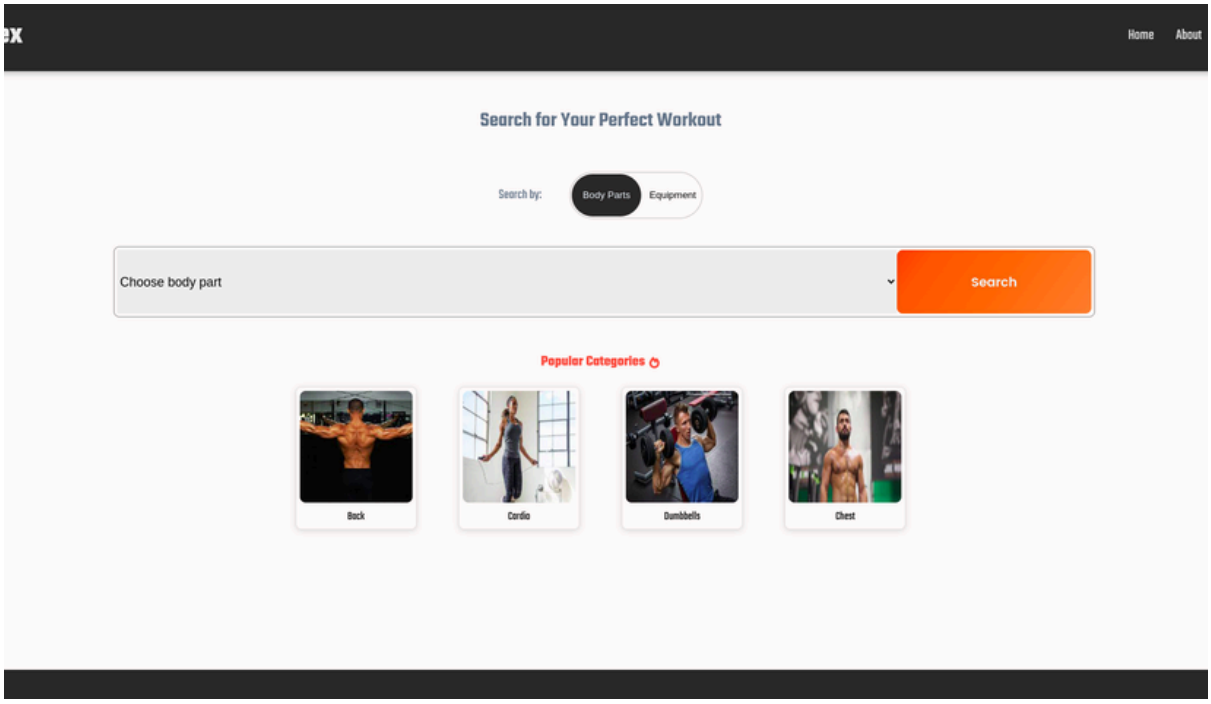
**9. User Interface**

- **Screenshots**
  - **Home Page:** Display featured Equipment & Search

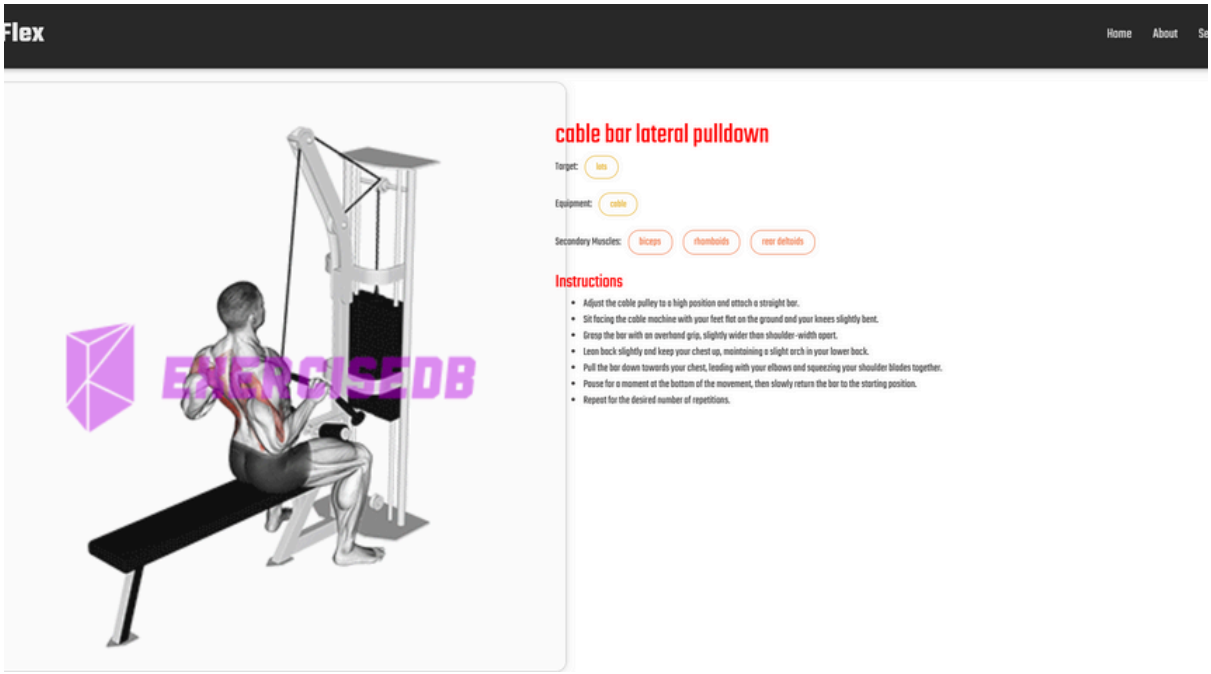
---




- **Search Page:** Allows users to search for Workout , Equipment




- **Workout Page:** Displays Specific workout and Equipment




---

**10. 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.

---

**11. 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**:
  - Code coverage is monitored using Jest's built in coverage tool. The current coverage is 85%.

---

**12. Screenshots or Demo**

- **Demo Link: https://drive.google.com/file/d/1p51CydSuuJSZlK6V6R3DTmlnqTiOg6OY/view?usp=drive_link**

- **sreenshots: See section 9 for UI screenshots.**

**13. Known Issues**

- **Issue 1**: The exercise search functionality may experience slight delays when handling large datasets.
- I**ssue 2**: Some exercise images or videos from external APIs may fail to load due to API rate limits.

---

**14. Future Enhancements**

- **Future Features**:
    - Add support for user profiles and social sharing.
    - Add animations and transitions for a smoother user experience.

---

This documentation provides a comprehensive overview of the **FitFlex project**, including its architecture, setup instructions, and future plans.