**FRONTEND DEVELOPMENT WITH REACT.JS**

**PROJECT DOCUMENTATION**

**INTRODUCTION**

* **Project Title: FITFLEX: Your Personal Fitness Companion**
* **TeamMembers:RAJESHWARI.P(TEAMLEADER)**

**(** [**rajeshwari15072005@gmail.com**](mailto:rajeshwari15072005@gmail.com) **)**

* **POOJA .S.K(** [**poojanirmala18@gmail.com**](mailto:poojanirmala18@gmail.com) **)**
* **SHAMILI.S(** [**shamilistalin0609@gmail.com**](mailto:shamilistalin0609@gmail.com) **)**
* **HEMA SRI.V(** [**vhemasri2005@gmail.com**](mailto:vhemasri2005@gmail.com) **)**
* **Project Overview:**

FitFlex is an innovative fitness application designed to revolutionize the workout experience for users of all fitness levels. The app provides an intuitive, user-friendly interface, dynamic search functionality, and access to a vast library of exercises from the ExerciseDB API. FitFlex aims to foster a vibrant fitness community by enabling users to discover, save, and share personalized workout routines, encouraging collaboration and engagement in pursuit of a healthier lifestyle.

* **Purpose:**  
  The primary purpose of FitFlex is to offer an accessible platform for individuals passionate about fitness, exercise, and holistic well-being. The app leverages modern web development technologies, particularly React.js, to deliver an efficient and enjoyable user experience.
* **Features:** 
  + Access to diverse exercises from the ExerciseDB API, covering various workout categories and fitness goals.
  + Visual exploration of workout routines through curated image galleries and GIFs.
  + Intuitive and user-friendly design for seamless navigation and exercise selection.
  + Advanced search functionality to find specific exercises or workout plans.
  + Future enhancements, such as community features, advanced routing, and multimedia integration (e.g., YouTube videos).

**Architecture**

* **Component Structure:**
  + FitFlex is built using React.js, with a modular component-based architecture. Major components include:
  + App.js: Main component responsible for fetching and rendering exercises.
  + Navbar: Header component displaying the "FitFlex Train Hard" logo.
  + Hero: Showcases trending workouts with the tagline "Transform Your Body" and a background image.
  + Categories: Displays exercises under specific categories (e.g., by target muscle or equipment).
  + Exercise Page: Shows detailed instructions, GIFs, and related videos for selected exercises.
  + Search: Implements real-time search functionality for exercises.  
    Components interact through props and state, ensuring a smooth flow of data across the application.
* **State Management:** 
  + Global State: Managed using React'suseState and useEffect hooks, with data fetched from APIs (e.g., ExerciseDB, YouTube) stored in global state for access across components.
  + Local State: Individual components manage local state (e.g., form inputs, toggles) using React hooks.
  + Future Consideration: Redux or Context API may be implemented for more complex state management needs.
* **Routing:** 
  + Currently, FitFlex uses basic navigation without routing.
  + Future Implementation: React Router will be integrated to enable navigation between pages (e.g., home, categories, exercise details).

**Setup Instructions**

* **Prerequisites:** 
  + Node.js and npm: Required for running JavaScript and managing dependencies. Download from <https://nodejs.org/en/download/>.
  + React.js: JavaScript library for building user interfaces.
  + Git: For version control. Download from <https://git-scm.com/downloads>.
  + Development Environment: Recommended editors include Visual Studio Code, Sublime Text, or WebStorm.
  + Basic Knowledge: HTML, CSS, and JavaScript for frontend development.
* **Installation:** 
  + **Get the Code:** 
    - Download the project from the Google Drive link: <https://drive.google.com/drive/folders/14f9eBO5WTVrd4PhP2W6PzOUHCV8UMex?usp=sharing>.
  + **Install Dependencies:** 
    - Navigate to the project directory:

cd fitflex

* + - Install libraries:

npm install

* + **Configure Environment Variables:** 
    - Create a .env file in the root directory.
    - Add your RapidAPI key:

REACT\_APP\_API\_KEY=your-api-key-here

* + **Start the Development Server:** 
    - Run:

npm start

* + - Access the app at http://localhost:3000.

**Folder Structure**

* **Client:** 
  + public/:
    - index.html: Entry point with meta tags and Cloudflare script for security.
    - manifest.json: Configures Progressive Web App (PWA) settings with icons and theme colors.
    - robots.txt: Allows web crawlers to index the site.
  + src/:
    - App.js: Main component handling exercise fetching and rendering.
    - App.css: Custom styles for the app’s layout and components.
    - index.js: Renders the app into the DOM.
    - index.css: Global styles with Poppins font settings.
    - reportWebVitals.js: Performance monitoring with web-vitals.
    - setupTests.js: Configures Jest for testing.
    - App.test.js: Unit test file (currently outdated, referencing "learn react").
* **Utilities:** 
  + Custom hooks (e.g., for fetching API data) and helper functions are implemented in the src/utils/ directory (future enhancement).

**Running the Application**

* Frontend:
  + Navigate to the project directory:

cd fitflex

* + Start the development server:

npm start

* + Access the app at http://localhost:3000.

**Component Documentation**

* **Key Components:** 
  + App.js:
    - Purpose: Fetches and renders exercises from the ExerciseDB API.
    - Props: None (currently).
  + Navbar:
    - Purpose: Displays the "FitFlex Train Hard" logo with navigation links (future enhancement).
    - Props: None.
  + Hero:
    - Purpose: Showcases trending workouts with the tagline "Transform Your Body".
    - Props: Background image URL (future enhancement).
  + Categories:
    - Purpose: Displays exercises under specific categories.
    - Props: categoryId, exercises.
  + Exercise Page:
    - Purpose: Shows detailed exercise instructions, GIFs, and related videos.
    - Props: exerciseId, exerciseDetails.
* **Reusable Components:** 
  + SearchBar:
    - Purpose: Implements real-time search for exercises.
    - Props: onSearch, searchQuery.
  + ExerciseCard:
    - Purpose: Displays exercise details in a card format.
    - Props: exerciseName, exerciseImage, exerciseDescription.

**State Management**

* Global State:
  + Managed using React hooks (useState, useEffect).
  + Stores data fetched from APIs, such as body parts, equipment, exercises, and related videos.
  + Example:

const [bodyParts, setBodyParts] = useState([]);

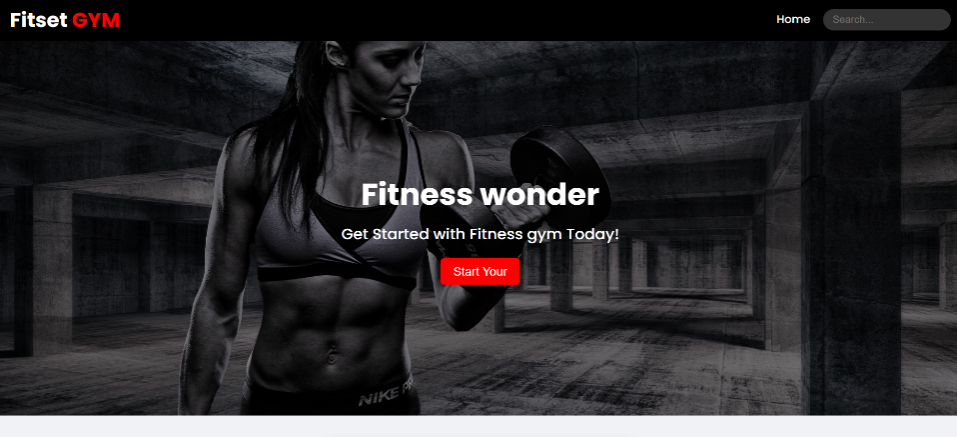
const [exercises, setExercises] = useState([]);

* Local State:
  + Managed within individual components (e.g., form inputs, toggles).
  + Example:

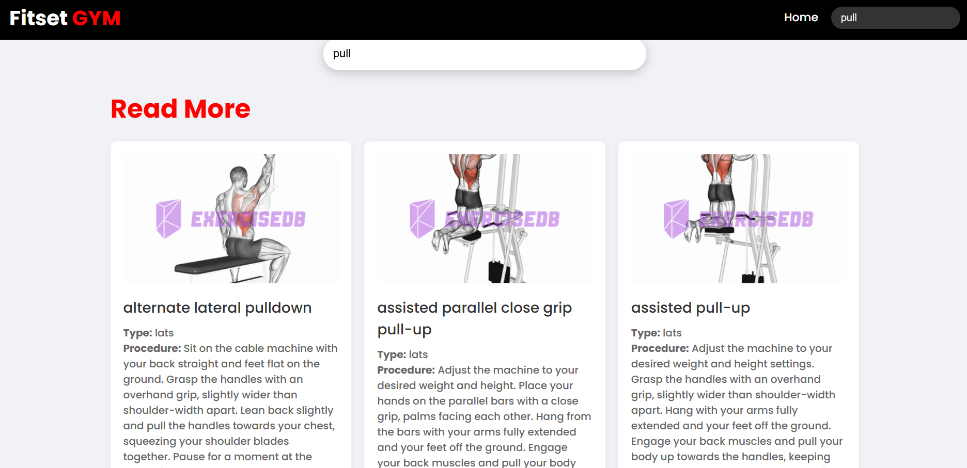
const [searchQuery, setSearchQuery] = useState("");

**User Interface**

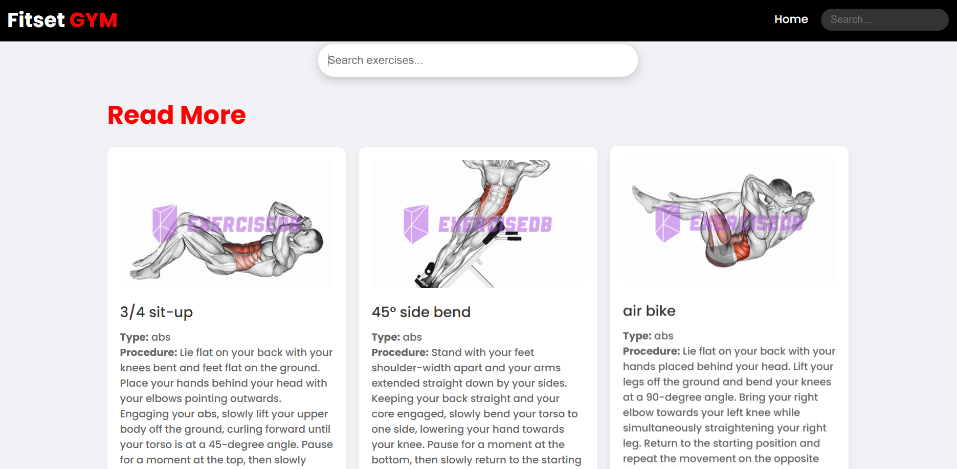
* Screenshots or GIFs:
  + Hero Component: Features "Transform Your Body" with a background image showcasing trending workouts.



* + Categories Page: Displays exercises under specific categories (e.g., by target muscle or equipment)



* + Exercise Page: Shows detailed instructions, GIFs, and related videos (future enhancement).



* Demo Link:
  + <https://drive.google.com/file/d/1IsiDrlcftIDbexBoD2rncj4cvggU__7Z/view?usp=drive_link>

**Styling**

* CSS Frameworks/Libraries:
  + Custom CSS with the Poppins font for typography.
  + No external CSS frameworks currently used.
* Theming:
  + Global styles defined in src/index.css.
  + Future enhancements may include a custom design system with light/dark mode support.

**Testing**

* Testing Strategy:
  + Unit Testing: Test individual components using Jest and React Testing Library.
  + Integration Testing: Test interactions between components (future enhancement).
  + End-to-End Testing: Simulate user flows using tools like Cypress (future enhancement).
* Code Coverage:
  + Use Jest's built-in coverage reports to ensure adequate test coverage.
  + Example command:

npm test -- --coverage

* + Currently, testing is limited (e.g., outdated App.test.js).

**Known Issues**

* Testing suite is outdated (e.g., App.test.js references "learn react").
* Limited error handling for API failures.
* Routing is not implemented, limiting navigation.
* Placeholder content for future features (e.g., newsletter, footer, YouTube videos).

**Future Enhancements**

* Implement React Router for advanced navigation.
* Integrate YouTube API for related exercise videos.
* Develop community features (e.g., sharing workouts, user profiles).
* Enhance styling with a custom design system and theming.
* Improve testing coverage and error handling.
* Add newsletter subscription and footer components.

**CONCLUSION**

FitFlex represents a significant step forward in personal fitness technology, blending a user-friendly interface with powerful API integration to deliver a transformative workout experience. By leveraging React.js and the ExerciseDB API, FitFlex empowers users of all levels to explore, customize, and enjoy their fitness journeys. As we look to the future, FitFlex is poised for growth with potential enhancements like advanced routing, community features, and multimedia integration, promising to remain a trusted companion in the pursuit of a healthier lifestyle.