**Frontend Development with React.js**

**Project Documentation format**

1. **Introduction**
   * **Project Title**: [Fitflex]
   * **Team Members**:
   * **Ganapthi** P
   * Gokulkanth S
   * Gopi K
   * Guna M
   * Gunaseelan R.
2. **Project Overview**
   * **Purpose**: Fitflex aims to provide an easy-to-use platform for tracking workouts and fitness goals through a responsive and engaging frontend.
   * **Features**:
   * User dashboard for progress overview
   * Workout plan creation and management
   * Exercise library with guides
   * Progress tracking with charts
   * Responsive and interactive design.
   * **Architecture:** React-based modular frontend with reusable components and centralized state
   * **Component Structure**:
   * App (root & routes)
   * Layout (header, sidebar, footer)
   * Dashboard, WorkoutPlans, ExerciseLibrary, Profile.
   * **State Management**: Uses Context API with hooks for global and local state.
   * **Routing**: React Router for main pages and nested routes for details.
   * **Setup Instructions**
   * **Prerequisites**:
   * Prerequisites:
   * Node.js (v16+)
   * npm or yarn
   * Git
   * **Installation**:
   * 1. Clone repo: git clone <repo-url>
   * 2. Go to project folder: cd fitflex
   * 3. Install dependencies: npm install or yarn install
   * 4. Create .env file and add environment variables
   * 5. Start app: npm start
   * **Folder Structure**
   * **Client**:
   * /components - Reusable UI
   * /pages - Dashboard, Plans, Profile
   * /assets - Images, styles
   * /context - Global state
   * /routes - App routing
   * App.js – Root
   * **Utilities**:
   * /utils - Helper functions
   * /hooks - Custom hooks
   * **Running the Application**
   * Provide commands to start the frontend server locally.
     + **Frontend**:
     + cd client
     + npm start

°**Component Documentation**

* + **Key Components**:
  + Dashboard - user progress (userData)
  + WorkoutPlans - routines (plans, onUpdate)
  + ExerciseLibrary - exercises (exercises)
  + Profile - settings (user)
  + **Reusable Components**:
  + Button (label, onClick, type)
  + Card (title, children)
  + Navbar (links)
  + **State Management**
  + **Global State**: Managed with Context API for user data, workouts, and authentication. Shared across pages via context provider.
  + **Local State**: Handled with useState and useEffect inside components for Ul interactions (e.g., form inputs, modal toggles)
  + **User Interface**

Dashboard (progress & stats)

Workout Plans (create/manage routines)

Exercise Library (browse exercises)

Profile (user settings)

Forms & interactions (login, inputs, modals)

**Styling**

* **CSS Frameworks/Libraries**: Used CSS Modules and Tailwind CSS for responsive, utility-first styling.
* **Theming**: Custom theme with consistent colors, fonts, and reusable style classes across components.
* **Testing**
* **Testing Strategy**: Used Jest and React Testing Library for unit and integration tests; basic end-to-end flows with Cypress.
* **Code Coverage**: Maintained via Jest coverage reports to ensure critical components and functions are tested.
* **Screenshots or Demo**
* Provide screenshots or a link to a demo showcasing the application’s features and design.

1. **Known Issues**

* Document any known bugs or issues that users or developers should be aware of.

1. **Future Enhancements**

Add social/community features

Advanced analytics & Al-driven workout suggestions

Offline mode for workout tracking

Dark/light theme toggle

More animations & enhanced styling