**INTRODUCTION**

**Project Title:** fitflex: your personal fitness companion

**Team Members:**S.Sathiyapriya

**E-mail id:**priya135lucky@gmail.com

**Name**:G.Pooja

**E-mail id**:poojasadha032@gmail.com

**Name:**S.Kaviya

**E-mail id:**suneelmca325@gmail.com

**Name:**T.Jayashree

**E-mail id:**jeni24062005@gmail.com

**Name:**M.Mareshwari

**E-mail id:**hseram2107@gmail.com

1. **Project Overview**

**Purpose:**

Provide a clear and concise description of the project's purpose, goals, and expected outcomes.

Explain what problem the application is solving or what functionality it aims to provide.

**Features:**

Highlight the key features and functionalities of the frontend, including but not limited to:

* + User authentication and authorization
  + Responsive UI/UX design
  + API integrations
  + Data visualization
  + Forms and validations
  + State management

1. **Architecture**

**Component Structure:**

Outline the major React components used in the project, how they are structured, and how they interact. You can use diagrams to represent the component hierarchy.

**State Management:**

Describe the approach used for managing state in the application, such as:

* + **React's built-in state** (useState, useReducer)
  + **Context API** for global state management
  + **Redux** (with Redux Toolkit if applicable)
  + **Recoil, Zustand, or other state management libraries** Explain how state is structured and shared between components.

**Routing:**

If using React Router or another routing library, explain:

* + The routing structure
  + Nested routes
  + Route parameters and query strings
  + Protected routes (authentication-based routing) **4. Setup Instructions**

**Prerequisites:**

List all necessary software dependencies required to run the project, such as:

* + Node.js (specify version)
  + npm or yarn
  + Any required backend service or API keys
  + Database setup if applicable

**Installation:**

Provide step-by-step installation instructions:

* 1. Clone the repository: git clone https://github.com/your-repo/project-name.git
  2. Navigate into the project directory: cd project-name
  3. Install dependencies:

npm install # or yarn install

* 1. Configure environment variables in a .env file.
  2. Run the development server:

npm start # or yarn start

**5. Folder Structure**

Explain the organization of the React application, including folders such as: project-name/

├── src/

│ ├── components/ # Reusable components

│ ├── pages/ # Page components

│ ├── assets/ # Images, fonts, etc.

│ ├── hooks/ # Custom hooks

│ ├── context/ # Context API setup

│ ├── utils/ # Helper functions

│ ├── styles/ # Global styles

│ ├── App.js # Root component

│ ├── index.js # Entry point

│ ├── routes.js # Routing configuration

│ └── services/ # API calls and integrations

└── package.json # Dependencies and scripts

**6. Running the Application**

Provide commands to start the frontend server locally:

npm start # or yarn start

For production build:

npm run build # or yarn build **7. Component Documentation**

**Key Components:**

Document the major components used in the project, their purpose, and the props they receive.

Example:

const Button = ({ label, onClick }) => { return <button onClick={onClick}>{label}</button>;

};

Props:

* label: The text displayed on the button.
* onClick: Function executed when the button is clicked.

**Reusable Components:**

Detail reusable UI components like modals, buttons, inputs, and form elements. Explain customization options available through props.

1. **State Management**

**Global State:**

Explain how global state is managed across the application, including shared data and actions.

**Local State:**

Discuss how individual components manage their own local states and how they interact with the global state.

1. **User Interface**

Include screenshots or GIFs showcasing different UI features, such as:

* + Home page
  + Login/Signup page
  + Dashboard
  + Form submissions
  + Interactive elements

1. **Styling**

**CSS Frameworks/Libraries:**

List any CSS frameworks or styling libraries used, such as:

* + Tailwind CSS
  + Bootstrap
  + Styled-Components
  + SCSS/SASS

**Theming:**

Explain how themes or custom design systems are implemented, such as dark/light mode toggle.

1. **Testing**

**Testing Strategy:**

Describe the approach for testing the application, including:

* + **Unit testing** (e.g., Jest, React Testing Library)
  + **Integration testing**
  + **End-to-end testing** (e.g., Cypress, Playwright)

**Code Coverage:**

Explain tools or techniques used for ensuring adequate test coverage and code quality.

1. **Screenshots or Demo**

Provide images or links to a live demo of the application. Include:

* + Deployed application link
  + Walkthrough video (if available)

1. **Known Issues**

List any known bugs, limitations, or issues that users or developers should be aware of.

1. **Future Enhancements**

Outline potential improvements or features planned for future development, such as:

* + Adding animations or micro-interactions
  + Implementing better state management
  + Improving performance and optimization
  + Expanding test coverage