# **Front-end Development with React.js**

# **Project Documentation format**

# **Introduction**

# **Project Title: [Fitflex]**

# **Team Members: Muthuvalli Ramya pooja sanjitha**

# **Project Overview**

# **Purpose: FitFlex is a brand or concept typically associated with fitness and wellness, though its specific purpose can vary depending on context.**

# **Architecture**

# **Component Structure:**

# **App.js: Entry point of the application, handles routing and initialization.**

# **A wrapper component that includes the overall structure (header, footer, sidebar, etc.).**

# **Handles user login functionality.**

# **State Management: Describe the state management approach used (e.g., Context API, Redux).**

# **Routing: Explain the routing structure if using react-router or another routing library.**

# **Setup Instructions**

# **Setup Frontend (React)**

# **Navigate to the frontend directory (if the project is organized in separate frontend/backend folders) and install the necessary dependencies.**

# **Configure Environment Variables: a .env file in the backend directory with your API keys, database URLs, JWT secrets, etc. For example:**

# **Setup the Database: If you're using MongoDB:**

# **Ensure MongoDB is installed and running on your machine or connect to a cloud-based MongoDB instance like MongoDB Atlas.**

# **If using PostgreSQL, ensure you have it installed and set up the required tables (you can use Sequelize or TypeORM for interacting with PostgreSQL).**

# **RunEnsure that the frontend React app is connected to the backend API by verifying the URL in the .env file. The backend server should be running on port 5000 or whatever port you've configured, and the frontend should point to that API.**

# **Running the Full Application**

# **With both frontend and backend running, you should now be able to access the full application.**

# **Bottom of Form**

# **Folder Structure**

# **Client: Describe the organization of the React application, including folders like components, pages, assets, etc.**

# **Utilities: Explain any helper functions, utility classes, or custom hooks used in the project.**

# **Running the Application**

# **Provide commands to start the frontend server locally.**

# **Frontend: npm start in the client directory.**

# **Component Documentation**

# **Key Components: Document major components, their purpose, and any props they receive.**

# **Reusable Components: Detail any reusable components and their configurations.**

# **State Management**

# **Global State: Describe global state management and how state flows across the application.**

# **Local State: Explain the handling of local states within components.**

# **User Interface**

# **Provide screenshots or GIFs showcasing different UI features, such as pages, forms, or interactions.**

# **Styling**

# **CSS Frameworks/Libraries: Describe any CSS frameworks, libraries, or pre-processors (e.g., Sass, Styled-Components) used.**

# **Theming: Explain if theming or custom design systems are implemented.**

# **Project Documentation format**

# **Introduction**

# **Project Title**: [Fitflex]

# **Team Members**: List team members and their roles.

# **Project Overview**

# **Purpose**: Briefly describe the purpose and goals of the project.

# **Features**: Highlight the key features and functionalities of the frontend.

# **Architecture**

# **Component Structure**: Outline the structure of major React components and how they interact.

# **State Management**: Describe the state management approach used (e.g., Context API, Redux).

# **Routing**: Explain the routing structure if using react-router or another routing library.

# **Setup Instructions**

# **Prerequisites**: List software dependencies (e.g., Node.js).

# **Installation**: Provide a step-by-step guide to clone the repository, install dependencies, and configure environment variables.

# **Folder Structure**

# **Client**: Describe the organization of the React application, including folders like components, pages, assets, etc.

# **Utilities**: Explain any helper functions, utility classes, or custom hooks used in the project.

# **Running the Application**

# Provide commands to start the frontend server locally.

# **Frontend**: npm start in the client directory.

# **Component Documentation**

# **Key Components**: Document major components, their purpose, and any props they receive.

# **Reusable Components**: Detail any reusable components and their configurations.

# **State Management**

# **Global State**: Describe global state management and how state flows across the application.

# **Local State**: Explain the handling of local states within components.

# **User Interface**

# Provide screenshots or GIFs showcasing different UI features, such as pages, forms, or interactions.

# **Styling**

# **CSS Frameworks/Libraries**: Describe any CSS frameworks, libraries, or pre-processors (e.g., Sass, Styled-Components) used.

# **Theming**: Explain if theming or custom design systems are implemented.