**PHASE-WISE DEVELOPMENT PLAN FOR FIT FLEX FITNESS APP**

**FRONTEND DEVELOPMENT WITH REACT.JS**

**Team members**

Santhosh K (**Team Leader**) - santhoshk-cswithai@srmasc.ac.in

Harish Karthikeyan T - t.harishkarthikeyancswithai@srmasc.ac.in

Vetrivel M- vetrivelm-cswithai@srmasc.ac.in

Premkumar A - premkumara-cswithai@srmasc.ac.in

Prabhakaran B  - [prabhakaranb-cswithai@srmasc.ac.in](mailto:prabhakaranb-cswithai@srmasc.ac.in)

**PHASE 1: REQUIREMENT GATHERING**

**Objective:** Define the purpose, target audience, and key features of the app.

**Tasks:**

* Extract the code from the provided Google Drive link by accessing the shared folder, downloading the project files, and ensuring all necessary assets are included.
* Set up the development environment by opening VS Code and creating a new project folder.
* Download and install the following essential packages using the terminal in VS Code:
  + **React.js:** For building the user interface components.
  + **React Router DOM:** To enable navigation between pages without refreshing the browser.
  + **Axios:** For making HTTP requests to fetch exercise data from the Express DB API.
  + **Tailwind CSS:** To style the app with a responsive and clean design.
* Identify and list the core functionalities required for the app, such as:
  + Exercise search by body parts and equipment.
  + Integration of workout tutorial videos via YouTube.
  + Display of categorized exercises.
* Gather data requirements by understanding the structure of the Fitness API and planning how to fetch and display work out details dynamically.
* Finalize the tech stack, confirming the use of **React.js** for the frontend and **Express DB** for handling workout data. **Deliverables:**
* A well-documented requirement specification.
* Project setup in VS Code with the basic structure and installed packages.

**PHASE 2: DESIGN**

**Objective:** Create wireframes and design user interfaces. **Tasks:**

* Use design tools (like Figma or Adobe XD) to sketch wireframes for key pages, including:
  + **Home Page:** Showcasing workout categories and a search bar.
  + **Exercise Page:** Displaying detailed workout descriptions and embedded videos.
  + **Categories Page:** Organizing exercises by body parts and equipment.
* Define a component structure, breaking the UI into reusable components such as:
  + **Navbar:** For navigation across pages.
  + **Footer:** To display app information and social media links.
  + **Hero Section:** Highlighting featured exercises and motivational quotes.
* Plan responsive design strategies to ensure the app works seamlessly on desktops, tablets, and mobile devices. **Deliverables:**
* Completed wireframes.
* A finalized design layout for all pages.

**PHASE 3: DEVELOPMENT**

**Objective:** Implement the core functionalities of the app. **Tasks:**

* Set up a new React.js project using the terminal command:

npx create-react-app fitflex-fitness-app

* Organize the folder structure as follows:
* /src
* ├── assets/ # Images, fonts, and static files
* ├── components/ # Reusable UI components (Navbar, Footer, etc.)
* ├── pages/ # Page components (Home, Exercise, Categories)
* ├── styles/ # Component-specific CSS
* ├── App.js # Root component containing routes

├── index.js # Renders the app into the DOM

* Build reusable components like:
  + **Navbar.jsx** for navigation.
  + **Footer.jsx** for app details.
  + **Button.jsx** for interactive buttons.
  + **Card.jsx** for displaying workout details.
* Integrate the Express DB API to fetch and display real-time workout data.
* Implement search and filter features using components like **HomeSearch.jsx**, **BodyPartsCategory.jsx**, and **EquipmentCategory.jsx**.
* Embed YouTube videos in the **Exercise.jsx** component to guide users through workouts. **Deliverables:**
* A working app with dynamic data fetching and a responsive interface.

**PHASE 4: STATE MANAGEMENT INTEGRATION**

**Objective:** Implement local and global state management. **Tasks:**

* Use React's **useState** hook for local state, handling:
  + Search inputs and live updates.
  + Modal visibility for pop-ups.
  + Exercise filters.
* Plan global state management by:
  + Setting up a **Context API** structure.
  + Defining shared data like user authentication status, selected filters, and theme preferences.
* Implement context providers and consumers for data consistency across components. **Deliverables:**
* A detailed plan for managing state.
* Integrated Context API (or placeholders for future use).

**PHASE 5: TESTING**

**Objective:** Ensure the app functions as expected. **Tasks:**

* Write unit tests using **Jest** to check individual functions and components.
* Conduct component tests with **React Testing Library** to ensure:
  + Correct rendering of UI elements.
  + Proper API calls and data fetching.
  + Navigation using React Router DOM.
* Identify and document bugs and inconsistencies. **Deliverables:**
* A comprehensive test report.
* A bug tracking sheet.

**PHASE 6: DEPLOYMENT**

**Objective:** Launch the app for public or test use. **Tasks:**

* Choose a deployment platform such as **Netlify** or **Vercel**.
* Build the production version of the app:

npm run build

* Deploy the app and obtain a live URL.
* Test the deployment link for broken routes, missing data, or design errors. **Deliverables:**
* A functional live app link.
* Deployment documentation.

**PHASE 7: MAINTENANCE AND FUTURE ENHANCEMENTS**

**Objective:** Monitor app performance and plan new features. **Tasks:**

* Track known issues such as:
  + Filter functionalities needing improvement.
  + Missing global state management.
  + Lack of user authentication.
* Plan future enhancements:
  + **Workout List Display:** Adding advanced filters for muscle groups and equipment.
  + **YouTube Integration:** Embedding customizable video players for a seamless experience.
  + **Dark Mode:** Implementing a toggle for light and dark themes.
* Set up a maintenance log to record bugs and fixes. **Deliverables:**
* A detailed roadmap for upcoming features.
* A maintenance log for ongoing improvements.