**Project Documentation**

FitFlex : Your Personal Fitness Companion

**1. Introduction**

Team ID : NM2025TMID33664

Team Size : 5

Team Leader : Mahasri.R (mahasri2642007@gmail.com)

Team member : ASHIHA.A (aashiha500@gmail.com)

Team member : GOWTHAMI D (gowthamanvpr@gmail.com)

Team member : KAVYASRI. M (manikandanradhakrishnaan@gmail.com)

Team member : MAHASRI.S (mahasrimahasrisekar@gmail.com)

**2. Project Overview**

Technology Stack: React.js, React Router, Context API / Redux, Axios, HTML5, CSS3 / Tailwind / Styled Components

Goal: Provide users with tools to log workouts, track progress, and get fitness insights.

Core Features:

User authentication (login/register)

Workout logging and tracking

Progress visualization (charts/graphs)

Responsive and mobile-first UI

Customizable fitness plans

---

**3. Architecture**

Frontend: React.js SPA

Routing: React Router DOM

State Management: Context API / Redux (choose one based on implementation)

API Communication: Axios for RESTful API calls

Component Structure: Functional components with Hooks

Client (React.js)

│

├── Components (UI + Logic)

│

├── Pages (Routed Views)

│

├── Context / Redux (Global State)

│

└── Services (API)

---

**4. Setup Instructions**

Prerequisites:

Node.js (>= 14)

npm or yarn

Steps:

# Clone the repository

git clone https://github.com/your-username/fitflex.git

# Navigate into the project folder

cd fitflex

# Install dependencies

npm install

# Start the development server

npm start

---

**5. Folder Structure**

fitflex/

├── public/

│ └── index.html

├── src/

│ ├── assets/ # Images, icons, etc.

│ ├── components/ # Reusable UI components

│ ├── pages/ # Main pages/views

│ ├── context/ # Context API or Redux setup

│ ├── services/ # API logic

│ ├── styles/ # CSS / SCSS / Tailwind

│ ├── App.js

│ └── index.js

├── .env # Environment variables

├── package.json

└── README.md

---

**6. Running the Application**

# Start the development server

npm start

# Build for production

npm run build

# Run tests

npm test

The app will be available at http://localhost:3000 by default.

---

**7. Component Documentation**

Component Name Description Props

WorkoutCard Displays a single workout workoutData, onEdit

ProgressChart Visualizes progress using charts data

Navbar Navigation bar user, onLogout

LoginForm Handles user login onLogin

PlanCustomizer Customize fitness plans userPlan, onSave

> Detailed documentation is available in each component file as JSDoc comments.

---

**8. State Management**

Tool Used: Context API / Redux

State Examples:

authState: Stores login token and user info

workouts: Stores logged workouts

plans: Stores fitness plans

Sample Context:

const AppContext = createContext();

export const AppProvider = ({ children }) => {

const [user, setUser] = useState(null);

const [workouts, setWorkouts] = useState([]);

return (

<AppContext.Provider value={{ user, workouts }}>

{children}

</AppContext.Provider>

);

};

---

**9. User Interface**

Design Principles:

Mobile-first responsive design

Minimalistic and intuitive layout

Consistent color theme and typography

Tools: Figma (for mockups), React Icons, Tailwind CSS / Styled Components

---

**10.Styling**

Approach: Utility-first (Tailwind) / CSS Modules / Styled Components

Theme: Light/Dark mode (optional)

Responsive Breakpoints: Based on mobile, tablet, and desktop views

---

**11. Testing**

Tools: Jest, React Testing Library

Test Coverage: Unit tests for components and utility functions

Sample Test:

test('renders workout card', () => {

render(<WorkoutCard workoutData={mockWorkout} />);

expect(screen.getByText(/Pushups/i)).toBeInTheDocument();

});

---

**12. Screenshots or Demo**

https://drive.google.com/file/d/1KgPGNkLIPoFcVpQWySEu46wWAFc-REQP/view?usp=drivesdk

---

**13. Known Issues**

Chart sometimes renders incorrectly on window resize

No offline support yet

Limited accessibility (WAI-ARIA incomplete)

---

**14. Future Enhancements**

AI-powered fitness plan generator

PWA support for offline use

Integration with wearable devices (e.g., Fitbit, Apple Watch)

Social features (e.g., friend challenges, community sharing)

Accessibility improvements (keyboard navigation, screen reader support)

---