NAAN MUDHALVAN PROJECT

Project Title:Fitflex

Your personal fitness companion

---

1. Introduction

Project Title : *[Fitflex]*

Team ID : *[*NM2025TMID47132*]*

Team Leader Name : *[A.Sowndharya]*

Mail ID : *[annamalaikrishnan87@gmail.com]*

Team Members:

S.Kaviya- document - kaviyasara07@gmail.com

P.Karthika- document - karthika14595@gmail.com

G.Soni priya- demo video making - sonipriya50424@gmail.com

2. Project Overview

Purpose:
The purpose of *Your Personal Fitness Companion* is to help users maintain a healthy lifestyle by providing personalized workout plans, nutrition tracking, and progress monitoring. The application is designed to be an all-in-one fitness solution that motivates users to achieve their health and wellness goals.

Features:

- Personalized workout routines based on user preferences and fitness level
- Nutrition tracking with daily calorie and nutrient monitoring
- Progress dashboard with charts and statistics
- Goal-setting and achievement reminders
- Integration with wearable fitness devices (future enhancement)
- User profile management and secure login

3. Architecture

Component Structure:

- **Header** – Navigation bar for quick access to different sections
- **Dashboard** – Displays user workouts, nutrition stats, and goals
- **WorkoutPlanner** – Allows users to create and edit workout plans
- **NutritionTracker** – Logs meals and calculates daily calorie intake
- **ProgressCharts** – Visualizes fitness progress with charts and graphs
- **Profile** – Stores user information and preferences
- **Footer** – Static links, contact info, and app details

State Management:

- **Global State:** Managed using React Context API to handle user data, workout logs, and nutrition data
- **Local State:** Used for form handling (adding workouts, logging meals) and UI toggles (dark mode, modals, etc.)

Routing:
Implemented with React Router v6

- `/dashboard` – Overview of fitness stats
- `/workouts` – Workout planner and history
- `/nutrition` – Meal and calorie tracker
- `/progress` – Charts and graphs of progress
- `/profile` – User settings and details

4. Setup Instructions

Prerequisites:

- Node.js v18+
- npm or yarn
- Git

Installation:

```
# Clone the repository
git clone https://github.com/yourrepo/personal-fitness-companion.git

# Navigate to client folder
cd personal-fitness-companion/client
```

```
# Install dependencies
npm install
```

Configure Environment Variables:
Create a `.env` file in the root directory and add the following:

```
REACT_APP_API_URL=https://api.personalfitnesscompanion.com
```

5. Folder Structure

```
personal-fitness-companion/
│
├── client/                 # Frontend application
│   ├── public/             # Static files
│   ├── src/                # Source code
│   │   ├── components/     # Reusable UI components (Button, Card, Chart,
etc.)
│   │   ├── pages/          # Page components (Dashboard, Workouts, Nutrition,
Progress, Profile)
│   │   ├── contexts/       # Context API providers (UserContext,
WorkoutContext, NutritionContext)
│   │   ├── hooks/          # Custom React hooks
│   │   ├── utils/          # Utility functions (auth.js, api.js,
formatters.js)
│   │   ├── App.js          # Main app component
│   │   └── index.js        # Entry point
│   │
│   └── assets/             # Images, icons, and screenshots
│
└── README.md               # Project documentation
```

Utilities:

- `auth.js` – Helper for authentication
- `api.js` – Wrapper for API calls
- `formatters.js` – Utility for formatting data

6. Running the Application

To start the frontend server locally:

```
npm start
```

This will run the React development server at:

👉 **http://localhost:3000**

The application will automatically reload if you make changes to the code.

7. Component Documentation

Key Components:

- **Dashboard.js** – Displays an overview of workouts, nutrition stats, and goals
- **WorkoutForm.js** – Create or edit workout routines *(props: onSave, initialData)*
- **NutritionTracker.js** – Log meals and calories *(props: userId)*
- **ProgressCharts.js** – Visualize progress with charts and graphs
- **Profile.js** – Manage user details and settings

Reusable Components:

- **Button.js** – Styled button component for consistent UI
- **Card.js** – Reusable container for displaying grouped content
- **Chart.js** – Wrapper for displaying different types of charts (bar, line, pie)
- **Modal.js** – Reusable popup modal for forms and alerts

8. State Management

Global State (Context API):

- **UserContext** – Stores user profile details (name, age, fitness goals, preferences)
- **WorkoutContext** – Manages workout lists, routines, and progress
- **NutritionContext** – Stores meal logs, calorie intake, and nutrition data

Local State (useState hooks):

- Form input handling for workouts and nutrition logs
- UI state toggles (dark mode, modal open/close, sidebar toggle)
- Temporary data before saving into global state

Why Context API?

- Centralized management of user data
- Easy access to fitness stats across multiple components
- Reduces prop drilling in nested components

9. User Interface

The application provides a clean and user-friendly interface to help users easily track and manage their fitness journey.

Screens / Views:

- **Dashboard** – Overview of daily stats, calories burned, and workout summary
- **Workout Planner** – Create, edit, and view workout routines
- **Nutrition Tracker** – Log meals and monitor calorie/nutrient intake
- **Progress Charts** – Graphical representation of progress (weight, workouts, calories)
- **Profile Page** – User settings, goals, and personal information

Visual Elements:

- Simple navigation bar for easy access to all sections
- Light and dark theme options
- Responsive design for desktop and mobile screens
- Interactive charts and cards for better visualization

Screenshots/GIFs (to be added):

- Dashboard with summary stats
- Workout planner form
- Nutrition logging screen
- Progress chart view

10. Styling

CSS Frameworks/Libraries:

- **Tailwind CSS** – Utility-first styling for fast and responsive design
- **Styled Components** – Component-level styling for reusability and modularity

Theming:

- **Light Theme:** Clean and bright colors for readability
- **Dark Theme:** Dark background with contrasting text for reduced eye strain
- Theme switching implemented using **CSS variables** and **React Context**

Design Principles:

- Minimalist UI with focus on content
- Consistent color scheme across all pages

- Reusable design patterns for buttons, cards, and forms
- Fully responsive layouts for desktop, tablet, and mobile devices

11. Testing

Testing Strategy:

- **Unit Testing:** Conducted using **Jest** and **React Testing Library** to ensure individual components (e.g., forms, buttons) work correctly.
- **Integration Testing:** Validates interactions between components (e.g., Workout Planner with Progress Charts).
- **End-to-End Testing (Planned):** Using **Cypress** to simulate real user interactions across the application.

Code Coverage:

- Jest coverage reports are generated after each test run.
- Goal: Maintain **>80% coverage** for all critical components (Dashboard, Workout Planner, Nutrition Tracker).

Error Handling Tests:

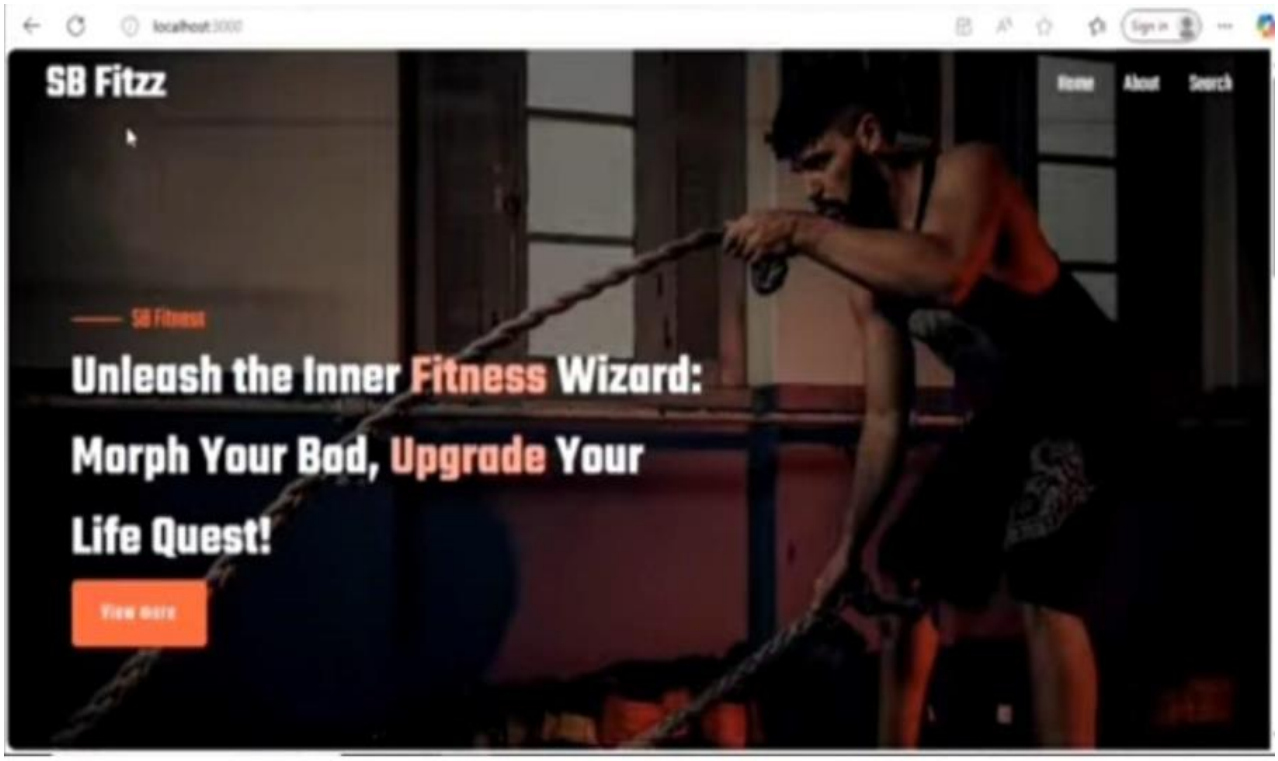
- Form validation (empty fields, invalid input).
- API error handling (server downtime, rate limits).

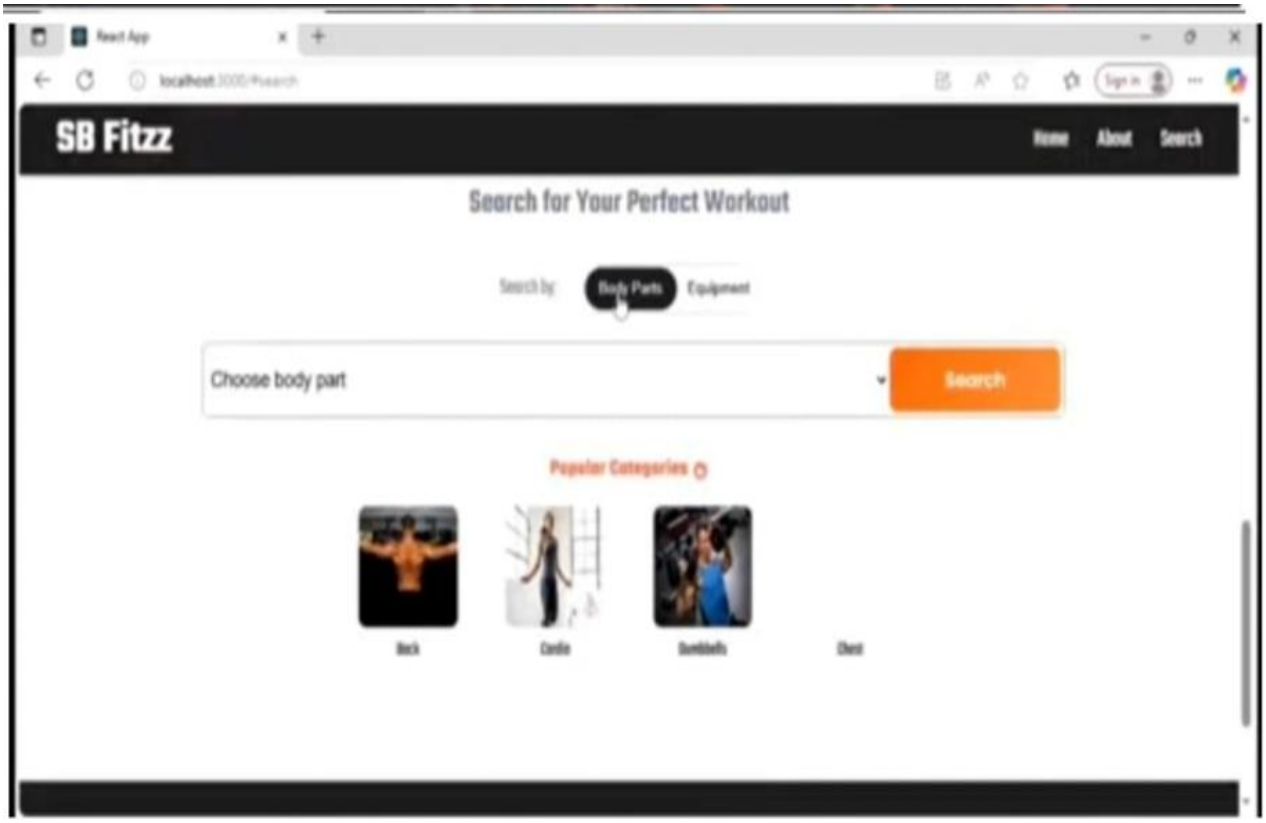
12. Screenshots or Demo

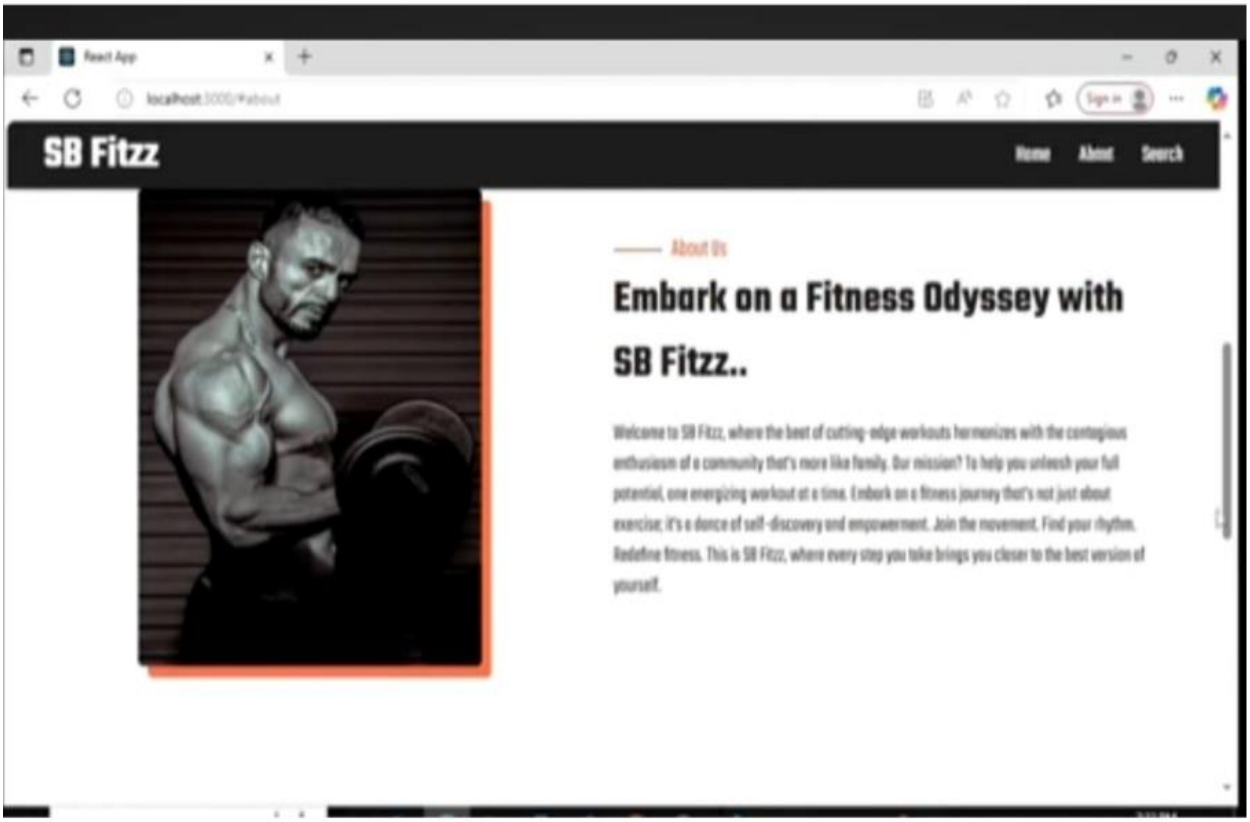
Demo Link:

👉https://drive.google.com/file/d/1fCi6QOvXKtOxMC2tCotY2Z7gfpUxiNMt/view?usp=sharing

Screenshots Folder:







Screenshots to Include:

- **Dashboard View** – Summary of workouts, calories, and goals
- **Workout Planner** – Interface for creating and editing routines
- **Nutrition Tracker** – Meal logging and calorie tracking form
- **Progress Charts** – Graphs displaying progress over time
- **Profile Page** – User details and settings

13. Known Issues

- API rate limits may occasionally cause delays in fetching user data
- Wearable device integration (e.g., Fitbit, Apple Watch) is not yet implemented
- Some charts may not display correctly on very small mobile screens
- Dark mode theme might have minor inconsistencies in older browsers
- Offline mode is not supported yet

14. Future Enhancements

- **AI-based Workout Recommendations:** Suggest personalized workouts based on user performance and goals
- **Integration with Wearable Devices:** Sync with Apple Health, Google Fit, or Fitbit for real-time tracking
- **Gamification Features:** Add badges, achievements, and challenges to motivate users
- **Mobile App Version:** Develop native iOS and Android applications
- **Social Features:** Allow users to share progress and connect with friends for group challenges
- **Offline Mode:** Enable logging workouts and meals without internet connectivity

---