1 . Introduction

- **Title:** Your Personal Fitness Companion
- **Team ID :** NM2025TMID29923
- **Team Leader :** SAHANA M & 202400503@sigc.edu
- **Team Members :**

YOGALAKSHMI R & 202400909@sigc.edu

DEVI SRI K & 202400723@sigc.edu

DHANU SRI S & 202400101@sigc.edu

2. Project Overview

- **Purpose:**

The Personal Fitness Companion is designed to help users track, monitor, and improve their fitness journey. It provides personalized workout plans, nutritional guidance, and progress tracking in one place.

- **Features:**

Personalized workout routines

Diet and nutrition tracking

Real-time progress dashboard

3. Architecture

Frontend:

- **React.js**: A powerful JavaScript library for building dynamic and responsive user interfaces.

Utilizes **React hooks** for managing state and lifecycle events.

Efficient **component-based architecture** for reusability and modular development.

- **Bootstrap**: Provides pre-built, responsive design components for quickly building user interfaces.

    # Helps create consistent layouts and UI elements without custom CSS.

- **Material UI**: A popular React UI framework that implements Google's Material Design guidelines.

    # Features include pre-designed components like buttons, forms, navigation, and typography.

Backend:

- **Node.js**: A runtime environment for executing JavaScript code server-side, using an event-driven, non-blocking I/O model for scalable applications.

    # Enables high-performance, real-time applications (like chat systems).

    # Leverages JavaScript across both the frontend and backend for consistency in development.

- **Express.js**: A lightweight and flexible Node.js web application framework.

    # Simplifies routing and middleware integration for handling HTTP requests.

    # Provides easy integration with RESTful APIs and third-party services.

    # Helps in setting up **API endpoints** for data exchange between client and server.

Database:

- **MongoDB**: A NoSQL document-oriented database designed for high performance and scalability.

    # Stores data in flexible, JSON-like **BSON** documents, ideal for structured or unstructured data.

    # Suitable for applications with evolving data models (e.g., user profiles, projects, and messages).

    # Offers **real-time data sync**, enabling instant updates between users and the database.

Mongoose ORM is used for data modeling and validation, providing a schema-based solution for MongoDB.

- **Bootstrap**: Provides pre-built, responsive design components for quickly building user interfaces.

  # Helps create consistent layouts and UI elements without custom CSS.

- **Material UI**: A popular React UI framework that implements Google's Material Design guidelines.

  # Features include pre-designed components like buttons, forms, navigation, and typography.

Backend:

- **Node.js**: A runtime environment for executing JavaScript code server-side, using an event-driven, non-blocking I/O model for scalable applications.

  # Enables high-performance, real-time applications (like chat systems).

  # Leverages JavaScript across both the frontend and backend for consistency in development.

- **Express.js**: A lightweight and flexible Node.js web application framework.

  # Simplifies routing and middleware integration for handling HTTP requests.

  # Provides easy integration with RESTful APIs and third-party services.

  # Helps in setting up **API endpoints** for data exchange between client and server.

Database:

- **MongoDB**: A NoSQL document-oriented database designed for high performance and scalability.

  # Stores data in flexible, JSON-like **BSON** documents, ideal for structured or unstructured data.

  # Suitable for applications with evolving data models (e.g., user profiles, projects, and messages).

Offers **real-time data sync**, enabling instant updates between users and the database.

 # Mongoose ORM is used for data modeling and validation, providing a schema-based solution for MongoDB.

4. Setup Instructions

- **Prerequisites:**

Node.js

MongoDB

Git

React.js

Express.js – Mongoose – Visual Studio Code

- **Installation Steps:**

Clone the repository git clone

Install client dependencies cd client npm install

Install server dependencies cd ../server npm install

5. Folder Structure

```
SB-Works/
    |-- client/                          # React frontend
        |__components/
        L__ pages/
    |__ server/                          # Node.js backend
        |__routes/
            |__ models/
                |__ controllers/
```

6. Running the Application

- **Frontend:**

```
cd client
npm start
```

- **Backend:**

```
cd server
npm start
```

- **Access:** Visit http://localhost:3000

7. API Documentation

- **User:**

– /api/user/register

– /api/user/login

- **Projects:**

– /api/projects/create

– /api/projects/:id • Applications: /api/apply

- **Chats:**

– /api/chat/send

– /api/chat/:userId

8. Authentication

- JWT-based authentication for secure login
- Middleware protects private routes

9. User Interface

- Landing Page
- Freelancer Dashboard
- Admin Panel

- Project Details Page

10 . Testing

- **Testing Strategy**:

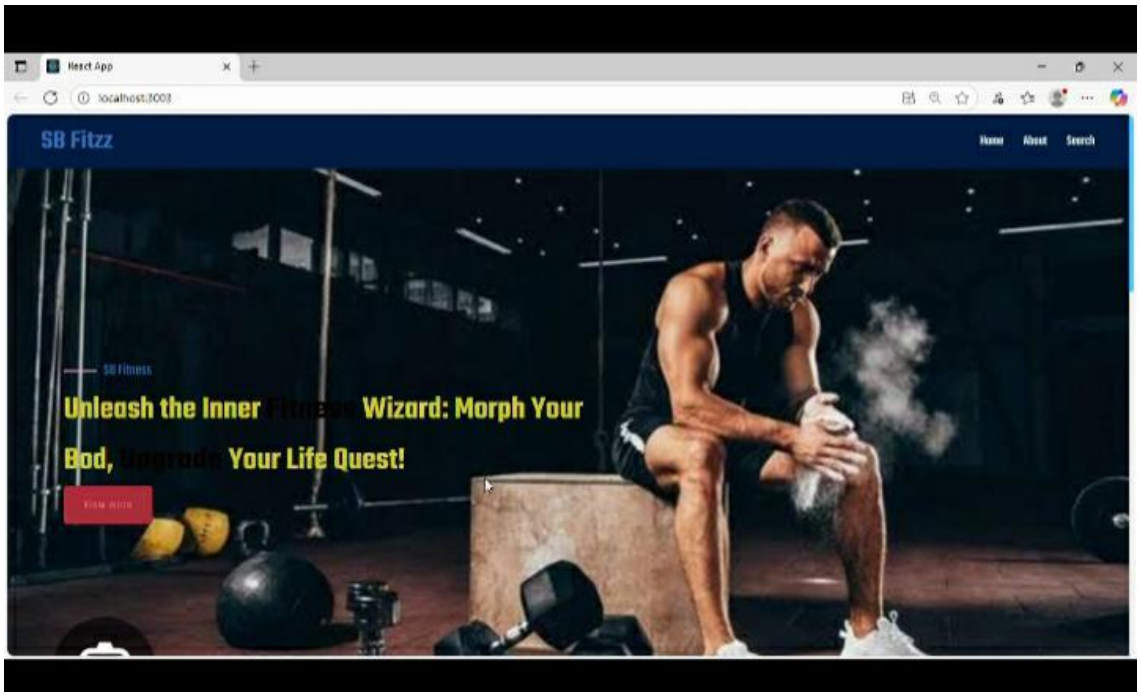
  Jest and React Testing Library are used for unit and integration tests to ensure components work as expected.

- **Code Coverage**:

  Coverage reports generated using Jest to ensure quality and maintainability.

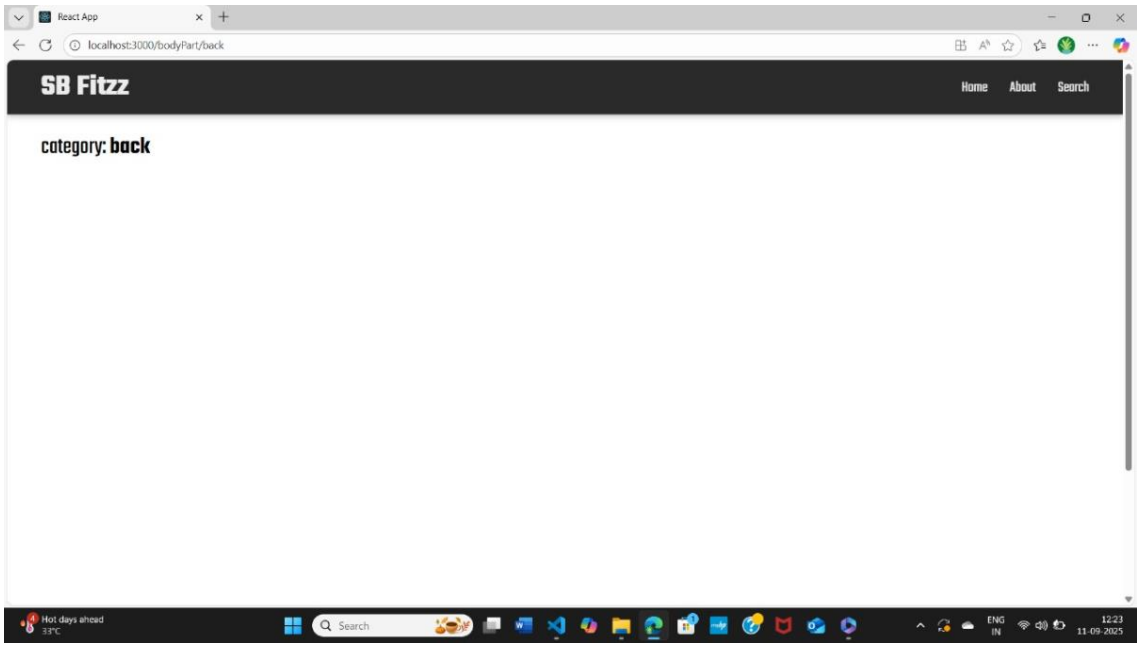
11. Screenshots or Demo

Demo video :
https://drive.google.com/file/d/1xUwHTlVgyODGvuN1wHFAzHFVVw01ku1c/view?usp=drivesdk



12. Known Issues

Currently, integration with third-party wearable APIs is in beta and may experience intermittent issues.



13. Future Enhancement

Planned features include:

- AI-based workout suggestions
- Social features to connect with friends
- Enhanced analytics with deeper insights