## Front End Engineering-II /Artificial

## Intelligence and Machine Learning

Project Report

Semester-VI (Batch-2022)

The Fitness Club

A red and white sign

Description automatically generated with low confidence

**Supervised By: Submitted By:**

Mr. Rahul Singh Rajput Naina, 2210991949 (G-22)

Arsh Jain, 2210991949 (G-22)

**Department of Computer Science and Engineering**

## Chitkara University Institute of Engineering & Technology,

## Chitkara University, Punjab

# Abstract

This project report details the development of a web-based The Fitness Club application built with React.js. This platform empowers both renters and item owners by facilitating a smooth Fitness experience through a user-centric interface for listing, and managing your workouts.

The report provides a deep dive into the technical aspects, including:

#### Functionality:

* + Workout creation: Users can easily add detailed descriptions of the workouts they are doing on each day, and get Smart suggestions for their respective goal.
  + Managing tools: The platform facilitates in managing the workouts that are being done on each day and also track calories for the same .
  + Reward Badges: Reward system has been implemented after a specific number of calories burnt.

#### React.js Implementation:

* + The report explains how React components are utilized to structure the user interface, ensuring a modular and reusable approach.
  + It delves into the usage of state management techniques (e.g., Redux, Context API, usestates) to maintain dynamic data and provide a responsive user experience.

The report further explores:

#### Challenges Encountered:

* + The report discusses specific technical hurdles faced during development, such as handling complex data structures, integrating with third-party APIs, or optimizing performance for a large number of listings.
  + It elaborates on the solutions implemented to overcome these challenges, showcasing the problem-solving approach taken during development.
  + Encountered some axios errors while data passing for smart suggestion feature.

#### Future Enhancements:

* + The report proposes potential improvements and additional features that could be integrated into the fitness hub in the future. This might include features like user reviews and ratings, payment gateways for membership for exclusive access to services.

Overall, this project report demonstrates the creation of a robust and user-friendly The Fitness Club application utilizing React.js technology. It highlights the technical choices made, the challenges overcome, and the potential for future improvements, showcasing a comprehensive development process.

### Introduction

This web-based fitness application, developed using React.js, allows users to efficiently track their workouts and calorie intake. To enhance the user experience, it also includes a smart suggestion feature that provides personalized workout recommendations based on each user's fitness goals, experience level, and available time. This helps users stay consistent with their routines and achieve their health and fitness objectives more effectively.

**Background**

The importance of physical fitness and mindful calorie management has grown with rising awareness of health-related issues. While numerous fitness applications exist, many fail to provide a user-friendly and customizable solution for tracking workouts and calories. The absence of a streamlined interface and personalized features often deters users from maintaining consistent fitness records. This project seeks to bridge these gaps by offering a simple yet robust platform for users to log their workout details and calorie data in an organized manner. Built with React.js, the application ensures an intuitive user experience and facilitates effective fitness management.

#### Problem Statement

Existing fitness tracking solutions often lack simplicity and flexibility, resulting in an unsatisfactory user experience. Users frequently face challenges such as cumbersome interfaces, limited customization options, and inadequate support for organizing their fitness data. Moreover, many applications fail to focus on providing users with an easy way to track progress over time. These limitations can hinder users from staying committed to their fitness routines.

This project aims to develop a React.js-based workout and calorie tracking application that addresses these shortcomings. By providing an intuitive interface, efficient data management features, and seamless access to workout history, this application ensures that users can focus on achieving their fitness goals without unnecessary obstacles.

# Software and Hardware Requirement Specification

#### Methods

* + 1. **Code Editor/IDE**:
       1. **Tools**: Visual Studio Code, Sublime Text, IntelliJ IDEA.
       2. **Availability**: Free download from their official websites (e.g., Visual Studio Code).
  1. **Programming/Working Environment**
     1. **Programming Languages and Frameworks**:
        1. **JavaScript (for Frontend and Backend)**:

a**.Tools**: Node.js runtime, npm (Node Package Manager).

b**.Availability**: Free download from the Node.js official website.

#### Frontend Libraries and Frameworks:

* + - * 1. **React.js**: Available via npm (install using npm install react).
        2. **Tailwind CSS**: Available via npm (install using npm install tailwindcss).
        3. **Availability**: Documentation and installation guides can be found on the official React and Tailwind CSS websites.

#### Backend Frameworks:

* + **Express.js**: Available via npm (install using npm install express).
  + **Availability**: Documentation is accessible on the Express.js official website.
  1. **Requirements to Run the Application**
     1. **Database Management System (DBMS)**:
        1. **MongoDB (NoSQL Database)**:
           1. **Tools**: MongoDB, MongoDB Atlas (for cloud database), Mongoose (ODM).
           2. **Availability**: MongoDB can be downloaded for free from the official MongoDB website. MongoDB Atlas offers a free tier for cloud-based databases.

## Database Analyzing, design and implementation

#### Database Analysis

The workout and calorie tracking application requires an efficient, scalable database to store various types of data, such as user profiles, workout logs, and calorie details. Given the nature of the application, MongoDB is selected due to its flexibility in handling dynamic data, scalability, and ease of use for storing documents in a JSON-like format.

The main entities that the database will handle include:

1. Users: The system needs to store user information, including credentials, profile details, and preferences.
2. Workouts: The database will store detailed workout logs, including workout types, durations, and calories burned.
3. Calories: Users can log their daily calorie intake and link it to specific workouts.

By organizing data into these entities, the database will ensure efficient retrieval and management of user information and fitness-related data

**Database Design**

**Database Design**

The database schema for MongoDB will be designed with collections that represent the main entities. Each collection will contain documents with attributes relevant to each entity. Here's an overview of the design:

1. **Users Collection**  
   This collection will store user-related data, including their login credentials and profile information.

**Attributes:**

* **email:** User\u2019s email address (unique).
* **password:** Hashed password.
* **name:** User\u2019s name.
* **createdAt:** Timestamp for when the user profile was created.

1. **Workouts Collection**  
   The workouts collection will store details about each logged workout, including the type of workout and associated calorie data.

**Attributes:**

* **userId:** Reference to the user who logged the workout.
* **workoutType:** Type of workout (e.g., "running," "cycling").
* **duration:** Duration of the workout in minutes.
* **caloriesBurned:** Estimated calories burned during the workout.
* **date:** Date when the workout was performed.
* **notes:** Optional field for any additional notes about the workout.
* **createdAt:** Timestamp for when the workout log was created.

1. **Calories Collection**  
   This collection will store details of the user's daily calorie intake and link it to specific days.

**Attributes:**

* **userId:** Reference to the user who logged the calorie data.
* **date:** Date for which the calorie data is recorded.
* **caloriesConsumed:** Total calories consumed on that day.
* **meals:** Array containing meal details, including meal names and calorie counts (e.g., "breakfast: 300 calories").
* **createdAt:** Timestamp for when the calorie log was created.

This design ensures that user data, workout logs, and calorie details are efficiently stored and accessible, providing a streamlined experience for tracking fitness goals.

#### Database Implementation

The implementation of the database involves setting up the MongoDB connection and defining models (schemas) for each collection using Mongoose.

1. MongoDB Connection: The backend connects to MongoDB using a connection URI, typically stored in environment variables. This allows the backend to interact with the database for operations like adding, updating, or retrieving user and workout data.
2. Mongoose Models:
   * **User Model:** Defines the structure for storing user information, such as email, password, and profile details.
   * **Workout Model:** Stores all details related to workout logs, including workout type, duration, calories burned, and date.
   * **Calorie Model:** Stores details about daily calorie intake, including total calories consumed and meal breakdowns.
3. Data Validation: Mongoose validation ensures that data entered into the database follows the expected formats, such as unique email addresses, valid calorie values, and properly formatted dates.

c. Future Enhancements

1. **Progress Tracking:**
   * Implement features to visualize user progress over time using charts and graphs, based on workout and calorie data.
2. **Search and Filtering:**
   * Introduce advanced search and filtering options to allow users to quickly find specific workout logs or calorie records.
3. **Database Backup and Scalability:**
   * Transition to cloud-based databases (e.g., MongoDB Atlas) and implement regular backups to ensure data reliability.
   * Explore database sharding and replication strategies for better performance and scalability.

## Program’s Structure Analysis and GUI Construction

**a. Program Structure Analysis**

The program's structure revolves around modular components to ensure maintainability, scalability, and flexibility. It follows a typical MVC (Model-View-Controller) architecture that divides the system into three layers:

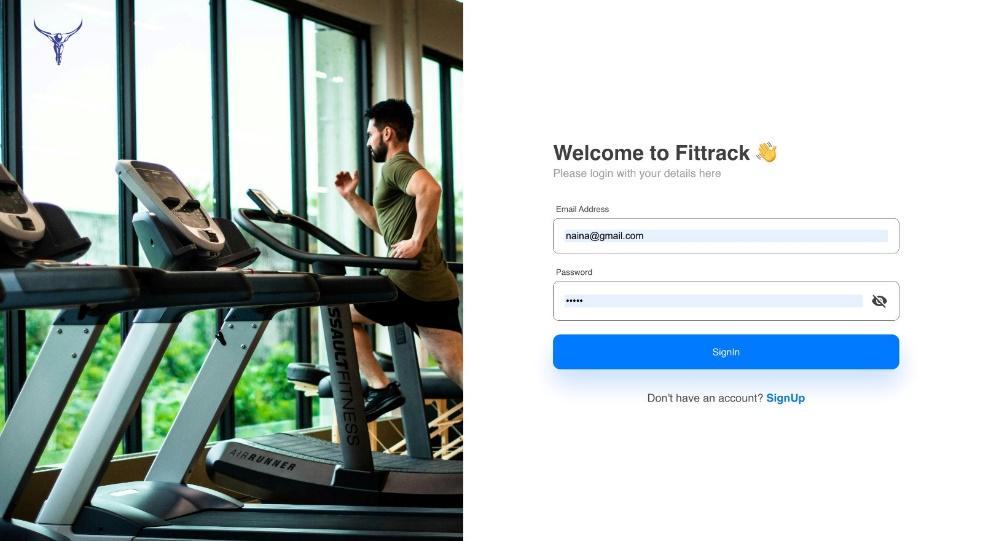
1. **Frontend (View)**
   * React.js Components: The frontend consists of reusable components built with React.js, such as WorkoutLog, CalorieTracker, UserProfile, and Dashboard. Each component focuses on a specific feature, such as workout tracking, calorie input, and user account management.
   * State Management (Redux): Redux is used to manage the application state across various components, handling user authentication status, workout logs, and calorie data.
   * Routing (React Router): React Router is used for navigating between pages (e.g., Home, Workout Logs, Calorie Tracker, User Profile).
   * UI/UX (Graphical Interface): The design emphasizes a clean and user-friendly interface using CSS frameworks (like Bootstrap or custom CSS) for responsive design. Features such as workout forms, calorie logs, and interactive charts are optimized for ease of use.
2. **Backend (Controller)**
   * Node.js and Express.js: The backend handles client requests, such as adding or updating workouts, managing calorie data, and user authentication. It communicates with the database to store and retrieve user and fitness data.
   * APIs:
     + Workout API: For managing workout logs (CRUD operations).
     + Calorie API: For handling calorie intake entries.
     + User API: For managing user account creation, login, and profile settings.
   * Authentication: JWT-based authentication to manage user sessions and secure routes.
   * Middleware: Express middleware is used to handle requests, such as validating user inputs, checking user authentication (JWT), and managing error handling.
3. **Database (Model)**
   * MongoDB: A NoSQL database stores data related to users, workouts, and calories.
     + User Collection: Stores user information, including profile data and login credentials.
     + Workout Collection: Stores details about workouts, such as type, duration, and calories burned.
     + Calorie Collection: Tracks daily calorie intake, linked to users and optional workout entries.
   * Mongoose: A MongoDB ODM (Object Data Modeling) library is used to define schemas for the collections and interact with the database.

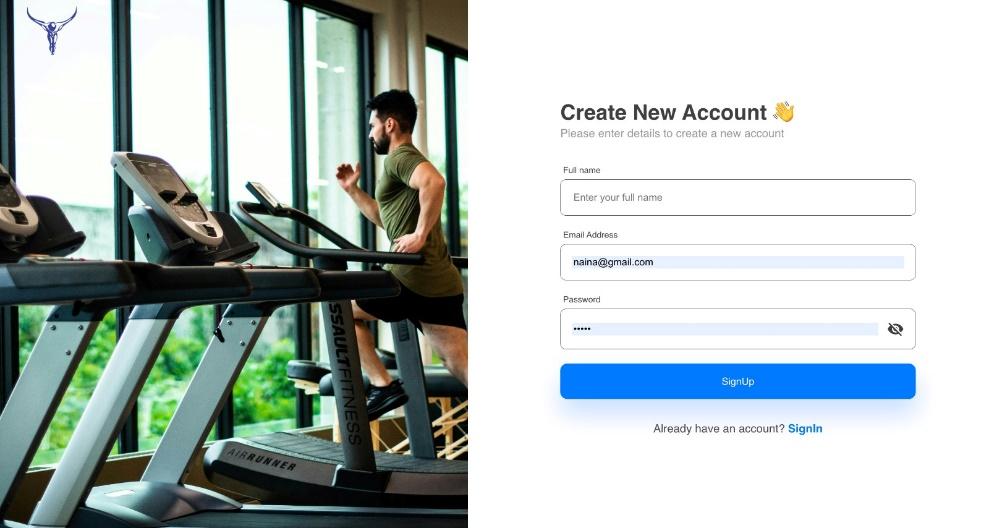
**GUI Construction**

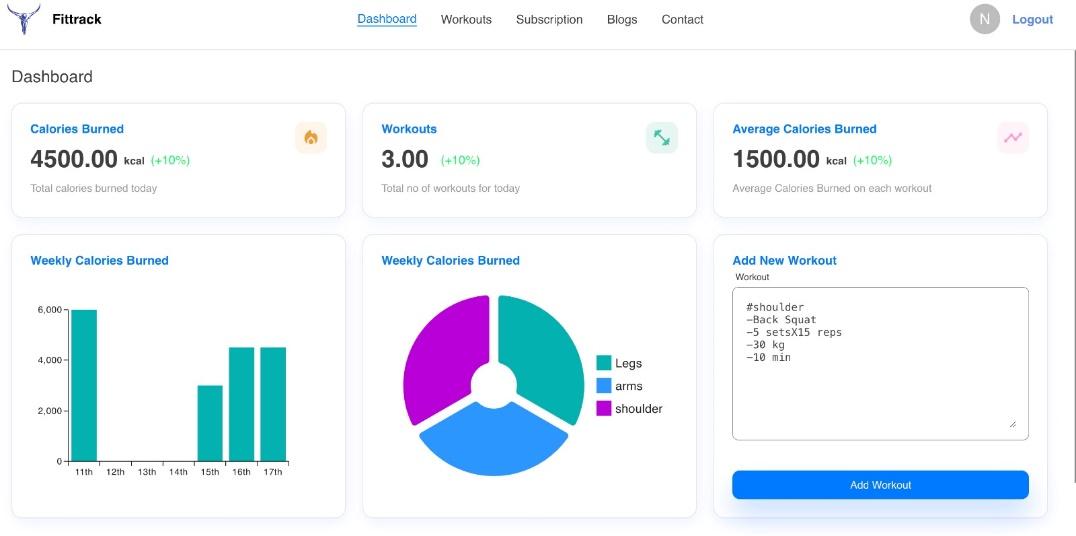
The GUI (Graphical User Interface) is designed to provide users with a seamless and engaging experience, ensuring accessibility and ease of use**.**

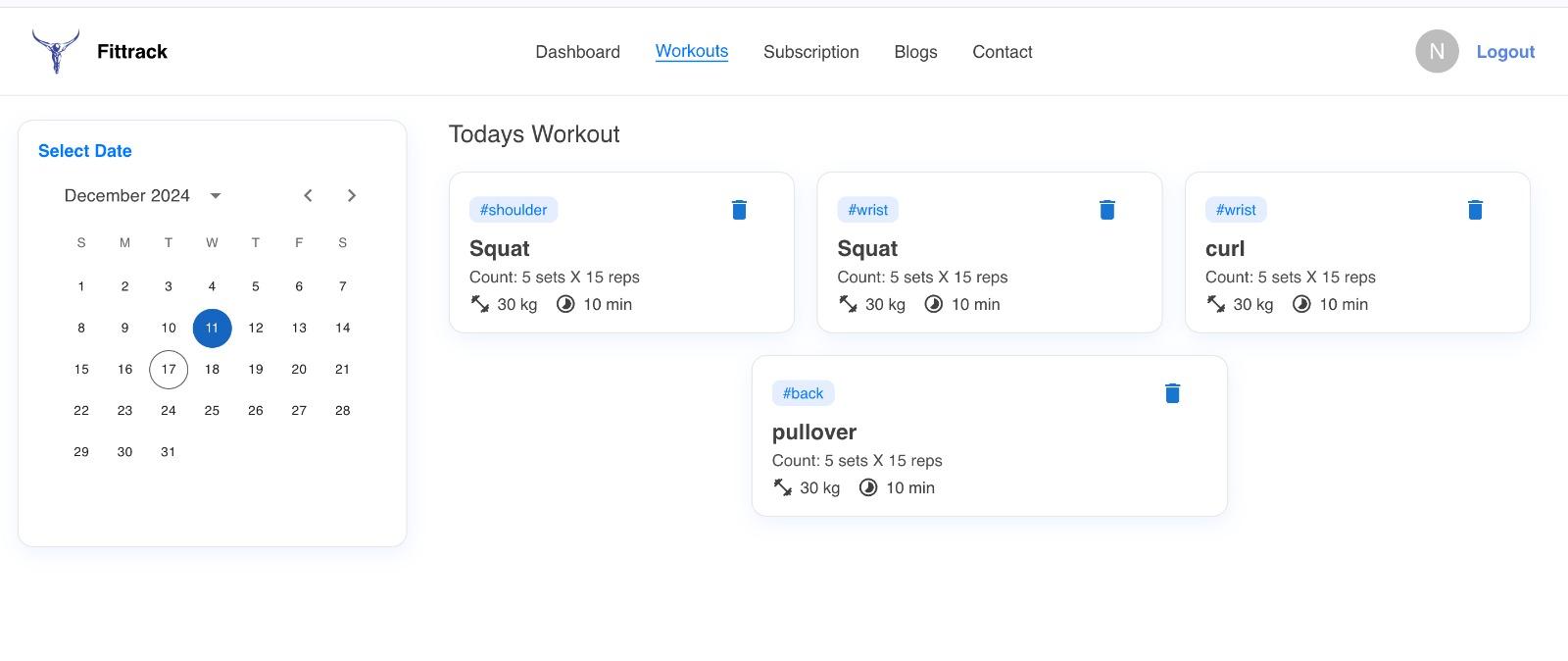
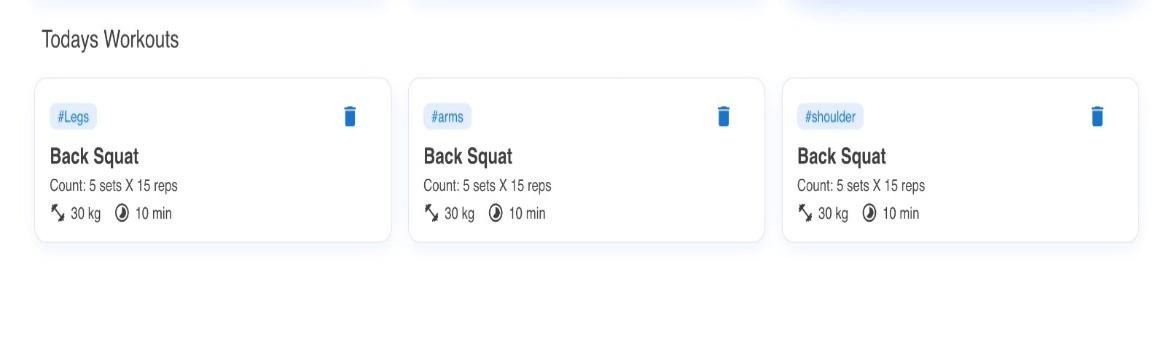
1. **Login & Registration Pages**
   * Forms: Simple and clean forms for user registration and login with input validation.
   * Responsive Design: Ensures that the pages look good on all devices (desktop, tablet, mobile).
   * Authentication: Uses JWT-based authentication to allow users to register or log in via email.
2. **Home Page**
   * Dashboard Overview: Displays a summary of recent workouts, calorie logs, and overall progress in an interactive and visual format (e.g., charts, graphs).
   * Navigation Bar: Includes links to Workout Logs, Calorie Tracker, User Profile, and Logout options.
3. **Workout Logs Page**
   * Workout Cards: Each workout entry is displayed as a card with type, duration, and calories burned. Users can click on a card to edit or delete the entry.
   * Add New Workout: A button or form to add a new workout log with fields for type, duration, and calories burned.
4. **Calorie Tracker column**
   * Daily Log: Users can input their daily calorie intake with fields for meal descriptions and total calorie count.
   * Graphical Overview: A visual chart or graph to display calorie intake trends over time.
5. **Progress Overview** 
   * **Interactive Charts:** Displays progress trends, such as total calories burned over a week or calories consumed vs. burned.
   * **Achievements:** Highlights milestones like completing a set number of workouts or reaching calorie goals**.**

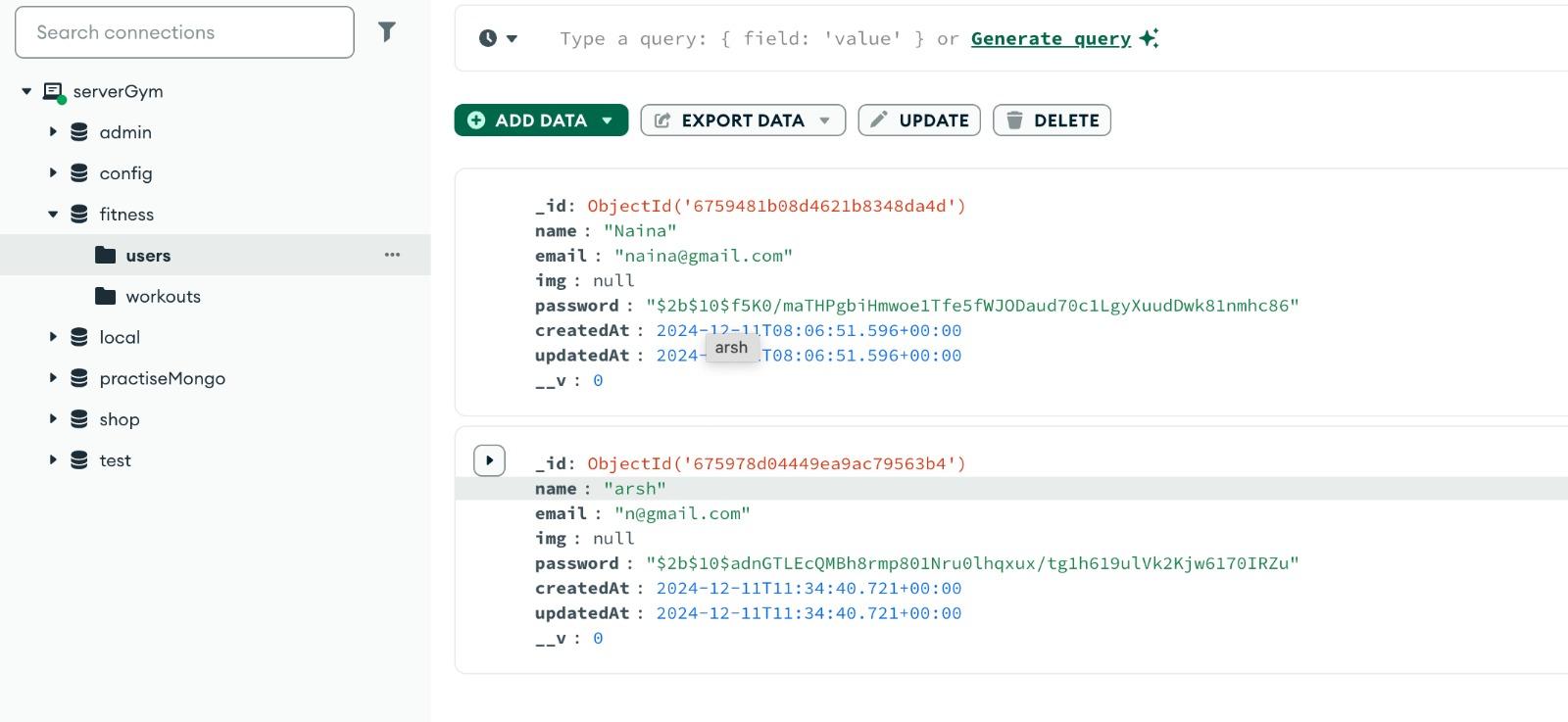
## Project Snapshots:

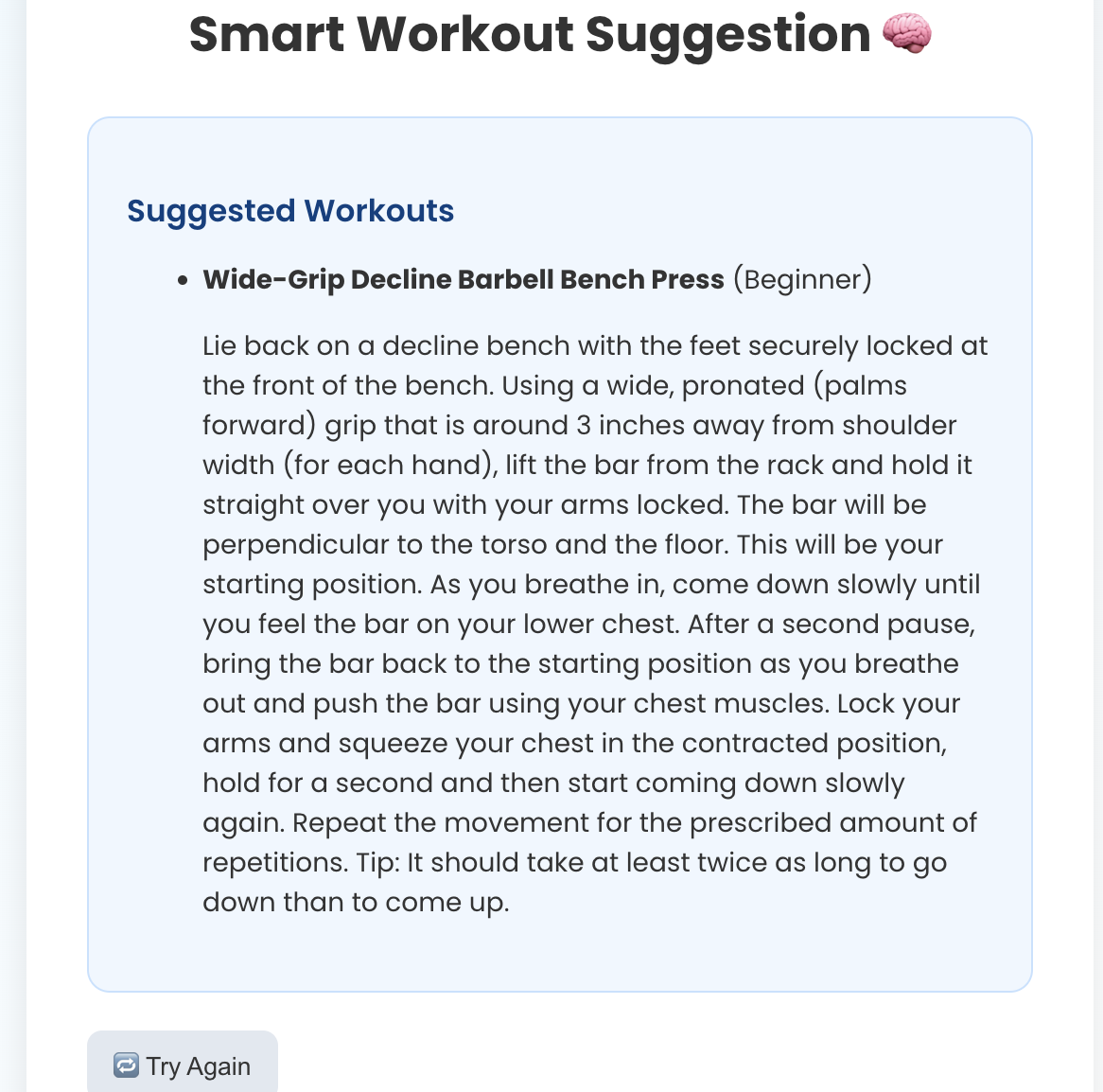
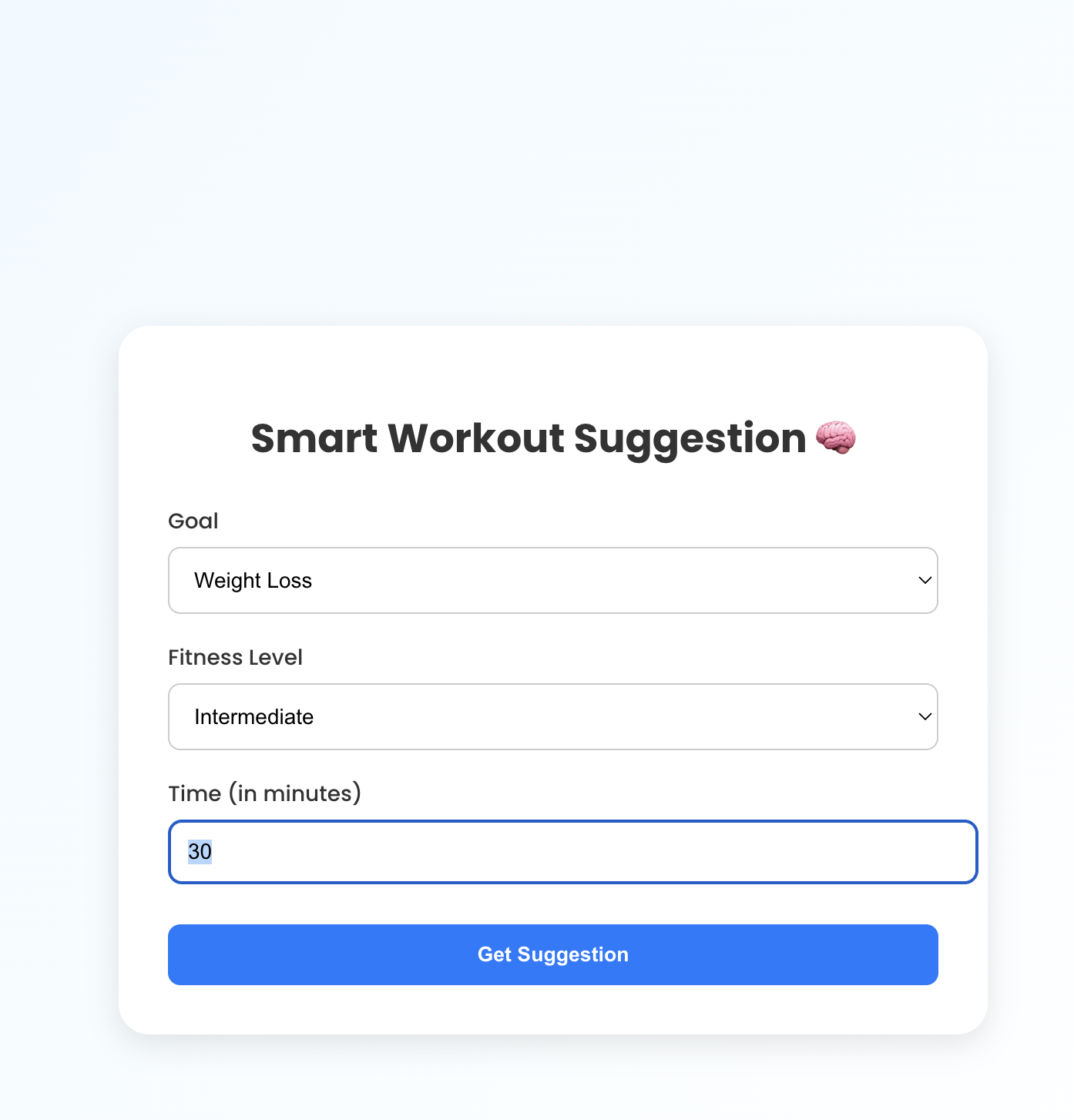










****

### Code-Implementation and Database Connections (If any)

#### Code Implementation Overview

**Frontend (React.js)**

The frontend of **The Fitness Club** application is structured using React.js components to handle various functionalities:

1. **Login Component**: A form that allows users to log in using their credentials. It validates inputs and redirects users to the dashboard upon successful login.
2. **Workout Log Component**: Displays a list of workouts fetched from the backend. It includes features like workout type, duration, calories burned, and options to add or delete workouts.
3. **Calorie Tracker Component**: Allows users to input their daily calorie intake and displays the data in an organized and visual format, such as tables or charts.
4. **State Management with Redux**: Redux is used to manage the global application state. It keeps track of user authentication, workout data, and calorie logs, ensuring smooth synchronization across components.
5. **Routing**: React Router is utilized to manage navigation between pages, such as the dashboard, workout logs, calorie tracker, and user profile.
6. **UI/UX Design**: The interface prioritizes simplicity and usability with a responsive layout optimized for both desktop and mobile devices. It includes features like progress tracking graphs, notifications, and a clean navigation experience.

**Backend (Node.js & Express.js)**

1. **User Authentication**: The backend uses JWT tokens for user authentication. Upon successful login or registration, the system issues a token to manage user sessions and secure API routes.
2. **Workout and Calorie Management**: The backend supports CRUD (Create, Read, Update, Delete) operations for workout logs and calorie entries. This ensures users can easily log, edit, and review their fitness data.
3. **API Endpoints**:
   * **User API**: Manages user registration, login, and profile management.
   * **Workout API**: Handles operations for creating, retrieving, updating, and deleting workout logs.
   * **Calorie API**: Manages daily calorie entries and progress data.
   * **Notification API**: Sends notifications for milestones, reminders, and updates.

**Database (MongoDB)**

The MongoDB database is used to store user and fitness data. The database schema is designed to handle:

1. **User Data**: Includes user information such as email, password (hashed for security), and profile details.
2. **Workout Data**: Stores information about workouts, such as type, duration, calories burned, and timestamps.
3. **Calorie Data**: Logs users' daily calorie intake and tracks it over time for trend analysis.

**State Management (Redux)**

1. **Store**: The global state, managed using Redux, keeps track of user authentication status, workout logs, and calorie data.
2. **Reducers**: Reducers handle actions such as adding workouts, logging calories, and managing user sessions, ensuring consistent updates to the application state.

**6.2 Database Connections**

1. **MongoDB Connection**: The backend connects to MongoDB using Mongoose, which enables seamless interaction with the database for querying, updating, and managing user and fitness-related data.
2. **Mongoose Models**:
   * **User Model**: Defines the schema for storing user details, including email, password, and profile information.
   * **Workout Model**: Defines the schema for logging workout details like type, duration, and calories burned.
   * **Calorie Model**: Defines the schema for tracking daily calorie intake and timestamps

#### Conclusion

The code implementation and database setup for The Fitness Club ensure a secure, efficient, and user-friendly experience. The React.js frontend provides an interactive and visually appealing interface, while the Node.js backend manages user authentication, workout and calorie data, and notifications. MongoDB offers robust and scalable data storage, and Redux enables seamless global state management. These technologies work together to deliver a reliable platform for users to track and achieve their fitness goals.

### System Testing (if any)

Not Applicable

### Limitations (if any)

Not Applicable

### Conclusion

In summary, the development of a user-centric web-based **The Fitness Club** application utilizing React.js represents a significant step forward in addressing the challenges faced by current fitness tracking platforms. By leveraging modern technologies and frameworks, this project aims to enhance user experience, simplify fitness tracking, and promote healthier lifestyles.

The focus on intuitive interfaces and robust tracking functionalities not only meets the needs of fitness enthusiasts but also caters to the growing demand for personalized and seamless fitness management solutions. With a responsive design and efficient data handling, the application promises to facilitate smooth interactions for users, enabling them to monitor their workouts and calorie consumption effectively.

**The Fitness Club** contributes to a more sustainable and goal-oriented approach to fitness, ensuring that users can achieve their health objectives with greater ease and consistency.

## Future Scope

### Social Media Sharing

* Social media integration: Users will be able to share their workout achievements, fitness progress, and personalized routines on popular social media platforms such as WhatsApp, Facebook, Instagram, and other apps.
* Easy sharing options: The platform will provide intuitive options to share workout milestones, calorie tracking achievements, and success stories directly via text, link, or image, making it simple for users to stay connected with their social network.
* Increased engagement: By sharing their fitness journeys, users will help inspire others while also increasing visibility and engagement for the fitness club, ultimately contributing to building a supportive community.

**10.2 Workout and Calorie Comparison Tool**

* Workout comparison: Users will be able to compare their workout routines over time, tracking improvements in duration, intensity, and frequency.
* Calorie tracking comparison: The application will allow users to compare their calorie intake versus their calories burned, enabling them to optimize their diet and workout plans effectively.
* Customizable filters: The tool will offer filters to compare different types of workouts, calories consumed, or even progress based on personalized goals such as weight loss, muscle gain, or endurance building.
* Visual comparison: The feature will display workout progress and calorie management in a user-friendly graphical format, making it easy to track improvement and make informed decisions about future fitness plans.

**10.3 Payment Gateway Integration**

* Seamless transactions: The platform will offer a secure and efficient payment gateway for users to make payments for fitness memberships, training sessions, or health plans directly through the application.
* Multiple payment options: Users will have access to a variety of payment methods, including credit cards, debit cards, and popular online services like PayPal or digital wallets, offering convenience and flexibility.

**10.4 Real-Time Fitness Notifications**

* Instant alerts: Users will receive real-time notifications regarding workout reminders, upcoming fitness classes, personalized goals, and calorie consumption updates via SMS and in-app alerts.
* Customizable notifications: Users can personalize their notification preferences based on specific fitness goals, workout types, class schedules, and calorie targets, ensuring they stay motivated and on track.
* Stay informed: With these notifications, users will always be up-to-date on their fitness schedules, new workout programs, and progress, encouraging consistent engagement with their fitness plans.

## Bibliography/References

1. Node.js. (n.d.). Introduction to Node.js. Retrieved August 15, 2024, from https://nodejs.org/en/learn/getting-started/introduction-to-nodejs
2. React. (n.d.). Getting Started. Retrieved August 15, 2024, from https://reactjs.org/docs/getting-started.html
3. Mongoose. (n.d.). Mongoose v6.1.0: Documentation. Retrieved August 15,2024, from https://mongoosejs.com/docs/
4. Express. (n.d.). Installing. Retrieved August 15,2024, from https://expressjs.com/en/starter/installing.html
5. MongoDB. (n.d.). MongoDB Atlas. Retrieved August 15, 2024, from https:/[/www.mongodb.com/cloud/atlas](http://www.mongodb.com/cloud/atlas)
6. Mozilla. (n.d.). JavaScript. Retrieved August 15,2024, from https://developer.mozilla.org/en-US/docs/Web/JavaScript
7. npm. (n.d.). npm. Retrieve August 15, 2024, from https://[www.npmjs.com/](http://www.npmjs.com/)