Health & Fitness Tracker

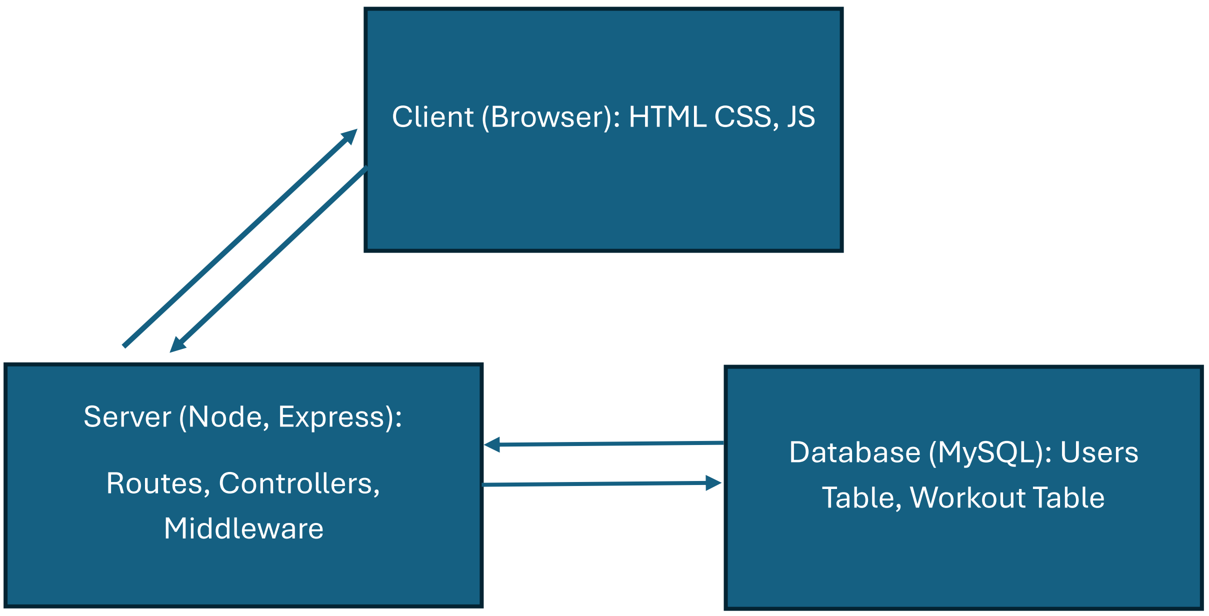
Outlines

This is a Heath Tracker to help users manage their fitness journey. It allows users to create accounts, log daily workouts (includes the duration and calories burned), and track weight loss progress against set goals to achieve either weight loss or weight gain. The application also contains a dashboard with a visual chart to monitor the calories burnt in the last 10 workouts logged. There is also a calculator that calculates the users BMI based on the metrics that they have given.

Architecture

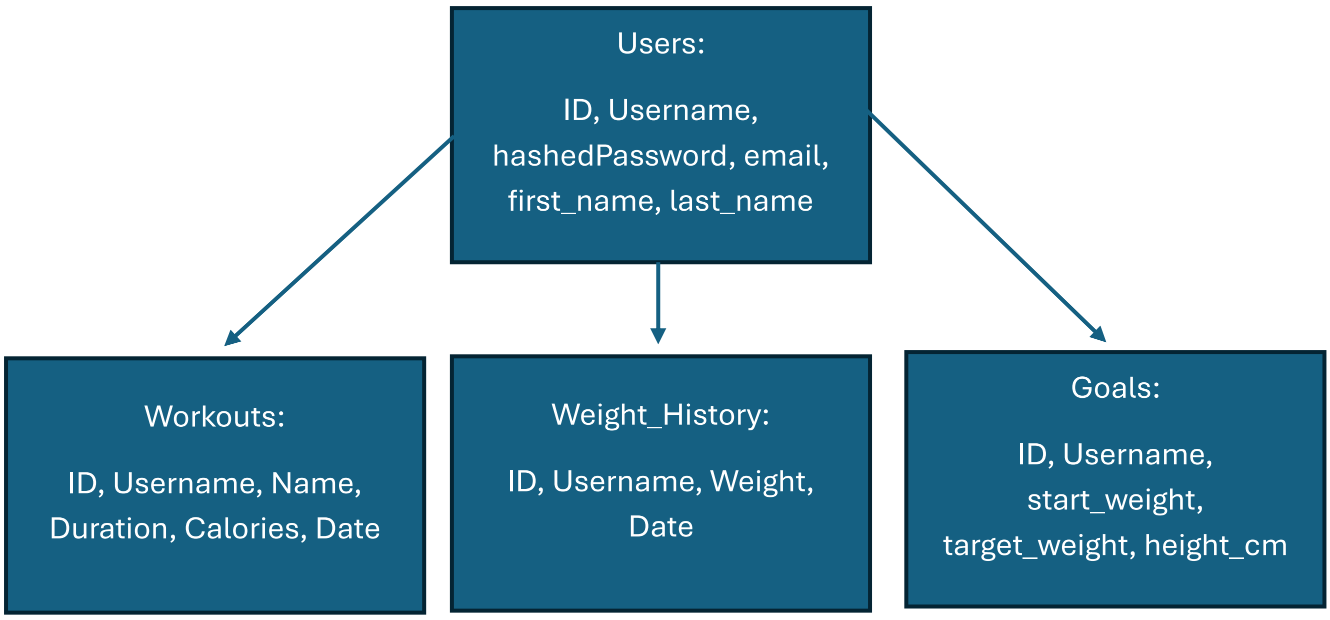
I designed the application using a 3-tier architecture using the Model View Controller (MVC) pattern to keep the code organized and scalable if I wanted to ever create further development.

- Client Tier (Frontend): This is what the user sees and interacts with. I used EJS templates to render dynamic HTML pages on the server side which are styled by the CSS
- Application Tier (Backend): This handles the logic, in which I used node.js with Express module to manage the routing, sessions and middleware like the sanitization and authentication.
- Data Tier (Database): I use MySQL for the database managing. This keeps all the user information like workout history, profiles, and any data in the users accounts all done by the mysql2 library.


Client (Browser): HTML CSS, JS
Server (Node, Express):
Routes, Controllers, Middleware
Database (MySQL): Users Table, Workout Table

Data Model

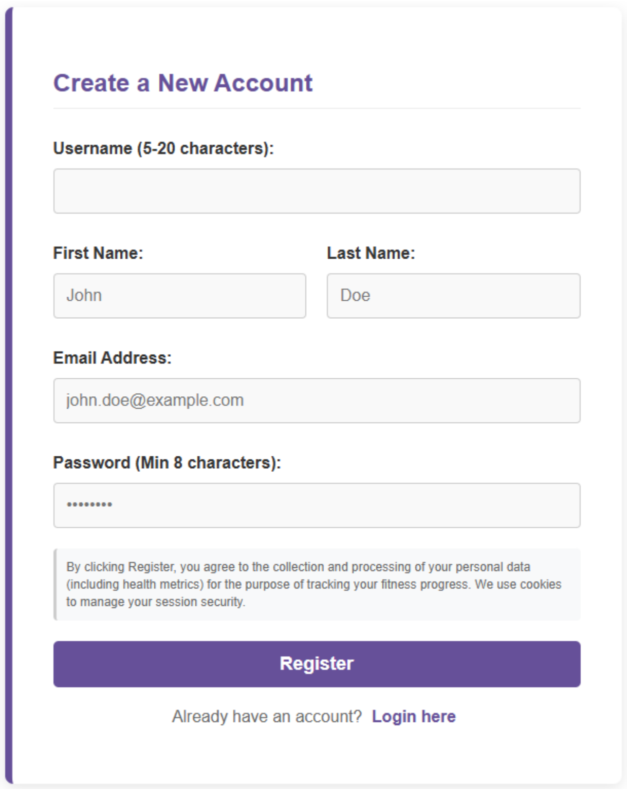
The Users table manages authentication and profile data. The Workouts table stores individual exercise sessions, while Weight_History table tracks progress over time. Finally, the Goals Table stores the user's current targets. All three tables are linked to the Users table via a username key, this enforces data isolation ensuring users can only access their own records and data.



Users:
ID, Username, hashedPassword, email, first_name, last_name
Workouts:
ID, Username, Name, Duration, Calories, Date
Weight_History:
ID, Username, Weight, Date
Goals:
ID, Username, start_weight, target_weight, height_cm

User Functionality

The user experience starts with a secure authentication process, where new users must register for an account using a form that enforces a set of strict validation rules such as a minimum length of 8 characters. Once registered, the credentials are securely hashed and stored, allowing users to log in and create a private session. When a successful authentication is made, the user is redirected to their personal dashboard, which serves as the hub for their fitness journey.


Create a New Account

Username (5-20 characters):

First Name: John

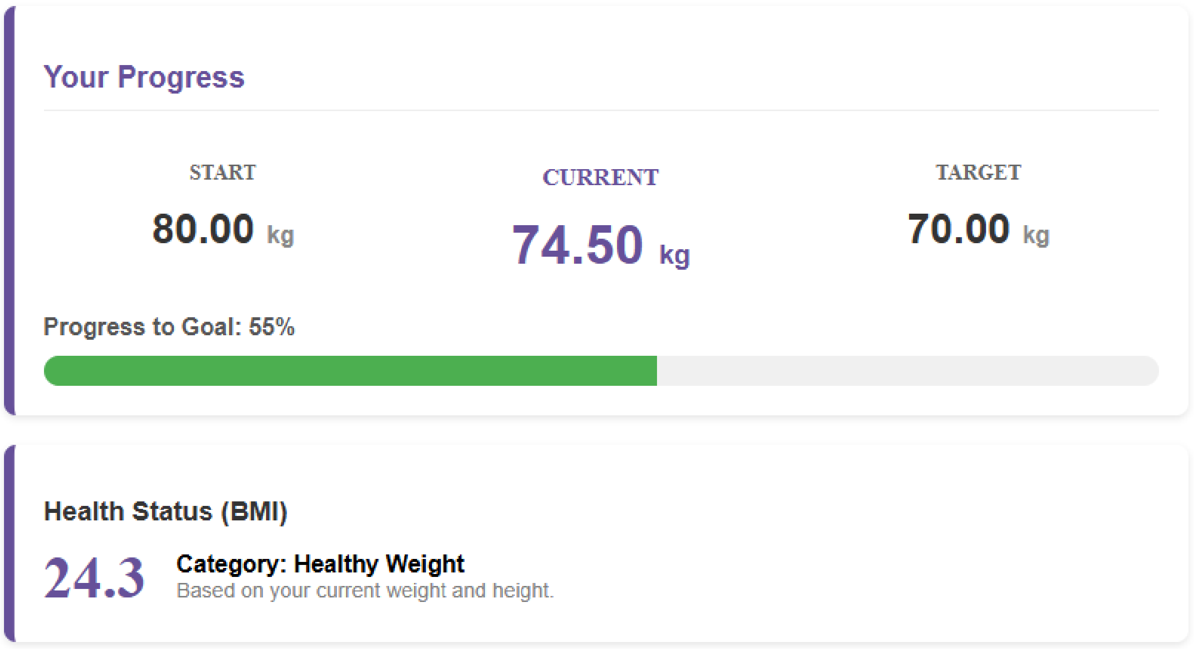
Last Name: Doe

Email Address: john.doe@example.com

Password (Min 8 characters): ••••••••

By clicking Register, you agree to the collection and processing of your personal data (including health metrics) for the purpose of tracking your fitness progress. We use cookies to manage your session security.

Here, the application calculates and displays the users Body Mass Index (BMI) and health category based on their height and weight metrics. A visual progress bar tracks their journey from their starting weight to their target goal giving immediate feedback on their physical progress.


Your Progress

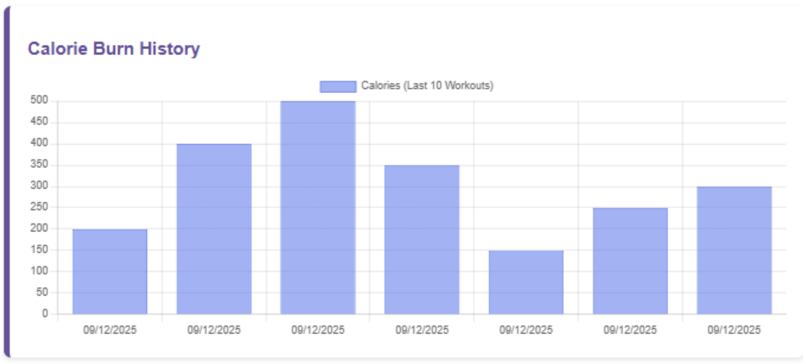
START	CURRENT	TARGET
80.00 kg	74.50 kg	70.00 kg

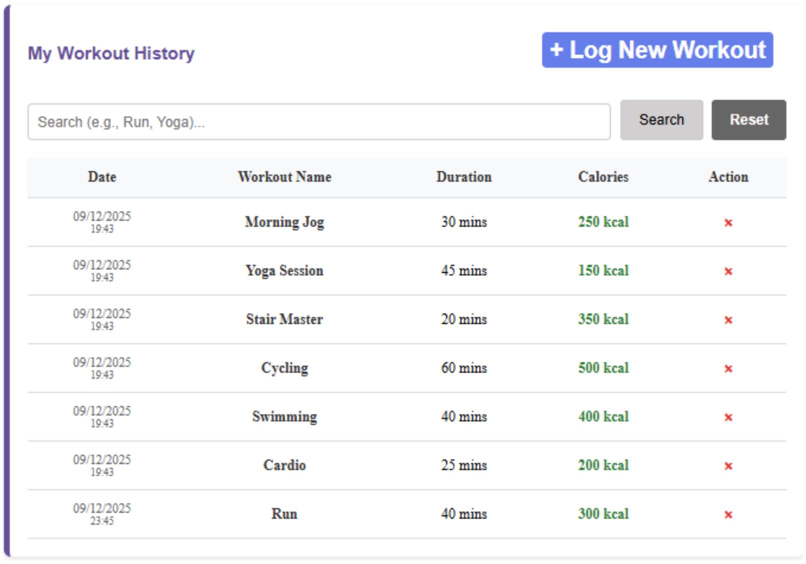
Progress to Goal: 55%

Health Status (BMI)

24.3 Category: Healthy Weight
Based on your current weight and height.

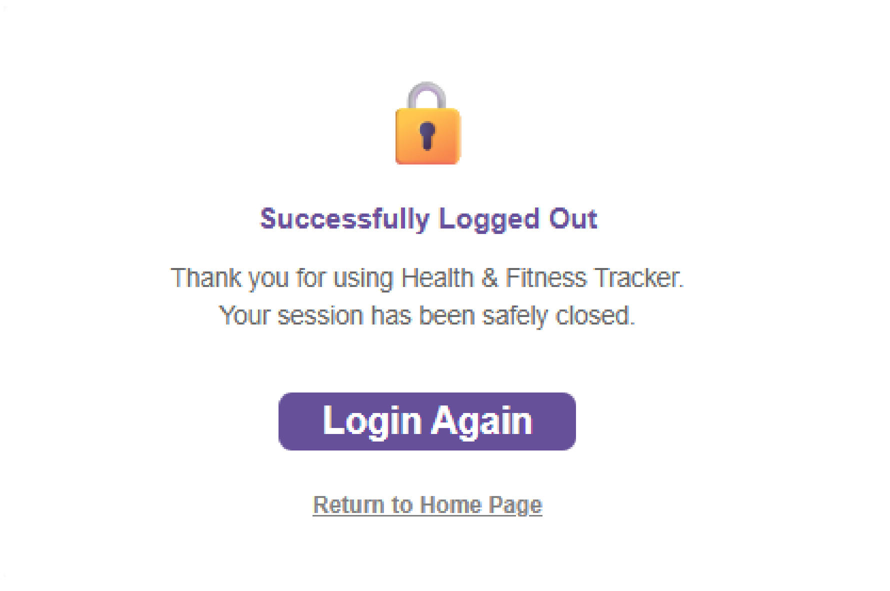
The core function revolves around the “My Workouts” interface, where users can Create, Read, Update and Delete operations. Users can log new work out sessions by specifying the activity name, duration and calories burned. This data is instantly rendered into the history view, allowing users to review their performance over time. To enhance usability, I created a search function to allow the user to filter their history by keywords, enabling a quick retrieval of specific activities that the user may want.





Date	Workout Name	Duration	Calories	Action
09/12/2025 19:43	Morning Jog	30 mins	250 kcal	×
09/12/2025 19:43	Yoga Session	45 mins	150 kcal	×
09/12/2025 19:43	Stair Master	20 mins	350 kcal	×
09/12/2025 19:43	Cycling	60 mins	500 kcal	×
09/12/2025 19:43	Swimming	40 mins	400 kcal	×
09/12/2025 19:43	Cardio	25 mins	200 kcal	×
09/12/2025 23:45	Run	40 mins	300 kcal	×

Furthermore, I integrated some advanced data visualization using Chart.js this feature automatically fetches the users last 10 workout entries via an internal API and renders a bar chart of their calorie burn history allowing users to spot trends in their performance visually rather than just looking numerically. Finally, users can securely terminate their session via the logout function which will stop their session cookie and then prevent unauthorized access.


Successfully Logged Out

Thank you for using Health & Fitness Tracker.
Your session has been safely closed.

Advanced Techniques

One of the main technical features I implemented was client-side data visualization using Chart.js. I wanted to show users their progress without making them reload the page constantly. To achieve this, I moved away from standard server-side rendering and instead built a custom internal API endpoint (/api/workouts). When the user visits their workout list, the page uses JavaScript to secretly fetch their workout data in the background. It then processes the last ten entries and draws an interactive bar chart right in the browser. This makes the application feel much faster and more responsive, similar to a modern single-page application.

Creating this chart required me to design the backend to handle RESTful API responses. Most of the application returns HTML pages, but the /api route is designed to return raw JSON data. A key challenge was making sure this data was secure. I didn't want just anyone to be able to access a user's workout history, so I reused my existing authentication middleware. This means that even though the API endpoint exists, it will strictly refuse to send data unless the request comes from a user who is currently logged in. This allowed me to add advanced features without compromising user privacy.

I implemented a robust Server-Side Validation Layer using the express-validator library. While many web applications rely solely on client-side

HTML attributes for validation (which can be bypassed), I implemented a middleware chain on my routes that strictly validates data types and formats on the server before processing. For example, the registration route validates that passwords meet length requirements and that emails are valid formats. This ensures that even if a malicious user bypasses the frontend interface, the application maintains data integrity and rejects invalid inputs.

Finally, I focused on securing the application against common web attacks. I used bcrypt to hash all passwords before they are stored in the database, which ensures that user accounts remain safe even if the database is accessed. I also implemented input sanitization using express-sanitizer. This tool automatically cleans up any data the user types into the registration or workout forms, removing potential malicious scripts. This prevents Cross-Site Scripting (XSS) attacks and ensures that the data entering my SQL queries is clean and safe.

AI Declaration

The development of this application is my own work. I used AI tools sparingly and only for minor assistance where documentation was unclear. Specifically, I used AI to help find the correct syntax for implementing the Chart.js configuration options and to look up specific JavaScript methods for data formatting. I did not use AI to generate the core logic, architecture, or database structure of the application as I had learnt how to operate and create this all in the labs.