Goldsmiths
UNIVERSITY OF LONDON

Report for Dynamic Web Applications

By Jannat Sultana 33736226

Goldsmiths, University of London
Dynamic Web Applications
Department of Computing

Submission date
16th December 2025

Outline

This Dynamic Web Application enables users to log, view, search, and manage fitness workouts. The application features an intuitive interface for adding workout records, reviewing previously logged workouts, searching by workout type or notes, and deleting workouts as necessary.

The application demonstrates server-side technologies for handling user input, interacting with a relational database, and dynamically generating web pages. It is developed using Node.js, Express, MySQL, and EJS, and follows a traditional client–server architecture.

Architecture

The application employs a standard **Model–View–Controller (MVC) architecture**, implemented with Express.

- **Server:**
  Node.js and the Express framework handle HTTP requests, define application routes, and manage middleware for body parsing, sanitisation, and session handling.
- **Views:**
  The user interface is built with EJS templates, enabling the dynamic rendering of HTML pages from database data. Common layout elements, such as the header and footer, are shared across pages using partials.
- **Database:**
  A MySQL database stores workout data. The server communicates with the database using the mysql2 library and a connection pool to efficiently manage queries. Served from a public directory using Express's static middleware.

This architecture provides clear separation of concerns, facilitating maintenance and future extension of the application.

Data Model

The application utilises a MySQL database containing a workouts table, which stores all workout-related information entered by users. The fields in the workouts table include:

- id (primary key)
- user_id (used as a placeholder for future multi-user functionality)
- workout_date
- workout_type
- duration_minutes
- notes

SQL scripts (create_db.sql and insert_test_data.sql) are included to create the database schema and populate it with sample data, enabling immediate testing after setup.

User Functionality

The application offers the following functionality:

- **View Workouts:**
  Users may view a table of all recorded workouts, which are dynamically retrieved from the database.
- **Add Workout:**
  A form enables users to add new workouts, specifying date, type, duration, and optional notes. Submitted data is processed on the server and securely stored in the database.
- **Search Workouts:**
  Users can search workouts using a search form. The server processes the search query and filters workouts in the database based on workout type or notes, demonstrating server-side search functionality.
- **Delete Workout:**
  Users can delete existing workouts. A confirmation prompt is displayed before deletion to prevent accidental data loss and enhance usability.

All functionality is implemented through server-side routing and database interaction, without reliance on client-side JavaScript frameworks.

Advanced Techniques

Several advanced techniques are employed to enhance security, usability, and robustness:

- **Server-Side Validation and Sanitisation:**
  User input is sanitised using middleware to mitigate the risk of malicious input.
- **Parameterised SQL Queries:**
  All database queries utilise parameterised statements to prevent SQL injection attacks.
- **Confirmation Prompt for Destructive Actions:**
  A confirmation prompt is presented before deleting a workout to prevent accidental deletions.
- **Connection Pooling:**
  A MySQL connection pool efficiently manages database connections.
- **Separation of Concerns:**
  The application logic, presentation, and data access are distinctly separated.**ion**

AI tools (ChatGPT) were utilised during development to support understanding concepts, debug errors, and improve code structure. All code was reviewed, tested, and adapted by the author, and the final implementation reflects the author's independent understanding and decisions.