IT211 Project 1
Score: _____/100 v3.0

Specifications

In this project, you will create a Workout Routine Tracker, a web application designed to help users plan and track their fitness routines. The application will allow users to add exercises to their routine, edit details such as the number of sets and repetitions, and track their progress by marking workouts as complete. You'll learn how to manipulate the DOM, handle events, and work with timers using `setTimeout()` and `setInterval()` to create dynamic and interactive behavior.

This project introduces timing functionality to improve user experience. You'll implement a countdown timer for rest periods between exercises, allowing users to prepare for the next workout. Additionally, you will provide auto-save feedback, notifying users when their workout routine is saved. By the end, you will have built a fully functional workout tracker that makes use of key JavaScript concepts, such as DOM manipulation, event handling, form validation, and timers.

To complete this project, you will need to focus on the following key features:

- Workout Routine Features:
  - The application should allow users to:
    - Add new workouts, which can either be repetition-based (e.g., push-ups for 3 sets of 10) or time-based (e.g., running for 20 minutes). Allow the user to define the exercise name.
      - When adding a time-based workout, the user should specify the duration. Once the workout is started, the application should initiate a countdown timer using `setInterval()` and display the remaining time for that workout. The timer should count down in real time, and when it reaches zero, the application should notify the user that the workout is complete (e.g., "Run complete!").
      - Differentiate between repetition-based and time-based workouts when adding them. For example, a push-up workout would require sets and reps, while a running workout would require a duration in minutes or seconds. The application should manage both types of workouts seamlessly.
    - Edit existing workouts and update their details.
    - Mark workouts as complete with a visual indicator (e.g., strikethrough or dimming).
    - Delete workouts.
- DOM Manipulation:
  - You will dynamically create and manage workout entries using JavaScript. This means generating new DOM elements as workouts are added, updating elements when workouts are edited, and removing them when they are deleted. This will require working with DOM methods such as `createElement()`, `appendChild()`, and `removeChild()`.

- Event Handling:
  - Attach event listeners to handle user actions like adding new workouts, editing existing ones, marking them as complete, or deleting them. Use `addEventListener()` to bind these events and avoid inline event handlers to keep your HTML and JavaScript separate for better maintainability.
- Workout Editing:
  - When a user clicks on an existing workout, allow them to switch into "edit mode," where they can update the workout details. The user should be able to confirm changes either by pressing the "Enter" key or clicking a save button.
- Form Validation:
  - Ensure that your application includes input validation. When adding new workouts, check that all necessary fields (like exercise name, sets, and reps) are filled out. Use the Constraint Validation API to enforce these rules, and prevent invalid submissions from being processed.
- Timers for Time-Based Exercises:
  - For time-based workouts (like running for 20 minutes), implement a timer using `setInterval()` or `setTimeout()`. When the user starts a time-based workout, display the countdown on the screen, showing the remaining time. When the timer reaches zero, notify the user that the workout is complete.

Constraints

- jQuery cannot be used.
- The console can only be used for debugging.
- Implement JavaScript as modules.
- No inline JavaScript can be used. All JavaScript must be contained in mjs files.
- Adhere to the specifications listed above.

Extra Credit Opportunities

- (5 pts) Add the ability to filter workouts by type (e.g., time-based vs. rep-based) or status (completed vs. pending).
- (7 pts) Save the user's workout list using localStorage so that the list persists even after the page is refreshed.
- (6 pts) For time-based workouts, add a feature to pause and resume the timer so that users can manage their workout sessions more flexibly.

Submission

You will submit the commit ID for the commit you want graded. Submit the commit ID on the Canvas assignment.

Tips

- Consider setting the display of some elements to “none” to make elements visible dynamically.
- Given the amount of JavaScript you’ll be writing, consider using multiple JavaScript files.
- Use `console.log()`: Debugging can be easier with `console.log()` statements to track variables and actions in your app.
- Focus on getting the basic workout adding, editing, and deleting features working before adding timers.
- Always remember to clear your timers using `clearInterval()` when the timer is no longer needed to avoid memory leaks.
- Make sure your timers handle edge cases, like stopping at zero or ensuring inputs are valid before starting a timer.

Academic Honesty Policy

All code submissions must be original and authored by the student. Any code sourced from another student, falsely presented as one's own, or derived from third-party websites or generated by AI, will constitute a breach of the school's academic honesty policy.

Daily commits and pushes to your assignment GitHub repository are <u>mandatory</u> while working on your project. Failure to comply will result in deductions from your final grade at minimum, and could lead to academic honesty violations.