**Documentation – Fitness Tracker (Java + Maven + JUnit 5)**

**1. Clean Code Practices**

My project follows several clean coding practices:

1. **Immutability**
   * The Workout class is **immutable** because all of its fields are declared final and there are no setters.
   * This prevents accidental changes to workout data once it’s created.
2. **Validation & Fail Fast**
   * Input values are validated when creating or logging workouts. For example, negative minutes or calories throw an IllegalArgumentException.
   * This ensures errors are caught immediately rather than causing incorrect state later.
3. **Single Responsibility Principle (SRP)**
   * Each class has a **clear responsibility**:
     + Workout: data structure (value object).
     + FitnessTracker: business logic (managing workouts, progress tracking).
     + Main: user input/output (CLI runner).
   * This separation keeps the code easier to maintain and test.

**2. Project Explanation**

The **Fitness Tracker** is a simple Java CLI application that allows users to log workouts and track progress toward weekly fitness goals.

**Features:**

* **Log Workouts**: The user can input minutes and calories for each workout session.
* **Track Progress**: The tracker calculates total calories burned and compares it to a set goal.
* **Set Goals**: A weekly calorie burn goal can be defined, and progress is displayed.

**How it works:**

* When the program runs, the user is prompted to enter workouts or quit.
* Each workout is stored as an immutable Workout object.
* The FitnessTracker maintains a list of workouts and calculates statistics (e.g., total calories, remaining to goal).
* Output is displayed in the console.

**3. Unit Tests**

The project includes **JUnit 5 tests** located in src/test/java/com/example/fitness/FitnessTrackerTest.java.

**Tests Written:**

1. **Test Add Workout (Positive Case)**
   * Verifies that when a workout is logged, it is stored correctly in the tracker.
   * A black screen with white text

     AI-generated content may be incorrect.
2. **Test Invalid Input (Negative Case)**
   * Ensures that creating a workout with negative minutes or calories throws an exception.
3. **Test Goal Progress**
   * Verifies that progress against the weekly goal is calculated correctly (e.g., percentage toward goal).

All tests pass both locally and in **GitHub Actions** CI.

**4. Dependencies**

The project uses the following dependencies, declared in pom.xml:

* **JUnit 5** → for unit testing.
* **Maven Surefire Plugin** → to run tests automatically.
* **GitHub Actions Maven Workflow** → for CI/CD automation.

Dependencies were sourced from **Maven Central Repository** and added inside the <dependencies> block of pom.xml.

**5. Problems Faced**

During development, I encountered and solved a few challenges:

1. **Maven Installation Issues**
   * On macOS, Homebrew failed to install Maven due to system version support.
   * Fixed by installing **SDKMAN** and using it to install Maven successfully.
2. **Running the JAR File**
   * Initially, I got a no main manifest attribute error when trying to run the packaged JAR.
   * Fixed by running the program with the fully qualified class name:
   * java -cp target/fitness-tracker-1.0.0.jar com.example.fitness.Main
3. **GitHub Actions Setup**
   * Workflow setup required configuring the Java version and Maven.
   * Once added to .github/workflows/maven.yml, the workflow ran tests on each push/PR and passed successfully.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Conclusion**

This project demonstrates:

* A fully working **Java fitness tracker app**.
* **Object-Oriented Design** with clean separation of concerns.
* **JUnit 5 tests** covering positive and negative cases.
* **Maven build system** and dependencies.
* **GitHub Actions CI pipeline** to automatically run tests.

**Keira Hancock SD13**