Assignment 5 - Unit, Mocking and Integration Testing

Background

In this assignment, you will automate test and perform quality checks using `GitHub Actions`, a CI (Continuous Integration Tool) platform that runs your build and test. In this assignment, we will only be focused on continuous integration.

You'll act as part of a engineering team responsible for maintaining code quality through automated testing and static analysis (linting).

1. **Workflows** – Automation pipelines defined in YAML files inside `.github/workflows/`.
2. **Triggers** – Events that start the workflow (e.g., `push`, `pull_request`, or `schedule`).
3. **Jobs** – Sets of steps that can run in sequence or parallel.
4. **Steps** – Commands or actions that perform tasks within a job.
5. **Runners** – Virtual machines (GitHub-hosted or self-hosted) that execute your jobs.

Example: A workflow that runs tests automatically on each `push`.

- Read GitHub Actions Quickstart before starting.

Part 1 – Project Setup and Testing Practice

Goal: Practice writing and organizing tests.

1. Download and open the `Assignment5code` project from D2L in IntelliJ.
2. Create a `README.md` file with:
   - A short overview of your project.
   - All your written responses and notes (no PDF submission).
3. Testing tasks – `BarnesAndNoble` Project:
   - Perform both Specification-Based Testing and Structural-Based Testing.
   - Use below to label your test type:

```
@DisplayName("specification-based")
@DisplayName("structural-based")
```

4. **Commit your changes** using:

```
{YourName} + Part1 added BarnesAndNoble Tests
```

Part 2 – Automate Testing with GitHub Actions

Goal: Create a workflow that automatically runs tests and analysis on every push to GitHub.

1. Create a workflow file:

```
.github/workflows/SE333_CI.yml
```

2. Configure the workflow to:
   - Trigger on push to the `main` branch.
   - Run on Ubuntu runners.
3. Add static analysis (Checkstyle):
   - Follow Maven Checkstyle Plugin.
   - Run `Checkstyle` during the validate phase (before tests).
   - Modify settings so Checkstyle does not fail the build on violations.
4. Upload Checkstyle Report:
   - After Checkstyle runs, upload its report using upload-artifact.
5. Add JaCoCo Coverage:
   - Configure JaCoCo in `pom.xml` (as discussed in Week 2).
   - Run after tests and upload its report using upload-artifact.
   - Make sure the static analysis runs before testing.
6. Add a Build Badge:

    - Display a build status badge in your `README.md`.

      See: Workflow Status Badge Docs.

7. Commit and Push:
    - Commit message:

      ```
      {YourName} + added Part2 Workflow
      ```

    - Push to GitHub to trigger your workflow.
    - Ensure all steps (static analysis, tests, coverage) run successfully.

Your GitHub Actions run should generate these artifacts:

- `checkstyle.xml`
- `jacoco.xml` (coverage report)

Part 3 – Writing Larger Tests

Goal: Write integration and unit tests for the **`Amazon`** package.

1. **Explore the package:** Understand how the components interact.
2. **Write two types of tests:**
    - **Integration Tests** → name file `AmazonIntegrationTest.java`
    - **Unit Tests** → name file `AmazonUnitTest.java`
3. For each, include:
    - `@DisplayName("specification-based")` tests
    - `@DisplayName("structural-based")` tests
4. **Integration Tests:**
    - Test how multiple components work together.
    - Likely involves a database — reset the DB before each test using `@BeforeAll` or `@BeforeEach`.
5. **Unit Tests:**

    - Test individual classes in isolation.
    - Use mocks or stubs on all your unit test that rely on external services

6. **Commit your work:**

   ```
   {YourName} + added Part3 Amazon Tests
   ```

7. Push to main branch to trigger the workflow.

Hint:

Before submission, make sure:

- You have made three clear commits for each part.
- GitHub Actions executed successfully.
- The workflow produced Checkstyle and JaCoCo reports.
- Commit messages clearly describe the change (additions, deletions, or modifications).

Grading Breakdown

Criteria	Description
GitHub Repository	Use Git & GitHub properly. Include 3 commits with clear messages.
GitHub Actions Workflow	Workflow runs automatically, includes log output for static analysis, testing, and coverage.
Reports & Artifacts	Checkstyle and JaCoCo reports are generated and uploaded successfully.
Test Quality	Includes both unit and integration tests with strong coverage (aim for 100% where possible).
README.md	Includes link to workflow, brief project overview, and confirmation that all actions passed.

Total: 10%

Submission

- Submit your GitHub repository link on D2L.

- You should also submit your code base.
- Confirm your workflow has passed successfully before submission.