HOCHSCHULE
ESSLINGEN

Major: Software Engineering & Media Computing (SWB)
Lecture: Software Testing (SWB 105 6043)
Lecturer: Prof. Dr. Dennis Grewe

Exercise 10 – UI / E2E Testing

Goal:

In this exercise, you will:

- Design and implement a basic UI for selected FocusFlow functionality
- Apply UI or E2E testing using tools like Cypress, Selenium, Playwright, etc.
- Automate and validate key user interactions
- Gain practical experience in connecting frontend implementation with automated testing strategies

Preconditions / Requirements:

- Test tools (e.g., Cypress, Selenium, etc.) are installed

Exercise 10.1 (15 Points): Implement UI for FocusFlow

Task: Design and implement a simple frontend interface choosing your selected tech stack. The UI covers at least one or more core features of FocusFlow, such as:

- User Login / User management
- Task dashboard listing all existing tasks
- Create, update, delete tasks
- Modify task status
- etc.

Hint: Mock the backend or use an existing local FocusFlow API and focus more on functionality over design.

Deliverables:

- Frontend interface available in code repository and corresponding documentation (e.g., how to start/run, etc.) are available.

Exercise 10.2 (10 Points): Automate UI or E2E Tests

Task: Use a testing tool like Cypress, Selenium, etc. to write automated test cases that validate user flows in your UI. Remember back exercise 7 (BDD) as an inspiration of test case scenarios to implement. Your test should include:

- At least one positive test case (e.g., task creation works)
- At least one negative test case (e.g., error when submitting an empty task)
- Proper selectors and assertions (e.g., testIDs, DOM checks, etc.)

Deliverables:

- UI or E2E test scripts available in your code repository.
- Documentation of your test cases and additional information on how to execute them

Exercise 10.3 (10 Points): Execute & Document your tests

Task: Run your test in a **headless** and **interactive** mode (if supported). Document the test output and note any **errors, flaky behavior,** or **insights.** Reflect briefly on what was easy / difficult about testing your UI.

Deliverables:

- Documentation explaining how to run the tests, output, insights available.

(optional) Exercise 10.4 (10 Points): Integrate Tests into CI Pipeline

Task: Extend your project by setting up **automated test execution in your GH CI Pipeline.**

Perform:

- Create a dedicated workflow for your UI tests
- Ensure dependencies are installed (e.g., Node, Cypress, Chrome, etc.)
- Run tests in **headless mode**

Deliverables:

- Working pipeline file in the repository
- Pipeline output included in the documentation
- Short explanation of the CI setup