Exercise 7 – BDD & Gherkin Syntax

Goal:

Based on the previous exercises, you gained knowledge about the FocusFlow application and its capabilities from an end users' perspective. While we exercised unit, integration and mock testing so far, the definition of end-user tests / acceptance tests is missing. The goal of this exercise is to practice the usage of Gherkin Syntax to use natural language to define use case / acceptance tests. Afterwards, the use case definitions will be implemented using a BDD testing framework.

Exercise 7.1 (10 Points): User Stories Definition

Based on your documents from previous exercises (e.g., requirements, quality model, review, etc.) and your knowledge about the planned functionality of the FocusFlow application, take your use cases from exercise 2 (e.g., task creation, team creation, user registration, etc.) as a basis for this exercise.

Next check the presentation slides of set 07: "ATTD, BDD, TDD" and check the content of the Gherkin syntax.

Task: Take your use cases from exercise 2 and review them carefully with your additional knowledge gained during the last weeks. Use the descriptions to define actual Gherkin scenarios describing functionality expected by a user to interact with the FocusFlow application. Document your definitions as part of `*.feature` file(s) and place it into the appropriate folder (e.g., in Java projects it is typically arranged at *src/test/resources/features/ **).
Finally, make sure your project dependency configuration (e.g., pom.xml, requirements.txt, etc.) file includes the correct test dependencies for a BDD test framework (e.g., Cucumber for Java, behave for Python, etc.). You can check a working setup in the *xyzTesting* repository on Github: https://github.com/dgrewe-hse/xyzTesting/
If you are interested in any guidance about the Gherkin syntax please check the Gherkin guide for best practices: https://cucumber.io/docs/bdd/better-gherkin/

Document your user stories within your project repository.

Exercise 7.2 (10 Points): Implement Steps Definitions

In the next exercise, you will implement the corresponding *steps* definitions for the BDD tests. Create a new **Steps* file (where * reflects to your FocusFlow scenario such as task management, etc.) within a appropriate steps folder in your project structure (e.g., *src/test/java/de/hse/focusflow/steps* folder in case of Java).

Task: For each of the three scenarios of exercise 7.1, implement the corresponding steps implementation of your acceptance tests. You can use the following working setup in the *xyzTesting* repository on Github: https://github.com/dgrewe-hse/xyzTesting/ as inspiration.

Exercise 7.3 (10 Points): BDD Test Automation

In the final exercise, you will finalize the implementation of the acceptance tests / BDD tests. Make sure your BDD tests are detected and executed using your unit test framework with the project (e.g., JUnit, pytest, etc.). For some programming languages (e.g., Java), it might be necessary to create a Runner for automating the test execution as well as to integrate it in your pom.xml definition – for others such as Python, you do not have to do anything else.

You can use the following working setup in the *xyzTesting* repository on Github: https://github.com/dgrewe-hse/xyzTesting/ as inspiration.

Document your configuration and implementation steps within your directory. What do you think makes more sense to execute the BDD test: during the integration or unit test stage of your project? Please document and reason your decision.