TDT4242 Spring 2022: Exercise 2 – DevOps and testing (25 points)

Purpose of the exercise

This exercise is designed to let you practice software development in the DevOps environment and white-box and black-box testing approach.

Context and exercise goals

From exercise 1, you got the requirements of some new features from your peer group. In this exercise, your group will enhance the existing web application by adding new features.

In addition, you need to practice the testing approached learned from the lectures to find bugs of the existing web application and to ensure the quality of the new features you add.

Exercise tasks

Task 1: Develop the new features requested by your peer group in the DevOps environment (5 points)

You will need to implement the new features in the DevOps environment. The existing web application is in this URL: https://gitlab.stud.idi.ntnu.no/aasmuha/tdt4242-base

You need to configure the CI and CD of the DevOps environment. We recommend using GitLab as the DevOps environment. A detailed guide is provided in the README.md file in the repository. You can also choose another DevOps environment if you are more familiar with the other environment.

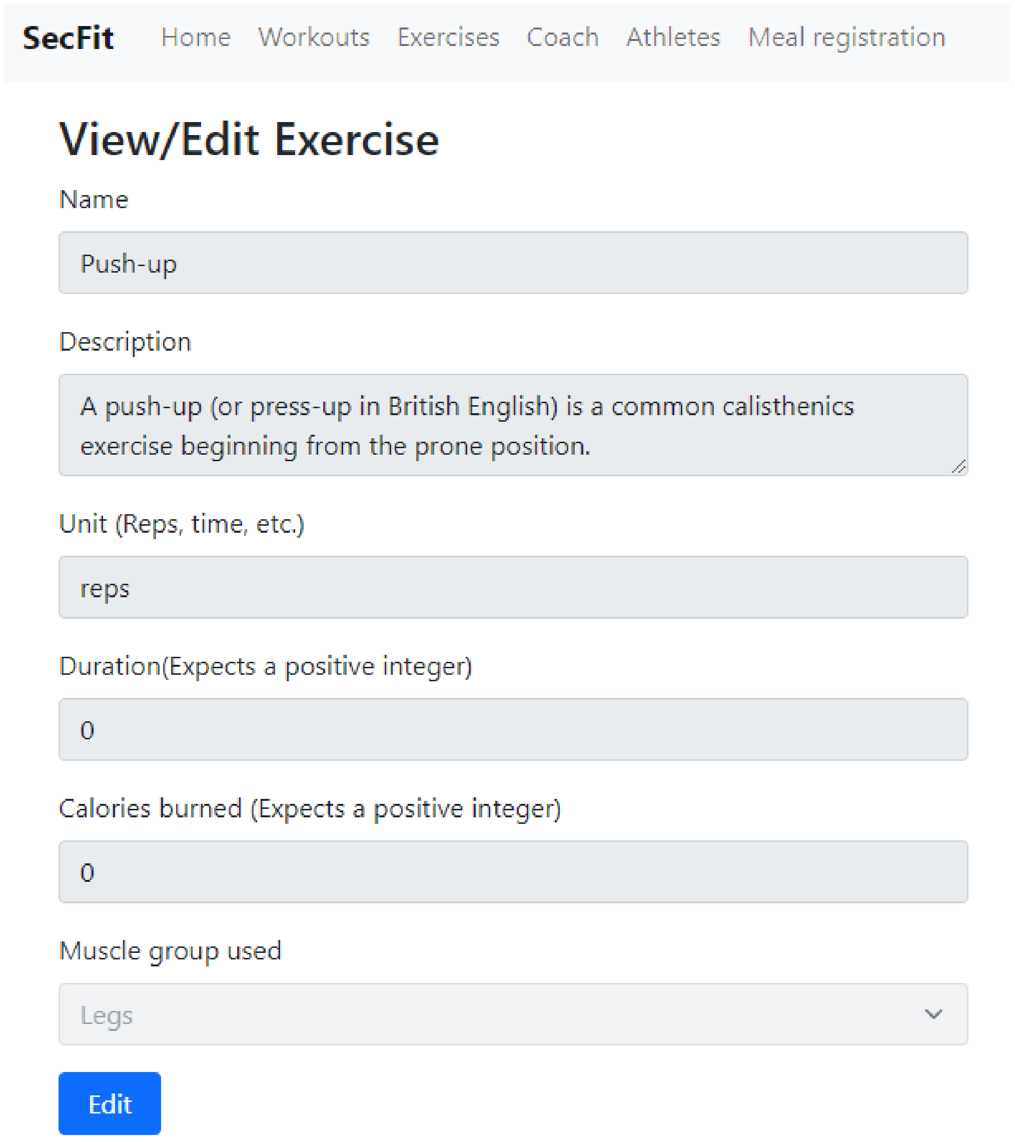
You need to **provide a short explanation of your** design and implementation of the new features in the report. You will also need to explain how you use the DevOps environment to implement the new features. During and after your implementation, you need to communicate with your peer group to let them check if the functions are the ones they want.

Task 2: Write test scripts to have full statement coverage of some functionality of the existing web application (5 points)

For the class **UserSerializer** (backend/secfit/users/serializers.py) and the entire file **backend/secfit/workouts/permissions.py**, you need to write automated test scripts to get full statement coverage. The test scripts should be integrated into the CI and CD of DevOps. The purpose is to let you be familiar with the test automation of the DevOps environment. The statement coverage results should be included in the report. **Hint:** In *backend/secfit/workouts/permissions.py*, the comments of each function detail how the functions are meant to work.

Task 3: Write test scripts to practice black-box test methods (7 points)

You should write test scripts to automate and run the following tests



View/Edit Exercise

Name	Push-up
Description	A push-up (or press-up in British English) is a common calisthenics exercise beginning from the prone position.
Unit (Reps, time, etc.)	reps
Duration(Expects a positive integer)	0
Calories burned (Expects a positive integer)	0
Muscle group used	Legs

- Perform boundary value tests on the register page AND the view/edit exercise page (All fields shown in image above except "Name", "Description" and "Muscle group used").
- Perform 2-way domain tests to test the register page.
- Perform integration tests on the new and implemented features required by your peer group
- Write black box system test scripts to verify some parts of FR5 (listed below)
    - **FR5:** The user should be able to view all of the details, files, and comments on workouts of sufficient visibility. For athletes, this means that the workout needs to be either their own or public. For coaches, this means that the workout is at least one of their athletes' non-private workouts OR the workout is public. For visitors, this means that the workout needs to be public.

Task 4: Practice acceptance test methods to accept the features developed by your peer group (3 points)

Run acceptance tests, which can be explanatory tests, of the features implemented by your peer group to test for bugs. In the report, you need to explain the scope of the test, the test steps and test results.

Task 5: Write data flow test methods with all usage (AU) strategy of a method in the workout gallery (5 points)

Write test scripts for the retrieveWorkoutImages function in gallery.js to satisfy the AU coverage. The test scripts should be integrated into the CI and CD of DevOps. The purpose is to let you be familiar with the test automation of the DevOps environment. If you find any bugs in your testing, you should include them in your report.

Evaluation criteria

Task 1:

The proper function of the new features. As the implementation needs feedback from the peer group. If a group does not give active feedback to the peer group of the implementation, we may deduct 2 points from that group. (5 points)

Task 2:

Full statement coverage of the two functions with automated test scripts (5 points)

Task 3:

Proper test scripts of the black-box testing (7 points).

Task 4:

Proper acceptance tests of the features delivered by your peer group (3 points)

Task 5:

Data flow test methods with the AU strategy (5 points)

Note: You should deliver the report and provide a link to your repository with your new code. The code should also contain all test scripts.