CITS5501 - Test and Automation Project

Document version 1.1 (2018-04-20)
Please check the CITS5501 website to ensure you have the latest version.

Submission deadline: 5:00pm Friday 18th May, 2018.
Value: 20% of CITS5501.

You should construct a python file containing tests, and an accompanying report, to fulfil the description below. You must submit your program using cssubmit.

This project is designed to be done individually, but if desired, it may be completed in pairs. However in this case, the number of browser-based tests must be doubled (6 total), and the report must include a section describing the individual contribution of each group-member. Students' contributions will be assessed individually. Details will be posted in the CITS5501 discussion forum.

You should make your election to complete the project individually or in a pair via the following Google Forms survey, by close of business (5p.m.) on Tuesday 24 April, 2018: https://goo.gl/forms/cTZ4q5Mr1OZbbhVX2.You will need to be logged in to Pheme and/or your student email account in order to complete the survey.

You are expected to have read and understood the University's guidelines on academic conduct. In accordance with this policy, you may discuss with other students the general principles required to understand this project, but the work you submit must be the result of your own effort.
You must submit your project before the submission deadline above. The penalties for late submission are described in the University's guidelines on assessment: For late submissions a penalty of 10% of the total mark allocated to the assessment item must be deducted per day for the first 7 days (including weekends and public holidays). After 7 days the assigned work will not be accepted and will receive a mark of zero (unless an application for mitigation is approved).

Overview

Imagine that you are working as a software tester with a small project team in an organization that is producing an integrated operating environment for a client, where one of the requirements of the environment is a basic to do list, written in Python. The project has acquired some software assets that fulfil this role, but as yet do not have any quality assurance associated with this asset.

Being in a small team, you are responsible for all of the four activities: test design, test automation, test execution, and test evaluation. You have been asked to build a quality assurance process around this basic to do list, given the following requirements:

- The quality assurance process should include tests at both the unit level and the user level. It may also stipulate other quality control processes such as assessments via code metrics. System and integrations tests should be planned for, but are not at a stage where they can be executed.
- The quality assurance process should have well defined metrics that can be tracked as the project proceeds.
- The tests should be automated.
- Your project manager has stated that the main drivers for this quality assurance process are ensuring software reliability, reducing cost, and minimising development time (in that order).

The computer program you will be testing on is the **CITS5501 To Do** application. The link to the download page will be provided in class. It is written in Python using the Django framework and the source code is available.

You are allowed to make reasonable assumptions about the requirements for the to do list, but all assumptions should be explicitly noted. As the rest of the project is behind schedule, you do not have to produce integration tests. However your test assets should be designed with reuse and extension in mind.

Deliverables

For this project you should produce an overview of your quality assurance process. It should include:

Part 1) (50%) A test suite, including:

a) Unit tests. Provide at least 3 (non-trivial) unit tests. These should test the specific functionality of the program, rather than the framework it is written in — e.g. it is not necessary to simply write a test confirming that data is really written to the database, but it is useful to write a test ensuring that methods work as expected and that invariants are preserved.
b) Browser-based tests. Provide 3 tests using the Selenium tool to simulate user behaviour. An example is provided which demonstrates logging in, ensuring that the user is on the correct page, then logging out again. These tests should simulate real workflows a person would conduct when using the software, and should test the expectations a real user would have about the software.

These tests should be included in your modified version of the 'tests.py' file found inside the 'todo' directory. Your modified version of this file will form the code portion of the submission; you do not need to modify or submit the rest of the project code.

Two sample tests are included with the project: one unit test and one browser-based test to simulate user behaviour. These tests can be found in the 'tests.py' file within the 'todo' directory, and they use the testing structure of Django, which is a popular web-app development framework. You will not need to alter any code other than this testing file, but you will need to read and understand the code of the project in order to write unit tests for it.

The 'README.txt' document within the project describes how to install and run the project, and run its automated testing suite. These instructions apply specifically to the computer science lab machines, but the project should run capably on your own computer; however, support and assistance will only be provided for running it on lab machines.

These resources may be helpful:
Introduction to testing in Django - https://docs.djangoproject.com/en/1.9/topics/testing/
UI testing with Selenium - https://django-selenium.readthedocs.org/en/latest/

Part 2) (50%) A quality assurance report, including:

a) A description of the tests you have created, and which functionality they test. For the browser-based tests, this should describe the expected behaviour of the software, and how you are testing that this matches the actual behaviour.
b) A description of how you selected these areas to test, and your rationale for how they contribute to quality assurance. Describe the coverage of these tests, and the metrics you would use to assess this coverage and the quality of the software.
c) A basic plan describing how you would go about extending these tests to provide further test coverage.
d) A description of any bugs or errors found during your testing, and a description of how these bugs should be reported and tracked if this were an ongoing project.

This report is to be submitted as either a plain text document, a Microsoft Word file, or a PDF.

Submission

Both parts should be submitted electronically to *cssubmit.* Part 1 should be in the form of a the 'tests.py' file. Part 2 should be submitted as either a plain text file, a word document, or a PDF.

Assessment

You will be assessed on:

- Completeness: provision of the correct number of tests, and addressing all criteria of the report;
- Correctness: whether your tests successfully test the things you are aiming to test, and how well both the tests and your report reflect a correct understanding of the practice of testing as it applies to this project;
- Clarity: how well the code and the report articulate your intentions and understanding.

Good luck!