COSC 442
Software Quality Assurance & Testing
Spring 2018

Project #6 – Web Application Testing using Selenium

"Automation does not do what testers used to do, unless one ignores most things a tester really does. Automated testing is useful for extending the reach of the testers work, not to replace it."

- James Bach

Directions: This project has 4-*ish* tasks that will guide you through testing web applications using a popular scripting/capture-replay tool, Selenium. This is an individual project, however, students are highly encouraged to discuss/help others, in general terms. I have created a Slack channel for general discussion for this project; I encourage you to use it to discuss problems encountered. Nevertheless, submitted projects must reflect the work, understanding and analysis of the individual student. **Please read through all instructions prior to starting**.

Goals: The intention of this project is to gain an introduction to testing web applications utilizing a scripting and capture-replay tool to capture, develop and automate acceptance test cases. In addition, this project should continue to familiarize yourself with several software development tools. Secondly, this project is again intended to put the process of testing into the context of the software engineering lifecycle by developing test cases from a requirements/design artifact (e.g., a UML Use Case scenarios). In doing so, you should further develop skills in test case design, program specification, unit testing as well as practical experience in programming and software development tools within a larger software engineering process.

Points: 10

Deadline: Monday, May 7, 2018 11:59pm with all project artifacts checked into your GitHub repository.

Language/Environment Requirements: Selenium IDE add-on for Firefox should be used for this project. Your GitHub repository must be used for this project. You should create a new repository, named *cosc442-lastname-project6* (where *lastname* is your actual last name) and add me (cosc442spring2018) as a collaborator.

Task 1 – Setting up the Testing Environment. The Selenium testing environment that will be used to capture-replay use case scenarios will be done using the Firefox web browser, the Selenium Firefox Extension, and the Selenium IDE. Follow the installation instructions for Selenium at http://www.guru99.com/install-selenuim-ide.html.

Task 2 – Using Selenium. As demonstrated in lecture, Selenium is a software testing framework for web applications that allows for capture-replay authoring of tests without the need to directly develop test scripts. There are a number of good tutorials on how to use Selenium, including http://www.softwaretestinghelp.com/selenium-tutorial-1/ and http://www.softwaretestinghelp.com/selenium-ide-download-and-installation-selenium-tutorial-2/ (highly recommended), http://toolsqa.com/selenium-ide/selenium-ide-features/, http://searchsoftwarequality.techtarget.com/tip/Tutorial-Introducing-Selenium-IDE-an-open-source-automation-testing-tool, etc. If you prefer video tutorials, check out any of the YouTube tutorials at https://www.youtube.com/results?search_query=selenium+tutorial+for+beginner.

Task 3 – Web Application Under Test. There are a number of realistic example web applications available to practice developing automated test suites. For this project, we will use the http://phptravels.com/demo/ website, that includes a user frontend, an administrator backend and a supplier backend for a fake travel agency. You should take some time to explore each of these portals that will be tested in the following task.

Task 4 – Developing Test Cases and a Test Suite. Using the Firefox Selenium extension, you should develop (i.e., capture-replay) test cases for the following Use Case scenarios:

- Successful and unsuccessful user login
- Successful and unsuccessful admin login
- Successful and unsuccessful supplier login
- User login and flight search from BWI to MKE
- User adding a tour to their wishlist
- User changing their password
- Admin adding a new tour

Each of these developed test cases should be appropriately/informatively named (save as an HTML file) and make appropriate assertions. Upon completion of these test cases, save the test suite (also as an HTML) and ensure that the entire suite runs successfully.

***Anti-Task* – Exporting Selenium Test Cases/Suites as Junit Code.** Typically, I would require you to export the Selenium test cases/suite to Junit code and run it through Eclipse (see https://www.youtube.com/watch?v=oc0XX27csk8 for a good tutorial) to complete this project. However, there seems to be a problem (i.e., a pairwise interaction fault) between Selenium's current plugin and Firefox. For this task, simply remember that, in industry, you would use the Firefox Selenium extension to generate the Junit (or other language) code as a time-saving approach and then modify the code as needed.

Grading: This project will be graded out of 10, all allocated towards Task 4 for the captured test cases and test suite.

Submission: You must submit all test cases and the test suite (as HTML files) to your GitHub repository. Failure to submit the project following these guidelines may result in your project not being graded.

Note: There are at least 3 Easter eggs (this isn't one of them) in this project. If you find one, let me know via direct message on Slack. Happy hunting! ☺