**Project Two – Summary and Reflections Report**

Drew Townsend

Southern New Hampshire University

CS-320: Software Test Automation & QA

Professor Rangitsch

June 20, 2021

For the past few weeks, we have been learning about and creating unit tests for modules developed for a mobile application. Each assignment is similar in nature, however the requirements for each module has been different. Specifically, we have been creating unit tests using the JUnit 5 library, which really simplifies the creation and automation of testing.

The software requirements must be at the front of your mind while testing the various modules. We need to ensure the requirements are met, and the best way to check that is to run tests against the software. For each assignment I made sure to test the requirements specific to those modules. The first module created was the Contacts module. This module required multiple variables to hold data, however this data had certain limits. For example, a contact’s phone required exactly 10 digits and could not be null. Therefore, I created a check for the contact number, and then tested to ensure that check worked.

setPhone function with checks:



And my JUnit 5 test for the phone:



What these examples show are my functions to set the phone number for the contact, and that set function checking against the software requirements. With my JUnit 5 tests, I am making sure the contact creation fails when using a phone number that is either too long, short, or null.

Without adding too much code to this report, I can affirm the other two modules, the task and appointment modules, were set up very similarly. The task module required an object containing a unique ID, a name, and a string. Each variable had limits, and I tested each limit to ensure they failed when breaching that limit. The appointment module required an object with a unique ID, a Date object, and a description. The date object was a little more complicated to test, but the appointment date couldn’t be set before the current time, and it couldn’t be empty. I went through and tested each of these variables’ limits until failure. Using JUnit 5 tests I also tested the service classes for each module, which would contain a list of the different objects.

I ensured that I had 100% passing for each test before sending the modules, and I covered each requirement when creating the tests. The work can be monotonous, but it’s certainly necessary and worth it. My experience with JUnit tests was great, and I had to ensure my code was technically sound and efficient when sending the modules off for grading. For example, I created functions for common tasks to be done by each object’s service. This is to reduce code redundancy and increase code readability. Although this task is simple, it does make the code more technically sound, and can also prevent errors. Along with this, I implemented the requirement checks into my variable “setters”. This lessens the number of lines of code, which makes my code more efficient and readable. Here is my code when creating a JUnit 5 test for the TaskService class:



My constructor calls the variable “setters”, and the variable “setters” checks the parameters against the requirements. This makes for relatively clean code.

It was important that we understood the various software testing techniques when writing the tests. The main techniques I used for this semester were unit, static, and dynamic testing. A unit test tests specific components in the application. Each component is isolated from the others, to ensure errors aren’t leaking from other units. Static testing is when you’re manually or automatically reviewing the code without actually executing any code. I did this type of testing before running my applications to try and find any syntactical, technical, or efficiency errors. Lastly, dynamic testing refers to the execution of code, and ensuring that output is expected. Most of my testing was done using dynamic and unit testing.

There are more testing techniques that I didn’t need to use, like integration, system, and acceptance testing. Integration testing is when you test your application by putting all of the modules or units together and ensuring they work as expected. System testing is when you run the application and objectively test that the entire system passes functional and non-functional requirements. Because of this objective outlook during system testing, sometimes an independent team is brought on, as the development team could be biased. Lastly, acceptance testing is done to ensure the main objectives or requirements are met by the software. The most common type of acceptance test is the user acceptance test. This is where the users, themselves, test the software to ensure their needs are met. This makes sense, as these users are going to be the people that know how the application should work, and they are an outside team.

As mentioned briefly, bias plays a role in software development. Whilst testing my own code, it can be difficult to remember all the various tests you need to create. I think this is a huge reason documented requirements need to be given to a development team. By creating individual tests for each attribute and object, I employed caution and didn’t let my guard down.

It can be easy to keep your bias and think, “I know what I’m doing. I wouldn’t make this easy of a mistake.” By rejecting these thoughts and creating tests anyways, you are practicing caution and limiting bias. I can absolutely see where veteran software developers don’t feel the need to create simple tests for their code, as they may think it’s a waste of time. This is why I feel independent teams are necessary for testing, if possible.

As a relatively new developer in the field, keeping my discipline will be important when showing my work. Shortcuts tend to lead to code that is harder to read, fix, and add onto later. So, you may as well create high-quality code from the start. To avoid technical debt, I’ll create efficient code where functions are created rather than reusing the same lines of code. I will add comments so other testers and developers understand why I’m doing what I’m doing. Also, assessing vulnerabilities via dependency checks will help increase security on my code, when necessary. I’m very grateful for what I learned throughout this course, and I definitely think it will make me a better developer from here on.