Summary & Reflections Report

CS- 320: Software Test Automation & QA

Dakota Keyes

**Introduction**

This project was quite time-consuming and challenging for me to complete. Out of the languages I have worked with, I feel the least comfortable with Java. Furthermore, I have never tested code before, so most of this project was all new ground. Moreso, the testing was the difficult part for me. If we were only prompted with the tasks and not the testing portion it would have been much simpler for me to complete and less time consuming.

This summary/reflection report provides an outline of the Java unit testing approach I utilized to develop the mobile app for the client. It includes my experience creating and implementing JUnit tests, and outcome/coverage and quality of the JUnit tests I developed. This document discusses the testing techniques utilized in the mobile application, and other techniques that were not implemented, and the effects of not exhausting different software testing techniques. In conclusion, I will discuss my process while working on the project, how to limit biases within the code review, and importance of the quality of the outcome from developing and testing code.

**Summary**

For the mobile application we were prompted to develop unit tests for different Java files, Contact, Task, and Appointment classes. To guarantee the tests were implemented correctly, I utilized a mixture of black and white box testing techniques. The black box testing techniques included entering the required data and confirming that the correct output was produced. The white box testing concerned examining the program’s code to make sure that the logic was comprehensive and complete. I also used JUnit test cases for edge cases and examined the syntax for errors. In doing so, I was able to increase the coverage of my code, verifying that it was effective and efficient. I struggled with implementing some of the tests for the Task files. Once I got the tests to run correctly, I realized the coverage was very low. Even though they did not have errors, they were not even testing any of the functions or components that I was hoping.

The overall experience writing the JUnit tests was a struggle. I struggled to write the files and test files, so they ran and testing the correct aspects. Without help from the teacher, and researching other resources like StackOverflow and GitHub, correctly write and to ensure my code was technically sound. I had trouble verifying that the logic was correct and that the expected outputs were produced. To ensure readability of the code, I employed different methods, such as reducing redundant code and refactoring. I was able to access different modules and external libraries for JUnit assertions and was finally able to get the results and coverage higher than before.

**Reflection**

For the testing techniques in this project, I incorporated black and white box testing. Black box testing involved inputting data and then ensuring the correct output was achieved. The white box testing technique inspected my code and checked the logic and made sure it was correct and effective. Lastly, I employed JUnit test cases for edge cases which helped examine and reduce errors within the code.

There are many other types of testing techniques that could have been implemented in this mobile application. Some of the software testing techniques that I did not utilize in the mobile application were security and performance tests. Security tests help identify and reduce the potential security risks that are possible in almost every software system. Performance tests measure different aspects of the system. These aspects calculate the speed of a program, and the responsiveness of a system, and

There are different practical uses for each of the software testing techniques I used in the mobile app. Each of these are useful to verify correctness and efficiency of the code. Testing techniques can be used to uncover errors, bugs, and identify edge cases. Some of the implications of not employing different testing techniques is not addressing security risks that may go unnoticed, and not checking the performance and responsiveness of the system.

Initially, this project made me irritated not being able to correctly test my code. I struggled with the syntax and JUnit tests. However, I tried to project remain positive and research different aspects to understanding the implementation of the tests and how to check the code. I limited my bias in the review of the code, through a combination of code reviews and unit tests, this combination helped to examine the components and functionality more thoroughly within my code. As a developer, it is crucial to be aware of the impact of potential bias and to be meticulous when testing code for quality.

In conclusion, having a commitment to the quality when writing and testing code is so important to the result and overall quality of a project or system. Developers need to avoid cutting corners and thoroughly write and test the code to avoid technical problems, low-quality releasable, and bugs and defects within a project. Code reviews and unit tests help to examine my code and minimize errors within the code more thoroughly. If I was going to develop a complete mobile application, I would have expanded on the files and would have wanted to enable some security features and then utilize unit tests to ensure the application was safe and security for the users and privacy and data leaks would not happen. I have never coded any intensive security features and have not tested for them either. I do not know how exactly I would develop my code to be secure, but I know it would be a necessary non-functional requirement of a mobile application like this one. Overall, this project was frustrating for me, but I persevered and kept working at it. In the end, I was so relieved that I was able to correctly write and develop code and test the code. Java is my least experienced language, and I have never done any form of testing, so this project was quite difficult and time-consuming for me to complete.