Cayman Casteel

CS 320

25th February 2024

Follow-Up Summary and Reflections Report

In this report, I will give a follow-up summary and report of how I analyzed various approaches to software testing based on requirements and applied appropriate testing strategies to meet requirements while developing the mobile application for the customer.

In my unit testing approach for each of the three features, I heavily referenced the requirements given to me to make sure that each attribute of the features were tested. Using the requirements of having a Task only be deleted with reference to a Unique Identifier, I could add in that the delete method only could work if the ID matched. This also allowed me to check if the task was deleted by searching for the Unique Identifier again and seeing what returned. Following this same procedure of evaluating the requirements closely for each test allowed me to have multiple tests with an overall professional quality.

Writing the JUnit tests also allowed me to continue reviewing the requirements in detail to keep consistent with making sure I have completed all requirements. Running the JUnit tests allowed me to make sure my code was technically sound my showing that every test marked with “#Test” passed. I made sure that the code that I wrote for these tests were efficient by removing any unnecessary lines or parts to the code, keeping it short and concise. This also greatly improved readability as well as adding in line notes for future reference to myself or others.

The software testing techniques I used in this project included manual testing, automated testing using JUnit testing, and System testing. For manual testing, I read through the code I wrote to look for errors or typos in the code. This took the longest since I wanted to make sure I did not miss anything. This technique is used again after other testing methods. The automated testing I used included the JUnit testing which told me where errors were. This was also a system test since it tested all components in the system I created. Other software testing techniques that I did not use during this project include Security testing and Performance testing. I did not need these since security testing check for vulnerabilities in security, however, although this system compiles and runs, this system is not complete and does not have any dependencies that could pose a threat if breached. I also did not need to use performance testing, again for the same reason of this system not being fully complete. It is also a very small-scale system where performance does not show any limitations to it running.

During the mindset I had while working on this project, I employed little, but some caution. I knew that this system must perform and be complete, however, I also knew that this system was not going towards an end user where it must perform for them. Instead, it was to test my capabilities in creating the system. Its important to appreciate the complexity and interrelationships of the code I was testing because it gets you more into a mindset of making sure everything works. Even in a simple system like this, it must work like clockwork, and appreciating that makes a true full system seem divine.

I can understand the testing code that the same person has written might show bias to let some things pass on even when they are failing. Unless you are a self-critic, this can apply to a lot of people. However, when testing the code I wrote, I felt no bias as my end goal was to have code compiled and working, not words in an IDE. This continues the importance of being disciplined in your commitment to quality as a software engineer professional. It is very important to not cut corners in the work you create and making sure that you are providing quality work even if it comes after a deadline. I plan to avoid technical debt in the field with the incentive of knowing that I will continue to be employed by a company if I have quality and timely work that is also efficient. I will also be honest, being paid should be more than enough to keep developers producing quality work!