Elliot Putnam

Project Two Summary and Reflection

CS-320 Software Test Automation & QA

12/13/2023

Summary and Reflection

For this project, I was tasked to develop three classes based on requirements set by Grand Strand Systems. The three backend services that I provided for the mobile application were contact, task, and appointment services. To ensure that I was completing all of the requirements that were asked of me, I carefully took notes. Small details such as ‘contactId shall not be changeable’ meant that I should add a final indicator [private **final** String contactId] to keep other blocks from altering it. To make sure my code was correctly written and performed as expected, I wrote test cases that did just that. For each class, my code coverage was above 80%, meaning that my code was designed thoughtfully with quality and intent on the mind. I had to be through when writing test cases, because when more variables are added they need to be tested. I made sure to test each variable in the parameters field for empty, null, or invalid. Doing so allows other developers to read my test cases and understand what the expected result is, along with what it is testing for. In my test class AppointmentServiceTest, line 19, my test ‘testAddAppointment’ was designed to add an appointment using the AppointmentService class, and check the parameters name, date, and description. If all three parameters met the conditions, the appointment would be added.

I used a testing technique such as boundary value analysis. I tested certain criteria to be sure that the values did not exceed a certain range in either direction. This is useful in software development because it validates input and ensures code can continue properly with defined parameters. I had to think deeply about what I was expecting the tests to do. I embraced caution when considering the edge cases and scenarios that were not incredibly clear. You must assume the user has no clue on what is going on and will enter something against their direction. You don’t put guardrails up because you assume people don’t understand driving, but in case something goes wrong.

Cutting corners when working in any profession is never good. Software developers must adhere to standards and respect their clients. To cut corners as a developer means that millions of people may be at risk from the lack of consideration. Clearly, not every developer sees applications that reach the masses, or even receive any input from people, but the implications are all the same. Laziness leads to leaks, which lead to company risk and danger. In the field, I have always viewed myself as slower than most. This stems from my commitment to producing the best product I can, which means that occasionally I am re-writing blocks a few times over.