Summary

Upon completion of the seven modules and coming close to the final module. I can say that it was a difficult road and that unit testing is important in the making of software answers. I noticed that these answers help take away the errors in any given or making of code, it also helps improve the quality of the code by searching and narrowing down what the problem/error is. A lot of times I noticed we were aiming to get the projects done close as possible to the requirements, the unit testing helped out greatly with this. For context, the contact service had loads of certain requirements that needed a unique ID string that did not exceed 10 letters and was not null. The thing about it too, it was asking for the first and last name that could not be greater than the 10 letters. However, the phone number had to be equal to 10 digits, not less or greater. The address had to be less than 30 letters and not to be null.

Putting all these specific requirements into effect really was a long process but it gave insight into what to expect from a customer and how to fulfill their requirements. The quality of my test was good, I was trying my best to test each field and area to make sure that the validity was good in each test. I still have a lot to learn and improve on within the Junit area, but so far my experience with Junit was good.

Most of the time I was checking to see if the ID was too long or null. A way to check was creating a test for it to see whether or not it was too long. This made checking it a lot easier and faster to move on to the next topic.

Reflection

After completion of the 7 module and soon 8, there is a lot to learn and relearn. One of the methods that I expanded on was dynamic testing. The meaning of the testing is within its name, so it would test the behavior of the code. This would find weak spots within the code as it was running the code. Another method I tried was static testing which is basically testing the code without running it. Sort of like when typing out the code in eclipse it will tell yo if something is missing or needed or if an error is present. A key difference in dynamic method is it doesn’t detect ID too fast or easily.

Throughout the term I was trying my best to finish all the test cases. When testing these cases I was double checking the various, as it was important to make certain requirements to make the code work as it should, taking into effect the relationships and complex of the code. I think all in all, to understand this it takes a lot to really understand what every part of the code is supposed to do and what it needs to do. For example the contact service had lots of specific requirements to be met as well.

I believe that when striving your best to get the best quality it comes from discipline and as a software developer it is a need. With disciplie can make the process of making the code more efficient that would have less holes or errors. An important aspect is not to cut corners when making the code. Cutting corners can lead to lots of money to be loss that can be avoided if the developer had discipline and did not cut corners. Perhaps cutting corners can even lead to a loss of a life or damage to a person. So it is important to refrain from cutting corners and to always remember to test the code as many times as needed to ensure it is the best code possible to send back to the customer, that way they get the best quality version of the application. If following all this and taking into account of what I learned from these modules then I hope to be the best software developer I can be, so I have much to thank from learning from this term and that I will apply to the next future terms.