Shane Griffith

CS-320

8-9-2021

Week 7 Final Project

Without fully understanding the requirements of a system it would be impossible to even explain to someone else what the system does, let alone attempt to create any test scenarios for it. In my opinion, the first step in any project would be for the developer to fully understand not only their portion of the project, but any other part of the project that might affect anything they are assigned to. Without this information, it’s possible that something vital piece of the puzzle could be missed because someone did not understand the scope of the project, but that is only one step of the process. On the other side of things, after a piece of the project is finished, it needs to be thoroughly tested. This was something that I think I missed in the first couple milestones for this project. I am going through this process right now in my professional life. We have a project that is set to go through an acceptance test period at the end of the month. So, what I am tasked with doing is created a test environment to ensure that the acceptance criteria for the project are met prior to us going through this acceptance testing. Without this, we risk going into the final step of the process essentially blind and not knowing whether the system is ready for hand off or not. Even though it might be a much simpler process, I tried to take that same approach here for this project.

I made sure that my code was technically sound by doing a few things. First, when going through and creating the classes I used constructors for each section of the arguments that were required for that class. Any example of this, would be lines 15 – 27 in the Task.java class. Personally, I also feel like there was little wasted space in my program as well in that I tried not to repeat any unnecessary code. Any example for this would be the exceptions used in all the class files. Instead of writing multiple lines for the null check and character checks, I implemented them into one if statement using the && operator.

The first programming language that I really started to understand on was C++ late last year and early this year. I have not really had a ton of experience in JAVA before this course, so that has been a challenge in and of itself. In C++, you are required to manage memory in unique ways, so these are techniques I have learned along the way and tried to implement them to the best of my abilities during this course as well.

Class partitioning is the main technique that I implemented here into this project. What I attempted to do was to break down each part of the class that was necessary to its functionality and test each one individually by creating Junit Test cases. Some testing techniques that I did not use were things like decision tables since I was able to test everything required without them, I did not see the point for their implementation. I could have used value analysis for the required restraints to the variable that were given which depended on which class and which variable was being referenced in any given class. I found it much more effective to use the class partitioning technique in this instance because of the way that the project requirements were laid out in the first place. It essentially required us to test each function of the class in the first place. So, I thought that the decision was obvious and, in a way, made for me.

My mindset when employing caution to this project was centered around whether whatever function or line of code, I was about to write was getting me closer to the finished product or not. The finished product being a project that meets all the requirements. During the review of the code, the only thing that I considered was a couple of things. 1. Did the code compile, if yes, then did it work. Which again, in my opinion was verified through the Junit testing that was required as part of this project. I feel that being in a project implementation role in my current professional job (while I am not necessarily a software developer or engineer yet) has given me the experience to know that ultimately, a successful project for the customer and for the implementor are really the only thing that matters. Whether this means you must call out things to the customer that they want in the project that either cannot be done, or don’t make sense to be done. Sufficient system testing is what has taken up the most of my time in these projects over the last few years. Without that testing, we would have ended up in a final acceptance test scenario with some piece of the system not operating as intended. Whether it be on the customer side, or the implementors quite frankly to me is irrelevant. It needs to be called out and corrected to ensure a successful implementation. I don’t necessarily buy into the mindset that all technical debt is a bad thing especially if you are in a true development process. I take this idea from one of my personal Idols, Elon Musk. For example, in many of his talks about the design process he employs at his companies, he preaches the concept of intentionally taking things out of the design just to add it in later. The biggest advantage of this is to ensure that basis of whatever problem you are trying to solve before adding back in the “nice to haves” or the icing on the cake and other less mission critical pieces of the puzzle.