Ethan Hutchison

CS320 Project 2

08/13/2024

When looking at how I wanted to approach the testing for these three features I wanted to make sure that I used the requirements given to me and tested that those were implemented properly. With all three features I feel like I made sure to implement and test for all the requirements given to me and used my testing to check for that. I made sure that I passed my testing with each feature and all of the tests I used passed fine.

When writing my Junit testing I wanted to use those test to make sure that the code I created was following the requirements that were given. I also used the testing to find any errors in the code and try to fix those errors. To make sure that my code was efficient I made sure it was not filled with any unneeded code and that the code worked quickly. Using the Junit testing I could see how fast the code was able to process. I also made sure that there was not any unneeded test that could slow things down.

One of the software testing techniques I used at first was manual testing to see if the code I was writing was running without any errors. Then I used automated testing with Junit to do Function testing and Acceptance testing to make sure that the code is meeting the requirements given. Another type of testing that I did not use is integration testing which would test how the system that I created would integrate into a larger system such as a website or app. This type of testing would be used a lot in large systems like the ones I tested for setting up other systems like a banking system or software built for businesses. Another type is system testing which test the system as a whole and different aspects of it that can be showed to shareholders and higher ups. This would be used near the middle to end part of the development lifecycle because it can be used to help decide on release date projections and that legal and regulatory standards have been met.

The mindset that I adopted was someone that was on a team working with others so I wanted to make sure that everything works as it should as if I was on a team. When employing caution I wanted to make sure that I didn’t want to only trust the Junit test results and made sure to double check my the code while also using the test to help. It is important to appreciate the complexity of the code I was testing because it helps make test to check for everything and makes you think about how different functions can connect and how to test those connections. With each of the systems for this project I had to understand how the functions in these systems connected inside the system to be able to create appropriate test.

I tried to eliminate bias in reviewing my code I tried to look at it as if I was an outsider. This made it easier to find flaws in my code that I might not have noticed or cared about otherwise. I can imagine that bias would be a concern for the same issues that I listed. Someone who is testing their own code could ignore and not notice issues in their code of they are minor. Some people also might not want to view their code as being flawed so they are biased in thinking that it is fine as is already.

It is important to stay disciplined to quality when working on projects because it is easy to slip up or get sloppy when working especially for long hours writing hundred of lines of code. Staying disciplined and not cutting corners also makes it so that a system as a whole can work better and less chance for bugs to occur in the system because of shortcuts and sloppy coding. I will try to avoid tech debt by making sure that systems are as efficient as they can be and that systems are up to date when they need to be updated.