Project 2

Jon Curts

Southern New Hampshire University

Project 2

I have been asked to summarize and reflect on the various approaches I used based on the requirements I was given while developing the mobile application for the customer This should include on how I applied the appropriate testing strategies to the application.

# Summary

* Describe your unit testing approach for each of the three features.

Each feature was unique but related in its functionality to what the other features did. While this did a similar approach in the testing of each feature, each feature had to have its unique functions accounted for. I think for me checking the date was the most complicated part of the testing. Devising how to make sure that the date was not before today but not accounting for the immediate time had sum tests failing for some time. I eventually through testing and trial and error found a way to get the tests to work.

* Describe your experience writing the JUnit tests.

While I had never written JUnit tests before I did find it a difficult one to start doing. What I did find complicated was writing tests that covered enough functions to get the test to cover 80% of the code. Sometimes I was shocked what wasn’t being tested and I would have to go back and either rewrite the code or rewrite the test. That was one of the hardest parts, getting the coverage up to 80%. It also provided invaluable insight into the code I had written like the parts that I wasn’t even using. I had to stop and rethink certain parts and how they had been written. So, I would describe JUnit testing as a positive experience in learning and becoming a better coder.

## Reflection

1. Testing Techniques
   1. What were the software testing techniques that you employed in this project? Describe their characteristics using specific details.

I want to say I start out using static testing because it is by far the simplest to execute, because it does not actually need to run the code to be done. This just means I review the code I have written looking for any errors or mistakes I may have inadvertently put into the code. Next, I do a compliance test to make sure the code while meet the requirements of the task I have been given. Then I moved into unit testing, this was accomplished with JUnit.

JUnit testing requires building tests that while run the code itself this fall under dynamic testing. For unit testing to be effective it has to check each part of the code and get as much coverage as you can get. Any part of the code that is not covered under the unit test does not get the benefit of being known functional. That is way unit testing takes some time to get the most coverage that it can so it can guarantee functionality.

* 1. What are the other software testing techniques that you did not use for this project? Describe their characteristics using specific details.

I did not do any integration testing for the application that would have required me to have access to the application itself and not just the features I had written. Integration is a important next step in putting together a functioning application to make sure all its parts work together. I also did not go into any security testing which is important for any program that stores customer information. That would check for known issues with the current software or connections I would be using. Lastly there was not any Functionality testing that would check to see if the application works as, it is intended too. That would require me to have access to thew GUI of the program to ensure all the buttons functioned and that all the process performed correctly.

* 1. For each of the techniques you discussed, explain the practical uses and implications for different software development projects and situations.

Building just the features like we are doing we really can’t do functional or acceptance testing, both those require a near complete software. They would like to be done after integration testing later in the SDLC. Integration testing could be accomplished at this stage if we had more interconnecting feature to test with. Lastly Security testing is also something we could be testing right now but was not a requirement for this application.

1. Mindset
   1. Assess the mindset that you adopted working on this project. In acting as a software tester, to what extent did you employ caution? Why was it important to appreciate the complexity and interrelationships of the code you were testing? Provide specific examples to illustrate your claims.

When I test my code, I first look to find common mistakes or code that is poorly written. This means I can ensure the code is readable and structurally sound before I start any dynamic testing. The more time I take to make sure the code is correct in the first place the less time I must spend later. When I did start unit testing, I aimed to get the most coverage I could. This would ensure I was test most of the code I was using and that I would not run into any issues any further down the road. For example, when I put the task service together, I went over it to check for errors manual first and only after did I write the JUnit tests. I add more test or altered test to ensure I was getting at least 80% coverage.

* 1. Assess the ways you tried to limit bias in your review of the code. On the software developer side, can you imagine that bias would be a concern if you were responsible for testing your own code? Provide specific examples to illustrate your claims.

I tried to eliminate by using JUnit to ensure the code I write is functional, I also try to stay aware that I can make mistakes in the code and lastly, I try to review best practices when writing code. If I had another person to review my code, I would but unfortunately, I don’t know anyone else who can code to check my work. I do think bias would be a concern when testing your own code in a professional setting, as people we tend to think we know the best way of doing things even though new techniques are constantly being developed. An example of the would be use a older code that is no longer used, while it may still work it there likely was a reason for it to no longer to be used.

* 1. Finally, evaluate the importance of being disciplined in your commitment to quality as a software engineering professional. Why is it important not to cut corners when it comes to writing or testing code? How do you plan to avoid technical debt as a practitioner in the field? Provide specific examples to illustrate your claims.

I think that since computer systems are a rapidly evolving technology I have to stay up to date on the current best practice and changes in the languages I am employing to write my code. If I did not, I would be like a doctor still trying to balance humors in the human body doing more harm than good. Just because I learn something does not mean that I would every really stop learning about it. I think that is the way of technology and knowledge in general it is a ever evolving landscape that you have to keep up with. I do plan to continue my education in the future I would like to continue study in AI and machine learning because I believe that it will be the next biggest wave. It may seem big now but as it develops, I think it will automate more and more of our lives.

I think technical debt is not completely avoidable, but most of it can be mitigated by maintain beat practices and keeping up to date with current technology. To do so I will endeavor to continue my personal studies and get outside help to check my code. I do not believe it is ever a good thing to rest on your laurels as the world passes you by.