7-2 Project Two Submission

Joel Garcia

SNHU CS 320

1. **Summary**
   1. Describe your unit testing approach for each of the three features.
      1. To what extent was your approach **aligned to the software requirements**? Support your claims with specific evidence.

My approach was completely aligned with the requirements in that I built it around the rubric, reading the rubric requirements as I went. For all components I built sections of code around each of the needed classes, methods, and tests. As I went through each milestone I also made an effort to refactor the code to more thoroughly match requirements.

* + 1. Defend the overall quality of your JUnit tests. In other words, how do you know your JUnit tests were **effective** based on the coverage percentage?

I feel fairly confident in the coverage and effectiveness of my tests, but more so in task service and appointment service than contact service. Tests were based on all rubric requirements for variable length, null validation, object creation, update, and deletion, and in the case of appointment service, “before” date.

* 1. Describe your experience writing the JUnit tests.
     1. How did you ensure that your code was **technically sound**? Cite specific lines of code from your tests to illustrate.

I had problems with contact service that prevented me from running it, which I think was due to me switching back and forth between SNHU apporto and my own native pc Eclipse IDE. When trying to test I encountered a runtime error for class versions, and because I didn’t have any time left for the assignment I opted to just turn it in without resolving the issue. For task service, I stuck to writing on my own native Eclipse IDE and didn’t encounter that problem. TaskTest.java completed 13 of 13 tests without issues, however TaskServiceTest.java only completed 4 of 5 without failure. In line 14 of TaskServiceTest there was a failure where taskId was returned as false instead of true. I believe there is a logic error somewhere but I am having difficulty figuring out if it is in the Test file or the TaskService file. In appointment service, while largely the same, the addition of the “before” date requirement called for some adjustment. This was solved by adding logic to throw an exception for any date entered that was before the current date, as well as tests to ensure that this feature worked.

* + 1. How did you ensure that your code was **efficient**? Cite specific lines of code from your tests to illustrate.

Certain things like unique ID creation and search were created as their own methods that could be called by individual tests, rather than have to declare IDs or complete searches in each test, in order to keep the code more efficient.

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

I used static testing and unit testing techniques. Static testing was employed before writing the code while looking over the rubric requirements, as well as being employed after writing the code but before running it by manually examining for errors. Unit testing was employed to check all of the software methods and constructors to ensure that it met all input validation requirements.

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

System testing and acceptance testing were not used as they were outside the scope of the assignments. System testing tests whether the software works as a whole and acceptance testing tests whether it is ready to be accepted by the customer, and neither was part of the project.

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

The practical uses for static testing is to ensure that you go into the code writing step with a plan in place as well as check that the code is clean and efficient before you even test, while the practical uses for unit testing is to make sure software components work properly. The practical uses for things like integration, system, and acceptance testing are to ensure that you have a working finished product that is ready to be handed over to the customer.

* 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.

I employed caution heavily, and this manifested in changing as little code as possible. I order to ensure the complexity of the interrelationships of the code was undisturbed, I made a point to copy each previous weeks code and modify the parameters rather than write entirely new code each week, in order to keep things consistent.

* + 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.

The best way I found to limit bias was to follow the rubric closely. If I only focus on what the requirements ask then I am less likely to insert my own tweaks to the code. Basically, I wrote the simplest code possible that achieves the requirements and nothing more.

* + 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.

There were a few instances where I had one or two tests that were running but failing. While I could have just left them alone and still met the needed coverage percentage, I chose to continue to refactor until I corrected the simple logic errors that were causing the problem.