Dylan Coulter April 19, 2023

ID#1896062

CS320

# CS 320 Assignment: Project Two

**Summary**

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

**To what extent was your approach *aligned to the software requirements*? Support your claims with specific evidence.**

The testing phase is integral to the software development life cycle. When we test our code, it allows for us to ensure that the product we develop is the highest quality and meets the requirements of the client and end users. We were tasked with creating multiple classes to create a program. The first step was identifying what the requirements for each class. The classes that were created were ContactService, TaskService, and AppointmentService. For the assignment we pulled the requirements from reading and understanding what the rubric was asking of us. We created test for each class and tested the code to ensure it met all of the identified requirements. We also focused on creating test with over 90 percent code coverage.

**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?**

The test that we utilized on our code classes was JUnit test. The focus was to ensure within each class we created code that would match each requirement. Next the focus turned to creating JUnit tests to test the code for each requirement to ensure there would be no errors or the code would be altered until it passed. For example, when creating AppointmentService there were two classes involve Appointment which the identified requirements were appoint ID which could not be more than ten characters, appointment date which could not be in the past, and appointment description which could not be more than 50 characters. The other class was AppointmentService which had two requirements. The first requirement is appointment service should be able to add appointments with a unique ID. The second requirement is appointment service should be able to delete appointments per appointment ID. As we can see in the example below first the code coverage is 93.4 percent which is above the goal of 90 percent. Second, we can see we created test to test each one of the identified requirements within our code.

**Text

Description automatically generated with low confidenceText

Description automatically generated**

**Describe your experience writing the JUnit tests.**

**How did you ensure that your code was *technically sound*? Cite specific lines of code from your tests to illustrate.**

When creating our code, we took a few steps to ensure it was technically sound. The first step that was taken was to make sure we had the proper names for our classes and methods. Next the focus was on creating clear and concise comments throughout the code. This is always important because just because someone starts or develops some or all of the code does not mean that someone else will not be finishing or updating it in the future. Having clear code and comments will help anyone working with the code in the future best understand the thought process of the original developer. Examples of this can be seen in the code below.

Graphical user interface, text

Description automatically generated

A picture containing timeline

Description automatically generated

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

Creating efficient code is one of the most important aspects when working with the development of it. When we are developing code, we must ensure we do not create extra or unnecessary line of code. When the program for the project was developed and tested the focus was on writing lines of code that were needed and ensuring we had no extra lines that were unnecessary. The focus was also on creating code that was not overly complex as we never know if another programmer would be working with it in the future. Looking at the example below we can see the code is not overly complex, we test what is needed, and there are no unnecessary lines of code.

Text, application

Description automatically generated

**Reflection**

**Testing Techniques**

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

To complete this project there were three milestones that needed to be completed and updated before submitted. The three milestones were to develop a contact service, task service, and appointment service application. During the development of these application, we used unit test to test the functionality and that all requirements were met. These unit test were in the form of JUnit test and the test allowed us to identify and issue created during the development of the code. When doing this the focus was to test small portions of the code for errors. The portions of code tested would range anywhere from single lines of code to classes. One important part of the testing was to ensure we had adequate code coverage which would allow us to see how much of the code was tested.

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

Throughout this project there were testing techniques that were not utilized. As previously mentioned, unit tests were the main form of testing used when completing this project. A software testing technique that was not utilized during this project was black box testing. The focus of black box testing is for the program to be tested by someone who has no known knowledge of the inner workings of the program. With the tester having no knowledge of the programs design it is being tested almost as if they are the end user. Black box testing was not utilized in the creation of the project because the code was developed and tested by the same individual.

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

Throughout the development of this project, we learned the importance of testing our code. The focus was centered on using JUnit test to ensure we had adequate coverage and that the identified requirements were met. Unit tests on our code allowed us to break our code down and test small portions of it to ensure it was of quality. Doing this allowed us to see that the code worked as required. We also focused on code coverage which showed how much of the code our test was covering. Without this tool we would not know how much of the code was being test and if it was functioning as it should risking the overall quality of the finished product. Unit tests are a great tool for many different software projects to use. When developing the project, we did not use black box testing, but this is a for of testing the is used for other projects. Black box testing has some pros and cons. As mentioned earlier the tester when black box testing has no known knowledge of the inner workings of the program. This can be a pro because the tester and the programmer work independently limiting any bias. A con of black box testing is that it cannot be utilized to test complex segments of code. Also, the test on the software can be redundant if they have already been complete by the programmer.