CS320 Summary and Reflections Report

Alexander Garcia

Southern New Hampshire University

Table of Contents

[Table of Contents 2](#_Toc117338013)

[1. Summary 3](#_Toc117338014)

[1a. Describe your unit testing approach for each of the three features. 3](#_Toc117338015)

[1b. Describe your experience writing the JUnit tests. 4](#_Toc117338016)

[2. Reflection 4](#_Toc117338017)

[2a. Testing Techniques 4](#_Toc117338018)

[2b. Mindset 5](#_Toc117338019)

[References 6](#_Toc117338020)

CS320 Summary and Reflections Report

# 1. Summary

## 1a. Describe your unit testing approach for each of the three features.

### Contact Service.

#### To what extent was your approach aligned with the software requirements?

I made sure to have the contact service follow every requirement set forth I the requirements document. Making sure it could create the contact in memory, delete it, search the contacts. I also made sure to include the ability to update, which was more or less finding the existing contact, deleting it, and adding in the new version, as the requirements asked for.

#### Defend the quality of your JUnit tests.

I created comprehensive Junit testing to address each of the requirements individually and ensure proper error catching. For example, if a string needed to be a certain size, I made sure that the string would throw an error when it was too long, or that it threw an error if it didn’t exist when it needed to. My code reached more than suitable coverage.

### Task Service.

#### To what extent was your approach aligned with the software requirements?

The task service was a similar design as the contact service. I made sure to have the task service follow every requirement set forth I the requirements document. Making sure it could create the task in memory and delete it. I also made sure to include the ability to update, which was finding the existing task, deleting it, and adding in the new version, as the requirements asked for.

#### Defend the quality of your JUnit tests.

Since the layout for the service was similar, the layout for the testing was equally similar to the contact service testing. I created comprehensive Junit testing to address each of the requirements individually and ensure proper error catching. For example, if a string needed to exist, I made sure that the string would throw an error when it was null. My code reached more than suitable coverage.

### Appointment Service.

#### To what extent was your approach aligned with the software requirements?

Again, the service was generally the same as the other two with some slight tweaks, so I just made sure to address each of the requirements specifically. Ensuring there was a way to create appointments and delete them.

#### Defend the quality of your JUnit tests.

The testing was equally similar to the other services. There were individual tests to address each requirement of each field or service. The code reached more than adequate coverage.

## 1b. Describe your experience writing the JUnit tests.

### How did you ensure that your code was technically sound?

I used both static and dynamic testing, by ensure that there were no errors or typos in the code beforehand and that the logic would play out correctly. Then by having tests try different inputs to the services for dynamic testing. I ensured that the response of the program was lining up with what was expected.

### How did you ensure that your test code was efficient?

For efficiency I relied on the expertise of the guides for this project. By following the style and coding practices of the tutorials, I was able to write the code which performed efficiently.

# 2. Reflection

## 2a. Testing Techniques

### What were the software testing techniques that you employed in this project?

In this project I used both static and dynamic testing techniques. Specifically for dynamic testing I used Junit tests.

### What are the other software testing techniques that you did not use for this project?

I did not use other testing techniques that may involve having a user test the software by using it. I was also thinking about using some sort of automated testing, but due to time I did not end up using that technique either.

### For each technique you discussed, explain their practical uses and implications for different software projects and situations.

Automated testing could have cut down on some of the time it took to manually code each of the j unit tests written, it would definitely be worth implementing if the requirements were more robust as the time to write tests for each requirement would shoot up. Having a user or QA person test the code would likely come a little later for this software, as there was not yet an interface.

## 2b. Mindset

### Assess the mindset you adopted working on this project.

When working on this project I tried to focus first on writing all the code for each service, and addressing the requirements. Then after I was finished with the coding and ensuring requirements are met, I would begin to create the tests.

### Assess the ways you tried to limit bias in your review of the code.

I definitely tried my best to write the code and tests slightly separate from each other, while still integrating any fixes needed to both. I also tried to take breaks in between writing the code and writing the tests, so that I could have a fresh look.

### Finally, evaluate the importance of being disciplined in your commitment to quality as a software engineering professional.

It is important to have strong testing for the code so that you can find any errors in the code as early as possible. In general it is good practice to have a solid way of testing your code so that you have a way to ensure that you code is doing what you want and expect it to do. This is especially true when creating software with specific requirements.