Olivia Edey

Module 7-2 Project Two

CS 320

Project One Reflection Report

Software requirements are important for software development. It gives developers a guideline of what the client wants their product to do. It also tests the functionality of the product to the clients’ desire. One of the requirements of the code for the Contact Class was to make sure none of the first or last names, phone numbers or addresses are longer than 10 characters. In order to comply with the requirements, I wrote the code so that it would not take any input longer than 10 characters, for example;

“if (contactID != null && contactID.length() <= 10) {“

This line of code is to confirm that the contactID is not null or longer than 10 characters. This was also added in for each variable in the Contact Class. I also had to make sure that the Task and Appointment classes also followed the same rules but with different lengths. Another requirement for this assignment was to make sure the task and appointment service is able to add, delete or update the information with the unique ID, for example;

public void addAppointment(Appointment appointment) {

appointments.put(appointment.getAppointmentID(), appointment);

}

To ensure my code is efficient and sound, I ran it to make sure all outcomes came out without any errors. It isn’t easy to write code, you have to look out for spelling and make sure the symbols are in the correct spot. In the code I needed to make sure my variables were spelt correctly and had semicolons on the end of each line, for example,

“private String contactID;

private String firstName;”

The variable contactID and firstName had to match throughout the rest of the code so that it can understand that I am referring to that specific variable. Spelling it differently would have created more errors and the code would not get what variable I am referring to.

I employed Unit Testing as the primary software testing technique. Specifically, I used a form of white-box testing, which is structure-based. “The purpose of unit testing is to examine the individual components or pieces of methods/classes to verify functionality, ensuring the behavior is as expected” (Jakubiak, 2022). This technique aims to guarantee that the code performs as intended and meets all requirements. In the AppointmentTest class, I implemented a unit test method called test AppointmentCreation. This test method verifies the correct creation of an appointment by setting its ID, date, and description.

Another software testing technique that I used is integration testing. Integration testing focuses on assessing how different components of the software work together as a whole. These tests help identify potential issues early in the development process, making debugging more manageable. Integration testing is especially valuable for ensuring the smooth flow of data and catching integration-related bugs between units. By employing a combination of these testing techniques based on the project's scope, complexity, and requirements, can ensure the delivery of a high-quality, reliable software products to end-users. Carefully choosing the appropriate testing approach enhances the overall robustness and performance of the software.

One testing technique that I did not use is performance testing. Performance testing ensures, “software quality by evaluating application output, processing speed, data transfer velocity, and network bandwidth usage” (Achhaa, 2022). Performance testing evaluates the software’s responsiveness under different conditions such as entering a large number of appointments, contacts or tasks and sees how the system handles the work. This test can evaluate how the software performs with heavy usage.

When working on this project, I tried to work diligently so that minimal mistakes were made. It was interesting to work using JUnit test because it is something I have used before. As well as writing the code for the separate classes that worked together to make a more complex code. While writing this code I was going back and forth through each one to make sure that the code is readable and imports from one another**.**

It is hard to be unbiased when writing your own code. A developer can see some flaws in a request that the client wants and may try to add that into the code or overlook potential issues that may not seem like a big concern. I wanted to make sure the software requirements align with the code as much as possible. For example, I made sure to implement the limit on characters that can be entered for the ID:

“if (contactID != null && contactID.length() <= 10) {“

This is a requirement the client specifically listed that they wanted. Not something that was added by the developer. This helps ensure that your review is based on predetermined requirements rather than subjective opinions. As a software engineer it is important to be disciplined to the quality of the code because the client is expecting a high-quality product that they can say is reliable and user-friendly. Cutting corners may decrease customer satisfaction when the product isn’t working as requested and that the product isn’t in the most optimal form it can be in. Avoiding shortcuts and delivering a high-quality product will not only keep the customers happy but the customers, clients happy as well. Along with keeping the customers content, their satisfaction will also boost your reputation and help you grow.

# Works Cited

Achhaa, N. (2022, 7 8). *What is Performance Testing?* Retrieved 8 12, 2023, from ACCELQ: https://www.accelq.com/blog/performance-testing/#What

Jakubiak, N. (2022, 12 6). *JUnit Tutorial: Setting Up, Writing, and Running Java Unit Tests*. Retrieved 8 12, 2023, from PARASOFT: https://www.parasoft.com/blog/junit-tutorial-setting-up-writing-and-running-java-unit-tests/#What\_Is\_JUnit