Professor Robert Tuft

Software Test Automation and QA

CS 320 – R4837

Tasheka Dillon

**Module 7-2 Project Two**

**Summary**

**Unit Testing Approach:**

For each of the assignments required for the project, there had to be a Contact class and Contact service class, a Task class and a Task Service class, and an Appointment class and Appointment Service class. These six features were tested by way of Junit testing. When it came to my approach for aligning my code with software requirements, I made sure that the code in the Test code met the requirements of the main code so that the tests would pass accurately and sufficiently.

The quality of my tests was sound, but some errors prevented my code from reaching 80% coverage. The effectiveness of my tests hinged upon how well the tests were executed in the Eclipse program.

I ensured that my code was technically sound by ensuring that there were no visible errors. I utilized Ecipse’s error log to help me catch errors and fixed them with Eclipse’s hint function, which allowed me to see what needed to be revised.

**Coding Exampes:**

For the code example, I will demonstrate how the test code met the instructions given in the rubric.

***Contact class code requirements:***

The contact object shall have a required firstName String field that cannot be longer than 10 characters. The firstName field shall not be null.

***Contact Code example:***

if (id == null || id.length() > 10) {

throw new IllegalArgumentException("Invalid ID");

***Contact Test Code example:***

void testGetFirstName() {

assertEquals("Barbara", contact.getFirstName());

To ensure that my code is efficient, I made sure that the test code ran adequately and it matched the coding requirements. I also

For the code example, I will demonstrate how the test code met the instructions given in the rubric.

***Contact class code requirement:***

The contact object shall have a required lastName String field that cannot be longer than 10 characters. The lastName field shall not be null.

***Contact Code Example:***

if(lastName == null | lastName.length() > 10) {

throw new IllegalArgumentException(“Invalid lastName”);

}

***Contact Test Class:***

void testGetFirstName() {

assertEquals(“Willowbird”, contact.getLastName());

**Reflection**

**Testing Techniques:**

The software technique I used in this project was a unit testing approach, where I implemented the Junit test on my written code. Within the project, I wrote code that would fit the description of the assignment instructions and test code that would pass the requirements. When it came to producing the code for the project with contact, task, and appointment, I made sure to use the Junit feature of Eclipse to test the code.

The other software testing technique I used for this project is “manual testing, which utilizes manual inspection of the code and testing of the software.” (CFG, Geeks) In this project, I also used functional testing in the project to ensure tat all requirements of the code were met.

The practical uses of the techniques I mentioned are typically done in companies that use manual testing. The Quality Assurance tester will utilize manual testing to test software and functional testing on large projects to ensure that the company is meeting its customers’ guidelines.

**Mindset:**

Working on this project as a software tester, I adopted the mindset of ensuring that all of my code had no visible errors or bugs. I also ensured that my code had no syntax errors and that I followed the assignment guidelines. It's important to appreciate the complexity and interrelationships of the code that is being tested because they can affect the coding process.

I tried to limit my bias in my review of the code by focusing on the task at hand and making sure my code was functioning to the liking of the customer at hand.

As a software engineering professional, it is crucial to avoid cutting corners and maintain the integrity of the code written in order to remain a trustworthy resource for the company I work for and its customers. There are many dangers of cutting corners and many examples of things going terribly wrong if corners are cut when it comes to writing and testing code. Companies have lost millions of dollars and some have even lost lives at the cost of inaccurate code. One example that I can think of that I recently did research on is the Denver International Airport Baggage Handling System defect, which was a case of system failure due to underestimation of the project complexity, which caused confusion and error in the building of the system, resulting in the city of Denver to lose out on 1.1M per day and a delay in airport opening for 16 months.

**References**

GfG, Geeks. “Software Testing Techniques.” *GeeksforGeeks*, GeeksforGeeks, 6 Dec. 2023, www.geeksforgeeks.org/software-testing-techniques/.

Calleam. "Why Do Projects Fail?" *Why Do Projects Fail*, calleam.com/WTPF/?page\_id=2086. Accessed 18 Apr. 2008.