Brian Blackman

7-2 Project Two Submission

08/24/2024

Southern New Hampshire University

Over the three featured programs, the approach that was used for the unit testing were all White-box test techniques. ”White-box test techniques are used to explore system or component structures at several levels.” (Morgan, 2019) Specifically, the Statement Testing and Coverage technique was used to ensure the programs aligned to the software requirements. Statement testing execute statements of code to ensure the code function as it suppose. This is done by forcing certain parameters into the code and running it. (Morgan, 2019) For example, the Task application required each task to have a unique id. The generateTaskById method (Figure 1) created a unique id. In the Junit test (Figure 2) a new task was created and a new id was printed. The coverage of the code is also tested to ensure at least 80% or above of the code was tested and covered.

At the beginning of this course, I had no experience writing JUnit test for testing Java code. I can say that I will definitely use this application to test code in the future. To Ensure my code was technically sound I written in checks within the code to check if certain instances occurred. For example, In the Task application, it was a requirement that the task name was not null and could not be longer than 20 characters. To make sure this did not occur an “If” statement was inserted and a JUnit test was created to test this (Figure 3).

As I mentioned in 5-2 Journal: Software Testing Techniques, several types of software testing techniques were used. The testing techniques were functional testing, unit testing, and usability testing. Functional testing tests the functional requirements of the software. Unit testing tested the units of code that the customer was asking for. Usability testing is making sure that the software was easy to use. Some forms of software testing I did not use were performance testing and system testing. There was no requirement for how fast the software needed to be or how stable the software needed to be although this should be tested before the software is released to ensure there are no failures. There are also no requirements for which system the software needs to be run on so without knowing the system I can’t test for it.

Over all the applications, I took on the mindset of a software tester and had to take caution to ensure that every aspect of the code was tested properly. It is important to appreciate the complexity and interrelationship of the code I tested because, it ensures that the quality of the application was to the customers expectation. The bias the can occur when testing your own code is that you assume your test will work as it supposed to so the test you provide is not as comprehensive as someone else testing your code. As history show it is important to stay disciplined in writing or testing code because we don’t want something bad happen like the Mars Climate Orbiter.

**Figure 1.**

A computer screen shot of text

Description automatically generated

**Figure 2.**

A blue and green text on a black background

Description automatically generated

**Figure 3.**

A screen shot of a computer code

Description automatically generated

**Figure 4.**

**A screenshot of a computer screen

Description automatically generatedA computer screen shot of a computer code

Description automatically generated**

References:

Morgan, Peter, et al. Software Testing : An ISTQB-BCS Certified Tester Foundation Guide - 4th Edition, edited by Brian Hambling, BCS Learning & Development Limited, 2019. ProQuest Ebook Central, http://ebookcentral.proquest.com/lib/snhu-ebooks/detail.action?docID=5837074.

Created from snhu-ebooks on 2024-08-25 00:53:47.