**Project Two: Summary & Reflection**

William Harris

Southern New Hampshire University

CS-320: Software Test Automation & QA

Dr. Angel Cross

June 20, 2022

1. Reflection
   1. Unit Testing Approach
      1. Alignment to Requirements
         1. Each class required a corresponding test class to test its functionality. Junit tests allow me the confidence to know I have met the client’s requirements. Most of the requirements were covered, which ensures me that at least those requirements are met within the program.
      2. Quality
         1. My testing was adequate. While I did not achieve 100% coverage of all classes due to time constraints, a total project test coverage demonstrated that 95%+ was achieved(Harris, 2022).
   2. My Experience

The JUnit tests I utilized in my program allowed me to check if every possible path of my program was taken. This is not only beneficial in ensuring all of the client’s requirements are met, but also serves to expose any areas of weakness in my program, such as security vulnerabilities and unhandled exceptions.

* + 1. Structure & Security
       1. Writing JUnit tests that test the structure of your code is useful. However, it is just as beneficial to test the functionality of your code to see how the various classes within the structure will interact with each other.
    2. Ensuring Efficiency
       1. I implemented a singleton pattern in the classes where it might be unnecessary or even disastrous to run multiple instances. Ensuring code coverage was above acceptable levels was also vital in determining the efficiency of my program.

1. Summary
   1. Techniques
      1. Software Techniques Used
         1. I used private constructors for security alongside singleton patterns so that only one instance of appropriate classes are instantiated. This will remove the possibility that unnecessary memory will be used by the program as well as ensure data is safely protected(Harris, 2022).
      2. Other Techniques
         1. I would have loved to implement some try-catch statements into my program. I have reached proficiency in that area and they would have been beneficial in not only preventing code-run error crashes but also displaying where and why errors are happening.
      3. Practical Uses & Implications
         1. I have used the singleton pattern in many of my personal projects. For the banking app, it is definitely important to avoid multiple instances of a banking session. You should not be able to withdraw from an account in two different sessions, as this may cause balance errors and inconsistencies.
   2. Mindset
      1. Caution
         1. I became more and more cautious about the way that I wrote code the more classes I added to the program. Due to their associated natures, it is best to write classes that will be accessed only in an authorized and efficient manner. Only classes that should have access should be able to access corresponding classes and the data within them.
      2. Bias
         1. A great way to eliminate bias is to constantly try to think objectively. You should have a firm grasp on the client;s requirements and only give the client what they requested and any additional required functionality necessary to fulfill the clients’ request. Frequently referencing the software specification document during the course of development will be essential in ensuring a deliverable the client will love. Effort should also be made to get the highest amount of coverage on your testing as possible.
      3. Discipline
         1. Cutting corners could include assuming your program will work in a certain area without testing it. For many of my functions use String inputs. These inputs should be checked to validate that they are indeed Strings and not longer than the allowed length as these could introduce security vulnerabilities into the code. Ensuring that every new piece of added code has a corresponding test case with full coverage will also be vital.

**References**

Harris, W (2022) Project One Submission

https://learn.snhu.edu/d2l/common/viewFile.d2lfile/Database/Njg5MDM3NDg/Module%206%20Submission.zip?ou=1069793