3-2 Milestone 2 Narrative

Michael P Clisbee

CS 499

Southern New Hampshire University

**Briefly describe the Artifact. What is it? When was it created?**

For Artifact One in Category One: Software Engineering/Design, I selected work done in CS320 – Software Testing, Automation and Quality Assurance, which focuses on locating and resolving software security vulnerabilities by creating secure code and testing procedures to locate issues before code implementation. My enhancement plan resulted in expanding the complexity of my Unit Test files created in Java and edited and implemented using Visual Studio, Terminal and XCode. In addition, enhancements were made to the associated ContactTest.java and ContactServiceTest.java files linked to their respective java files. This demonstrated skills learned in altering Java code to be more secure, as well as in the creation of efficient test files, created specifically for their respective java files for testing.

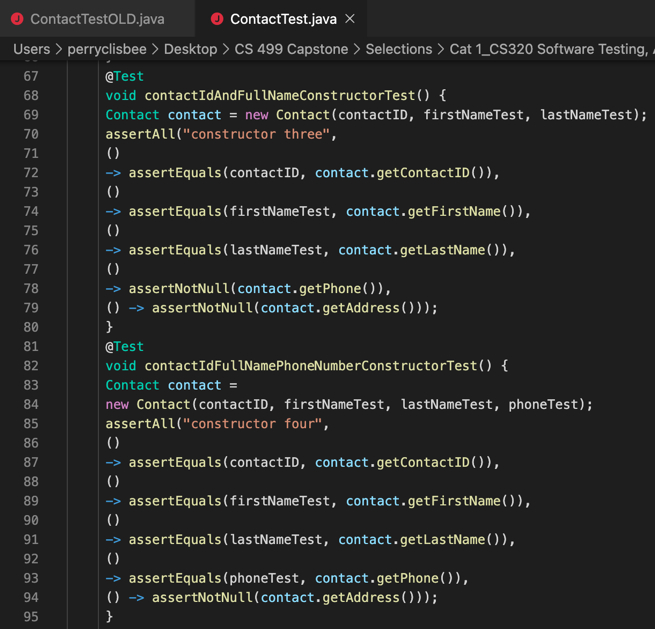
The artifact includes a ContactTest.java file that implements JUnit methods to test the attributes of the Contact class. I created tests that verified the ID is not null, updateable, and less than or equal to 10 characters. Tests were created to determine a potential client’s needs for other variables such as firstName, lastName, phoneNumber and address. These JUnit tests were adjusted to be used with any additional parameters needed to validate the Contact class data.

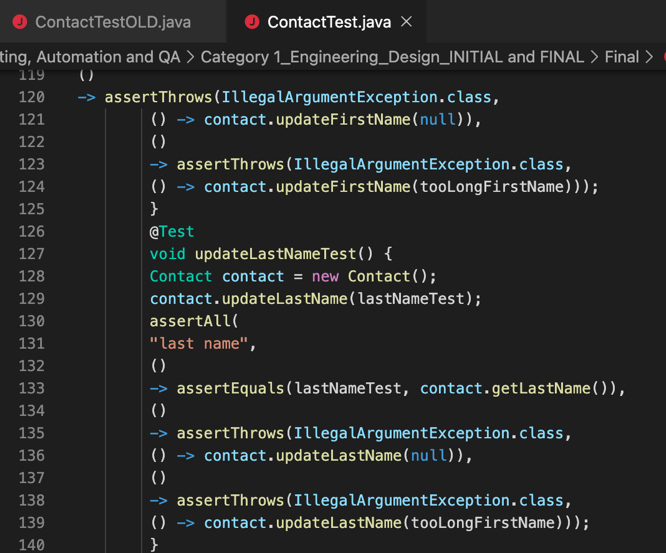
**Justify the inclusion of the Artifact in your ePorfolio. Why did you select this item? What specific components of the Artifact showcase your skills and abilities in software development? How was the artifact improved?**

The ContactTest.java file and its associated Contact.java file were included in the ePorfolio as the test file code displayed the test variations that could be employed to effectively test variable combinations entered by the user. The test file includes specific constructors that tested to ensure parameters for a new contact were asserted to have data, or a NotNull status. The first one tested that all necessary variables were not null. The next one tested to make sure the parameter entered matched the contactID, and that all other variables were NotNull. This continues with testing for assertEquals() for both contactID and firstNameTest. I added several other assert variations such as assert.Update and assertThrows() and assertEquals() methods to validate or notify of an illegal argument.

Seen below are these additions.

Text

Description automatically generated These display constructor tests adding additional assertEquals() parameters 

****

**Text

Description automatically generated**

**Text

Description automatically generated**

These three sections above of added code include assertThrows() methods pinpointing an error with an Illegal Argument notification under conditions

Documentation was added for each @test to ensure other developers will understand what was attempted here.

**Did you meet the course objectives you planned to meet with this enhancement in Module One?**

The enhancements met my goals for overall enhancement as it added JUnit tests that covered a wide range of possible outcomes, to ensure minimal error on the backend. Additional code documentation was added that added clear description for each code segment.

**Reflect on the process of enhancing and/or modifying the Artifact. What did you learn as you were creating it and improving it? What challenges did you face?**

When modifying the artifact, I did a lot of code cross referencing online to better understand the purpose of the different assert methods, and how to implement them. I used available software such as XCode to view and alter as necessary.