Testing Techniques and Evaluation for PizzaDronz

Range of Techniques:

All three layers of testing—unit, integration, and system—were completed. Test classes FlightPathTest and RestServiceTest are two instances of integration testing. The SystemTest class contains the system test. The remaining test classes are instances of unit testing. JUnit, a Java unit testing framework, was the primary instrument utilized for testing the application.

Systematic Functional Testing:

Specification-based testing was used to create the unit and integration tests. This approach enables us to confirm that the project's functional requirements for each component are met, integrate actual data from the REST server, and ensure that proper functionality is maintained.

Structural Testing:

This process was meant to be used in addition to functional testing. It made sure every element of the functionality under test was covered and that proper behavior was observed. This method has been quite helpful in identifying particular test scenarios with unique or distinctive behavior (which, of course, call for extra care). For instance, there were many odd test cases in the process of determining every reason why a card number would not be legitimate. Consequently, every scenario that could arise in a branch has been examined.

Boundary value analysis:

Boundaries of central regions and no-fly zones can be tested using this methodology. To ensure that the system makes the right choice given the parameters, we check the edge cases, taking into account scenarios in which the input is precisely on the edge of these zones. Making sure the system manages edge cases correctly and efficiently is made possible by the application of boundary value analysis.

For testing **measurable attributes**, I have done the following:

Performance:

-Timer was used to measure the system computation time and ensure that this falls within the required timeframe of 60 seconds.

-Stress Testing has been done to test if the system is capable of handling enormous amounts of data

Security: javax.net.ssl.HttpsURLConnection; has been used to make sure HTTPs connection has been made and for SSL certificate verification

Functionality: By running the test suite and recording the number of criteria that were satisfied, we were able to take into account every need in the requirements specification.

Evaluation criteria for the adequacy of the testing

1)Code Coverage:

This is a crucial indicator since it shows how much code the tests have actually processed. The key idea here is that you can't find a flawed assertion until you try it. It takes into account various paths and encompasses control and data flow. As a result, this criterion provides a reasonable assessment of the testing's sufficiency.

Class Coverage: is one evaluation criterion that can be used to guide the selection of testing methods. The test suite's percentage of the system's classes being represented by it is the criterion in question. This criterion supports systematic functional testing as a testing methodology since it allows us to perform unit and integration testing on all individual classes and system components.

Statement coverage: is another parameter that can be used to guide the selection of testing methods. This criterion is the percentage of program executable statements that the test suite is able to run. This criterion supports boundary value analysis testing methods since they allow us to evaluate the system more fully by taking into account several test cases.

Test coverage: is another factor that can be applied. This criterion is based on the test suite covering a significant amount of the system code, which includes executable statements and classes. This criterion supports any testing method employed because it takes into consideration the two previously mentioned characteristics.

Specification Coverage: A measure of the amount of tested system requirements is called specification coverage. We can ensure that every functional need for the system is tested and that no requirement is left unchecked by doing this. This has been used for technique of systematic functional testing since this technique checks to perform specification-based testing for the system’s functional requirements.

2)Test Passing: If the developed tests pass, the application is functioning properly.

3) Diversity of Testing: The program's proper behavior in accordance with the given specification can only be ensured by meticulously built tests that preferably include every potential kind of input.