

# **Deliverable 4, Chapter 4**

## **Automated Testing Framework: Adding New Logic for New Tests**

Some changes were made in how our testing framework works since Chapter 3. Now, instead of using the names/numbers of the tests to run separate files in the testCaseExecutables directory, we run all tests using information in the .txt files found in the testCases directory. These files contain all information needed to execute the tests without needing repeated code in the testCaseExecutables files. We use two different types of cases in our testing: tests written in Python that use Selenium to simulate interactions in a browser window to test the user interface, and tests on the JavaScript code that EnDe is largely written in.

In runAllTests.py, we check whether we need to be running Python code to test the user interface or JavaScript code to test methods directly. For the JavaScript tests we used QUnit, which is a JavaScript unit testing framework. We were able to get the results of whether a test passed or failed using QUnit, then used those values in our report.

Almost all of the functionality for our automated testing framework is in runAllTests.py. This includes searching through the testCases directory, pulling out relevant information from the files and invoking the appropriate methods, and writing and saving the report. This was done mainly to avoid repeated code in separate executable files for each test case.

## **The Test Case Text Files**

Our test case text files contain different information depending on whether the information will be used in a Python user interface test or a JavaScript test for functions. All test case files contain a test number, requirement being tested, component being tested, test input, and expected outcome (oracle). For the user interface tests, we also include a list number which allows us to locate which item in the list of types of encryption we need to click in the user interface and a link id that allows us to click the proper specific encryption/decryption method in a sub-menu under that type. For our JavaScript tests, we include the method being tested and how to invoke the method. We also include arity, so if a user enters too many arguments it will only accept the first ones up to the amount required.