**CS 3110 Final Project Test Plan**

In our project, a lot of our code is devoted to GUIs and calls to various APIs for getting ticker information (stock related). By nature, these are difficult to test. However, we were able to figure out clever ways to test those functions. The first goal was to achieve good Bisect coverage. While 100% coverage is extremely difficult to obtain with the nature of our application, we at least wanted good coverage. At the time, we defined it to be 75%, and in fact, we exceeded that in our overall testing. To break up the API calls with tests, we test each call and check to make sure it has the correct keys inside (to check an error vs. containing data). Note that the tests can error if the API keys run out of calls. For example, Alpha Vantage only allows 25 requests per day. Then, we stored an example response from each API call and then check the validity of the data retrieval within that JSON.

Then, we also had helper functions that were rather easy to test. These were utilized in our analysis and API compilation units. That includes testing the functions “days\_before,” “leap\_year,” “take,” “sub,” “calculate\_average,” and “range\_x.” Finally, we also took some example data from highly credible websites for simple explanations of indicators, imported them via text files, and then tested our functions to match those values.

We believe that our test suite demonstrates the correctness of our system because we have provided extensive black-box testing for the majority of helper functions, as well as arbitrarily test API calls. In addition to this, we have also tested example data (from API calls) that checks proper access to various numbers (entries). In our testing, all of our test cases utilize OUnit in making a test suite for the project. Note that for the helper functions, we have manually defined test cases that test various cases (including corner cases).