Buy Bitcoin
November 30, 2016
Pair A: Reader: Christophe Rimann, Recorder: Nigel Schuster
Pari B: Reader: Jordan Vega, Recorder: Julian Serra

Team Assignment 6: Second Iteration Development and Code Inspection (and Demo)

Static Analysis: We have incorporated static analysis as part of our continuous integration. The report can be found at:
https://github.com/Neitsch/ASE4156/tree/master/documentation/assignment6

Pair A component to be Analyzed (written by B): Stocks Component (/stocks/historical.py, /stocks/models.py, /stocks/stock_helper.py)

Pair A's notes and finds: The development of this component was quite progressive and reviewed continuously due to its importance to the platform. In fact, every pull request has required review. As such, very few issues were found. The code looks good, static analysis has no problems with it, and it works as expected. It also makes sense to us reading it. We noticed some potential issues with filling in past data progressively, and the need to automate this 'filling' to ensure that the datasets are complete, accurate, and descriptive (the method called fill_quote_history in historical.py). This issue is in the process of fixing. Overall, our reaction to the code is that it is in great condition, barring the fix of the aforementioned filling issue.

Pair B component to be Analyzed (written by A): Authentication Component (/authentication/plaid_wrapper.py, /authentication/models.py, /authentication/graphql.py)

Pair B's notes and finds: Because every pull request performed by our team requires review, we had already interacted and analyzed most modules as a whole. Similarly, since our continuous integration runs static analysis checks, as well as a plethora of tests, every commit made to date is well revised. However, we decided to jump in a bit deeper to see if we could find weaknesses or issues in this authentication component. While going through the component, we found that the code was clear and concise, what was written made sense to us and the files seemed complete (and obviously working). The interaction between components and classes is efficient and logical, and we think that this component should run into no problems in the future, and if it does, they will be easily identifiable due to the rational, logical, and organized nature of the code.

Notes from Demo with our IA team Mentor: We demoed for our IA team mentor, Yu, on thursday. He mentioned during the demo that he was happy with the test coverage we had for our project, and happy with the continuous integration tools we had set up. During the demo, we ran into no problems with Yu. We did realize that our stock history had few flaws and maybe our stocks database needed to be filled with more stocks. Overall, the demo ran smoothly and Yu was happy with our progress (and testing).