Apple 2.0

Scrum Report

Team Members: Courtney Colbert, Mike Gutkind, Henning Kjoeita, Lindsay Scott
Scrum Master: Henning Kjoeita
Product Owner: Lindsay Scott
Estimate of total person-hours spent on all aspects of job until now: 16

Summary: The first half of the sprint consisted of API research, the wireframing of the software's user interface, the research of Java's Swing, and the creation of GitHub accounts. After careful consideration, the team decided to use AlphaVantage as the API for the software. Drawings were developed as wireframes and were used as the basis for the design of the user interface. During the second half of the sprint, the team was able to begin the software's user interface. Furthermore, the team was unable to connect to the AlphaVantage API and made the decision to switch to Quandl, to which they successfully made a connection.

User Stories:

- Easily Accessible UI - The design of the user interface was developed from a wireframe drawing and was implemented with Java's Swing library. The interface required an efficient way to traverse through the application. The user interface's implementation of tabs allows users to easily look through the different pages of the application. **Implementers:** Courtney, Henning

- Scrollable Homepage - A scrollable homepage was implemented to allow the user to access several sources of information at once. **Implementers:** Courtney, Henning

- Real Time Information Access - Real time information access was implemented with the connection of the backend of the application to the Quandl API. The feature was tested using Quandl's test URL, which retrieves a JSON string of real time data. From the backend, the data is parsed using libraries like GSON or JSON.Simple to manipulate the data to a more readable fashion. **Implementers:** Lindsay, Mike

- Change graph data according to specific time range - Quandl's easily manipulatable URL format makes it easy to access a time range. **Implementers:** Lindsay, Mike

Integration Testing: The UI was developed under two operating systems (Mac OS and Windows). Upon inspection, Courtney and Henning noticed that the UI behaved noticeably differently based on the operating system. The issue was specifically the behavior of JTabbedPane when placed left on Mac OS. We developed and implemented a work-around. Running the UI and checking that everything behaves and looks as intended.
The code is clean, we found and fixed one issue with the UI. The quality is good.

Scrum I retrospective: Each team member's perspective of their individual contribution and accomplishment

Courtney's Contribution: My main focus for this sprint was starting the UI. To begin with, I drew up sketches of a possible UI and went to my team for feedback. Then I began the code for the UI. Originally, I was going to have buttons on the frame that takes the user to the different pages, but after a little bit of research, I found that a JTabbedPane actually looks more professional and is easier to implement. I added a titled border to each page and some drop down boxes where I knew were necessary at the time. We then added some small things like setting a minimum size and a default close operation for the frame. We also created a scroll bar that appears when needed. We organized our code better by creating a function that sets the title orientation, font, and size that is used for each pane, since the same lines of code were present in each pane. This made our code shorter and less cluttered.

Michael's Contribution (Team Member): I focused on several things during this first sprint. At first i set up a GIT repository to store the code after we finish editing and integrating it. After that i began looking for valid api's to use for the data source of our entire project. After i found several valid and capable api's i decided to begin writing a service layer to call the URL, retrieve the json, bring it back to the service layer and parse it for specific information. In order to do this i chose to use Maven dependencies so that i can access parsers such as Gson and Spring MVC. Once i wrote the code to parse the api's i tried three different API's to realistically check which data would come through the cleanest and most standardized for our application. After doing that i narrowed down our groups api choices to a single api. I then was able to successfully retrieve real time price data from any stock i specified. I unit tested this with stocks such as AAPL, MSFT, FB and TSLA to make sure the parser would be able to read it. Towards the end of the sprint i integrated my code with the UI's code. At the end of the first sprint i started making a list of api calls that would be needed in the future developments of our application.

Henning Kjoeita's Contribution: Beside being the Scrum Master for the first sprint and organizing meetings with the team as well as other non-coding work I also worked on the UI. First I made a quick sketch that visually showed how I would imagine a UI to look like, Courtney also made a sketch and we decided to follow her's as especially the way the different 'tabs' of the program was organized looked nice. I then looked over her code, where I made some changes. There was a couple of lines that could be removed, some other lines that could be put into a seperate method, and I made the code be cleaner overall. To do this I again worked with Courtney, so that we both agreed on how to change to code. Then we researched how to add a scroll bar, and I eventually ended up used JScrollPanel to get that done, which I also tested was actually working (by adding a lot of text and resizing the frame).

Lindsay Scott's contribution: I began the first sprint by researching API's that would be easy to use with Java. The group decided on AlphaVantage, but Mike and I were unable to make a connection from the backend of our software. Mike was able to find the Quandl API, which he was able to connect to successfully. From there, I created a Dataset object that employs JSON.Simple to parse the JSON that is returned in the form of a string from the API. Using the test URL provided by Quandl, I was able to add Stock details to the front-end's summary page to test the Dataset object's ability to parse data.

Product Owner's statement of quality of product: In its current state, the software displays a simple and easily accessible interface. The layout employs tabs to display content, which emphasizes an easily accessible UI and is one of the most important user stories for the software. The software also connects to the Quandl API, which satisfies the on of the user story requirements to retrieve real time information.

Scrum Master's statement with estimate of total effort expended (person hours) during this scrum and discussing what items may need to be done next;
The team have spent approximately 16 hours working on this project. In the next sprint we plan create graphs that visualises trends in a stock (8 story points) and complete the Summary page (8 story points). In this sprint we did 22 story points so it is possible we will be able to do more than 16 story points the next sprint too. Current progress is good and the quality of the product is good.

Set up for sprint 2: Add new user stories in preparation for scrum II

- As a user, I would like to have charts of my stocks so I can visualize trends and see how my portfolio is divided.
    - Story Points: 8

- As a user, I would like a summary on the homepage so I can easily check and analyze my favorite stocks.
    - Story Points: 8
    -

Signatures of SM, PO, Team members.