Apple 2.0
Stock Portfolio Project

Team Members: Courtney Colbert, Mike Gutkind, Henning Kjoeita, Lindsay Scott

Sprint I report:

Scrum Master: Henning Kjoeita

Product Owner: Lindsay Scott

Team members: Courtney Colbert, Mike Gutkind

Estimate of total person-hours spent on all aspects of job until now: 16

Comments:

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:

<u>Easily Accessible UI</u> - 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. It is natural that the interface will require work as the other parts of the project are developed. The interface will be manually tested by clicking around and see if the behavior is as intended. In the early phases we found some difference in the behavior in MacOs and Windows, and how they treated the left side of JTabbedPane, the solution was to simply have the tabs on top rather than the left side. As of now there is no issues, but as other features are implemented we will have to keep testing.
Implementers: Courtney, Henning

- Story Points: 16

Scrollable Homepage - A scrollable homepage was implemented to allow the user to access several sources of information at once. Testing that the scroll bars worked was done by creating Jlabels with so much text that a scrollbar was necessary to see all the text, the frame was also resized and the scrollbar behavior noted as the size changed. Initially after testing it was not behaving as expected, but after some changes the scroll function is confirmed to work after testing. **Implementers:** Courtney, Henning

- Story Points: 2

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

- Story Points: 4

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

- Story Points: 2

Comments:

Total story points: 22

Integration Testing

Connected the api to the UI and observed that selecting stocks in the graphical user interface correctly lead to api calls.

The team is happy with the progress and quality of the product and code so far.

Scrum I retrospective:

Courtney's Contribution (Team Member)

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 (Scrum Master)
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 (Product owner)
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
- 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

Sprint II report:

Scrum Master: Mike Gutkind
Product Owner: Courtney Colbert
Team members: Lindsay Scott and Henning Kjoeita
Estimate of total person-hours spent on all aspects of job until now: 17

Summary:

During sprint two we focused on a lot of the functionality of our program. In doing this we found that we should organize our "service layer" and "Back end" so that the last two sprints we can focus on making the GUI much more practical and useful for the user. This sprint was a lot of trial and error. At the end of sprint one, we set out to create an accurate depiction of any selected stock symbol and parse that specific stocks information in a graph form. We also aimed to connect our stock portfolio program to a backend service (database) so that we can store a specific user's information. On top of all this we needed to add an interface that allowed for a user to sign into the application with their specifically designated credentials. As of 4/10/18 we were successfully able to meet all these goals and then add even more functionality than we expected. We were able to accomplish several goals related to functionality during this sprint. We still have more functionality we would like to add along with many other UI goals we'd like to meet in the upcoming sprints. Thus we will be able to design the best user interface and efficiently compliment the functionality of the program.

User Stories:

<u>Working Charts -</u> As a user, I would like to have charts of my stocks so I can visualize trends and see how my portfolio is divided. **Implementers:** Mike and Henning. This user story got very difficult to implement because it was incredibly complex. We worked on the graph format during every sprint at some point in order to produce what we really liked. After putting a lot of time and debugging into it, we successfully produced a resizable functional graph. The graph required special attention as we saw in testing: little changes can break an entire graph and ruin a user's experience.

- Story Points: 8

<u>Password Protection-</u> As a user, I would like to have the option of setting a password so my portfolio information is secure. **Implementers:** Courtney and Lindsay. This was difficult because we needed to connect a user interface component with a MongoDB database. After a full sprint we were able to create a working back-end sign in method that also gives the option to create a new user. We unit tested this several times in order to work out all bugs in connecting to the database and hide the password text. After

completing multiple drafts and insuring the sign in service would never crash, we successfully completed our sign in service.

- Story Points: 8

Specific Stock News- As a user, I would like to be able to select an individual stock such that it opens a browser displaying any news in regards to said stock. **Implementers:** Mike and Henning. Specific Stock News was based on an API call that retrieves major news company headlines regarding a specific stock. Considering every stock had a different number of headlines about them we needed to make the feature dynamic. We also had to make the UI components hyperlinks so that a user could read the full article if they wanted to. The outcome of this page was very good in utilizing the articles found by the API. It was difficult in testing the hyperlinks because it required the desktops default browser to run outside the application. Making this dynamic was difficult and had few bugs in testing but we were eventually able to fully integrate it.

- Story Points: 4

Search Functionality- As a user, I would like a search bar with stock suggestions so I can follow market trends and potentially find stocks to add to my portfolio. **Implementers:** Mike, Henning, Courtney and Lindsay. This feature focused on the user experience. We turned to an outside source and decided to go with SwingX, a java library that uses suggestive search in a JComboBox to help the user complete their typing. It was difficult finding this library but incredibly easy to implement using Maven. Now both our symbol and company name JComboBox's help suggest what a user should search.

- Story Points: 4

Backend Database - As a user, I would like to sign into my profile on any computer so that incase my computer dies, my personal account data will be saved in the back end. **Implementers:** Lindsay. This took longer than we expected. Lindsay ran into a few bugs while implementing this. After we got it to work with simple string's we began trying to find ways to store lists, such as a user's favorite stocks. After several days of frustration and programming we were unable to save lists in time but we were able to create a working sign in service that saves user's to the database. This service worked very well but we are still looking into saving a user's designated favorite stocks.

- Story Points: 8

Integration Testing:

We had to integrate the back end database with the sign in and in doing this we created several bugs. In order kill these bugs we attempted creating a user, making our program query the database for that specific user and insure that the entered credentials are correct. This took several iterations before we could claim the sign in service was finished. Another thing we did was insure the Stock News Panel coincided with the Stock Search Functionality. The news panel depends on the Stock search to change the headlines and headline sources, we found many bugs in this as we integrated the components. Debugging this feature took about three days to full integrate which we then found it to be quick and efficient in looking for news articles. We would also test certain things like if we could sign in search for no stocks or if a stock has no information in the API how the error is handled. These tests help squash bugs and create a much smoother flowing program for our Sprint 2 final product.

Our product at the end of sprint two was fully functioning with the specific search panel, the News panel and the sign in service. This kept us on time as we entered sprint three. The proper integration test allowed for our product to remain working, intuitive and high quality at all times.

Michael's Contribution (Scrum Master):

Apple 2.0 accomplished a lot this sprint. As scrum master i approached our development in a slightly different way. At the very beginning of the sprint i divided each tab into different classes. Mike did this so that we could each work on a specific page without stepping on each other's toes. This allowed me to focus most of my energy on the Stock Search functionality and back-end API calls. first, Mike fully integrated Maven into our project and imported specific libraries regarding charts, parsing and json retrieval so that we could get all the information from the API cleanly. The next thing Mike did was implement the fully functional search function which allowed for any stock to be searched for and then displayed in chart form with fully accurate data. Mike then integrated this with the entire project upon finishing testing.

Courtney's Contribution (Product Owner):

My main focus for this sprint was creating a menu screen that requires a password to access the stock ticker. I made the password hard coded into the program. The password can either be submitted by clicking the login button or pressing the enter key. If the text in the JPasswordField matches the designated password, it closes the menu frame and opens the stock ticker, and also clears out the text in the JPasswordField. I also added a logout button to the summary page that closes the stock ticker and reopens the menu screen. I rearranged the summary page a bit and created an array list that will store the user's favorite stocks. I added some placeholder stocks to the array list to make sure my loop displays all of the stocks from the array list on the frame properly.

Henning Kjoeita's Contribution:

This sprint I worked on implementing a way to graph the data from stocks, I used JFreeChart as the base and worked from there to implement it into our project. I created a separate class that can be used to get a JPanel containing the graph. (The getChart() function places the chart on a ChartPanel, and then the ChartPanel on a Jpanel which it returns, that is so it's easy to use to chart for my team members). The chart class handles everything to create the chart, it only needs to be given some series (list of data points) that it can graph. I also enabled the graph to show more than 1 series (or more than 1 stock in this case) on the same chart. I then implemented the class into the dataPanel tab to demo it's functions, and test that it works as intended. For this purpose I have created two buttons, one that displays a random stock, and one that adds a random stock as a series. To to that I also made the dataPanel implement the actionlistener interface.

Lindsay Scott's contribution:

For the second sprint, I deployed a NoSQL database using MongoDB Atlas to house user credentials and portfolio data, though I focused solely on creating and accessing user credentials for this sprint. MongoDB stores collections of data as documents in BSON format, which is an extension of the JSON format. The application's database consists of a "users" node, or collection, where all users are stored. Each user has a unique id, a field for storing a username, and a field for storing a password.

Though it's considered bad practice to store passwords as plain text, for the sake of time and simplicity, user passwords will be stored in the database as a string. This can be easily improved by hashing the password before storing it in the database. I created a MongoConnect class to connect to and access the database. The class currently has two main functions, user creation and user authentication. The user creation method takes a username and password string from the user and scans the "users" collection to see if the username passed already exists in the database. If it does exist, the user will be asked to pick another username for their account. If the username doesn't exist in the database, the username and password are pushed to the database to create a new user. Like the user creation method, the user authentication method gets username and password strings from the user trying to sign in. The "users" collection is scanned for the username entered. If the username exists, the document is converted to a JSON object and parsed to get the stored password for that username. If the password retrieved from the database matches the password entered by the user, then the user has successfully signed in.

Product Owner's statement of quality of product:

In its current state, the UI is simple and easy to use. There is a search bar that gives suggestions which allows for easy access to data on the stock of my choice, and displays a chart so I can see how that stock has been doing. Displaying real time data is one of the most important aspects, so I am satisfied with the way it currently displays and works. A password was also implemented on a menu screen which will keep my portfolio secure.

Scrum Master's Statement:

Overall, the team collectively put in approximately 17 hours of work this sprint. A lot of these hours were dedicated to each person's designated task and then about an hour designated to integrated each specific feature. We were able to add much more functionality and features to our program without the risk of interfering with each other's code. While we have a lot more functionality than our last sprint, we must spend time on finishing the functionality aspect and cleaning up the user interface. Our user interface has not been our first priority and that is why we must cut off functionality in the next sprint so that our program is not only functional but also intuitive and easy to use. Another feature we plan on adding in the upcoming sprint is finishing the back-end schema so that user's can make new accounts and can save their favorite data accordingly.

Set up for sprint 3, New User Stories for Sprint Three:

Intuitive UI- As a user, I would like to have a much more accessible user interface so my experience can be a lot more enjoyable than staring at a disorganized grey screen.

- Story Points: 16

Functional Summary Page- As a user, I would like a summary on the homepage so I can easily check and analyze my favorite stocks.

- Story Points: 8

Quick Access to Stock Gains and Losses- As a user, I would like a page displaying gains and losses so I can see the state of my portfolio by having my stock gains and losses in one place.

- Story Points: 8

Color Coded Gains/Losses- As a user, I would like to color code stocks according to gains and losses so it's easier to see whether stocks grew or fell.

  - Story Points: 1

Graph Variable Time Ranges- As a user, I would like to set a time range for graphs so I can identify patterns in designated periods of time.

  - Story Points: 2

Sprint III Report:

Scrum Master: Lindsay Scott

Product Owner: Mike Gutkind

Team members: Courtney Colbert, Henning Kjoeita

Estimate of total person-hours spent on all aspects of job until now: 16

Summary: Having one less sprint than we originally thought we would, the team thought it was important to work on the user interface, but at the same time, did not want to compromise the functionality of the program. First, a frontend was added to the "login" and "sign up" functions created in the previous sprint. We knew having a functional Summary page was incredibly important, as this is the first screen the user sees after they sign in. Users' favorite stocks are now displayed on the Summary page, along with the most currently active stocks. Both sets of information display the gains or losses for that stock and are color coded to reflect that. Users can view and change the order of their favorite stocks on the Your Stocks panel. The News panel provides users with the latest headlines for whichever stock is selected in the Stock Search panel. In the Stock Search panel, users may now set a time range for stock data. Lastly, the Help panel encourages users to send any comments or concerns to Apple 2.0, and provides a button that creates a new email to our customer service for convenience. In total, these improvements and implementations have contributed to our most important goal, which is an intuitive UI. Every addition to this sprint contributed to the organization of the UI and overall improvement of user experience. The user stories we implemented in the third sprint are

Stories incorporated in Scrum III

- Intuitive UI- As a user, I would like to have a much more accessible user interface so my experience can be a lot more enjoyable than staring at a disorganized grey screen. **Implementers:** All members. Since most of the other user stories for this sprint had UI elements, this user story entailed each member of the team to complete their designated user stories and organized their UI elements in a neat and professional manner.
  - Story Points: 16

- Functional Summary Page- As a user, I would like a summary on the homepage so I can easily check and analyze my favorite stocks. **Implementers:** Mike and Lindsay. For this user story to be successfully implemented, the program will load the user's favorite stocks when they launch the program. Lindsay was able to implement a method to save a user's favorite stocks locally, which are retrieved as a list on the summary page. We tested this story by logging in to our own accounts and saving stocks to our favorites, which we were able to do without bugs.
    - Story Points: 8
- Set Time Range- As a user, I would like to set a time range for graphs so I can identify patterns in designated periods of time. **Implementers:** Henning. When the user selects a stock to view, the graph populates the selected stock's price information for the previous month. They are also able to view the stock data for the past six months and year. The user is also able to add more than one stock to compare them. After implementation, Henning found a significant bug in the graph. While the graph successfully displayed data for more than one stock at one month, the graph would not display data for more than one stock at the six month and one year time ranges.
    - Story Points: 2
- Gains and Losses Displayed- As a user, I would like a page displaying gains and losses so I can see the state of my portfolio by having my stock gains and losses in one place. **Implementers:** Mike and Lindsay. The user is able to see the gains and losses for their favorite stocks on the summary page. After Mike and Lindsay successfully implemented the functional summary page user story, Mike tied in the API to retrieve the gains and losses for each stock in the favorites list.
    - Story Points: 8
- News Page- As a user, I would like to be able to select an individual stock such that it opens a browser displaying any news in regards to said stock. **Implementers:** Mike. When the user selects a stock from the search page, news headlines for that stock are dynamically retrieved and displayed on the news panel. These headlines are hyperlinks which will open the specified article with the user's default browser when clicked. To test this, we selected stocks from the search panel to make sure headlines were being populated in the news panel, and clicked the headlines to make sure the hyperlink was functional and opened the appropriate article.
    - Story Points: 4
- Color Coded Stocks- As a user, I would like to color code stocks according to gains and losses so it's easier to see whether stocks grew or fell. **Implementers:** Mike. The stocks displayed on the summary pages are green or red to reflect gains or losses respectively. Mike was able to implement this story after the gains and losses user story was implemented. He was able to color-code the text, such that stocks with gains are green and stocks with losses are red.
    - Story Points: 1
- Password Protected (using backend)- As a user, I would like to have the option of setting a password so my portfolio information is secure. **Implementers:** Lindsay. Users are able to create an account with their email and a password. Once a user creates an account, they are able to login to the software and are taken to the summary page.
    - Story Points: 8
- Order of Stocks- As a user, I would like to change the order in which my stocks are displayed, so I can prioritize which stocks are more important than others. **Implementers:** Courtney. The user is able to change the order of the stocks on the Your Stocks panel. The Your Stocks panel

displays a list of all stocks for the user. The stocks are displayed alphabetically, but the user can change the order to reverse alphabetically.
    - Story Points: 2
- Help Panel- As a user, I would like a support button so I can get help if the application is not working properly. **Implementers:** Courtney. When the user needs help, they can open the help page. The page displays a link to our customer service email. When clicked, a Gmail email is opened for the user to send to our customer support.
    - Story Points: 1

Integration Testing:

Since everyone in the group worked on a different task, we got together several times during the sprint to ensure successful integration interface. Considering the amount of functionality we currently have, we will have more than enough time to improve and build a high quality user experience for our final product.

Michael's Contribution: My focus for this sprint was split between the landing page and the News tab along with cleaning and documenting my code. I initially started by dynamically retrieving news headlines on whatever call was currently being looked at in the search functionality. Thus the major headlines involving that specific stock are then displayed in the News Panel. I also made it so the headlines presented on the page are clickable and that when the specific headline is clicked the program will launch your computer's default browser and travel to that link for the full article. The second major feature i worked on was the landing page or "Summary Panel". This tab is designed two show two major aspects. The first being your specific user's list of favorite stocks and if they've gone up or down recently and if-so by how much. The Second section of the page displays today's most active stocks in the sense that it shows stocks with the most drastic changes over the course of the day. These stocks are analyzed by the API and reset every time the Summary tab is selected to insure the most accurate data is displayed. Finally i focused a lot on cleaning up smelly portions of our code. I went through and deleted all lines of code that were not being used and renamed variables to represent the data they hold better. I also attempted to eliminate code repetition and utilize more methods. This Sprint was very full and had many moving parts which is why i refused to finalize a feature until i tested it and fixed as many bugs as i could find.

Courtney's Contribution: My main focus for this sprint was having the favorite stocks displayed automatically refresh on the Summary and Your Stocks Panel once a stock is added in the Stock Search Panel. Also, I completed the combo boxes on the Your Stocks Panel so the displayed stocks can either be just favorites or all of your stocks, and they can be ordered alphabetically or reverse alphabetically. I finished the Help Panel by adding our logo along with a button that automatically opens the browser and creates a new email to our customer service.

H**enning Kjoeita's Contribution:** The third sprint I worked to ensure that everything was working as intended, and during testing I found a significant bug in the Graph. As initially it was created assuming we would only support one month, and then later 6 months and a year was added. However, while 6 months and a year works when the graph have a single stock it does not work when there is multiple stocks in the graph. I have modified the class containing the graph (CP) so that it tracks what stocks are currently in the graph. Then from the stock search panel I can use those stocks to redraw the graph correctly for 6 months and year. Unfortunately, implementing it has proven to be more time consuming than expected and I've not yet managed to deploy a fix. Though I have a good idea of how to best implement it.

Lindsay Scott's Contribution: For the third sprint, I added a login and sign up screens and connected them to my previously created login and sign up functions from Sprint 2. I also implemented a method to save stocks added to the user's favorites. I decided to implement this locally first as opposed to the database. When a user creates an account, a folder is created in the user's Documents. Once the user adds a stock to their favorites from the StockSearchPanel, the stock is added to a text file inside of the folder created when the user signed up. I was then able to add the favorites to the SummaryPanel and display daily gains or losses for the stocks by integrating with Mike's code for this sprint.

Product Owner's Statement of Quality of Product: In the current state, our product has come together very well. So far, we have been able to develop every tab available in our program to some extent. We have a login service that's connected to the backend which is exactly what we wanted. We have a Summary page that displays a users favorites and the most active stocks of the day. We have a Your Stocks panel which displays all the stocks you have specifically discussed as being your favorite. We also have a stock search functionality which allows users to search for different graphs and compare them to one another. This feature also allows the time range for the graph to be changed. We also have a News Panel which finds and displays the most recent headlines regarding the current events around a specific stock. Finally we have a Help Panel which is mainly focused on getting the user in contacts with a support system incase they do have any problems.

Scrum Master's Statement: For Sprint 3, the team put much of the focus into improving functionality and working on the user interface. The team spent roughly 16 hours on the project for this sprint. Each member had a designated task and were able to work independently, but were able to meet several times and integrate code with minimal complications. Both the interface and functionality of the program has improved considerably since the previous sprint. Functional improvements include implementation of the News panel, the Help panel, connecting the user authentication functions to the UI (login and sign up), and improvements to the graph. The Summary page has become more user-friendly by displaying the gains or losses for favorite stocks and the most active stocks.

Further Development:

- Updates
    - Bug fixes
    - Performance improvement
    - Maintenance

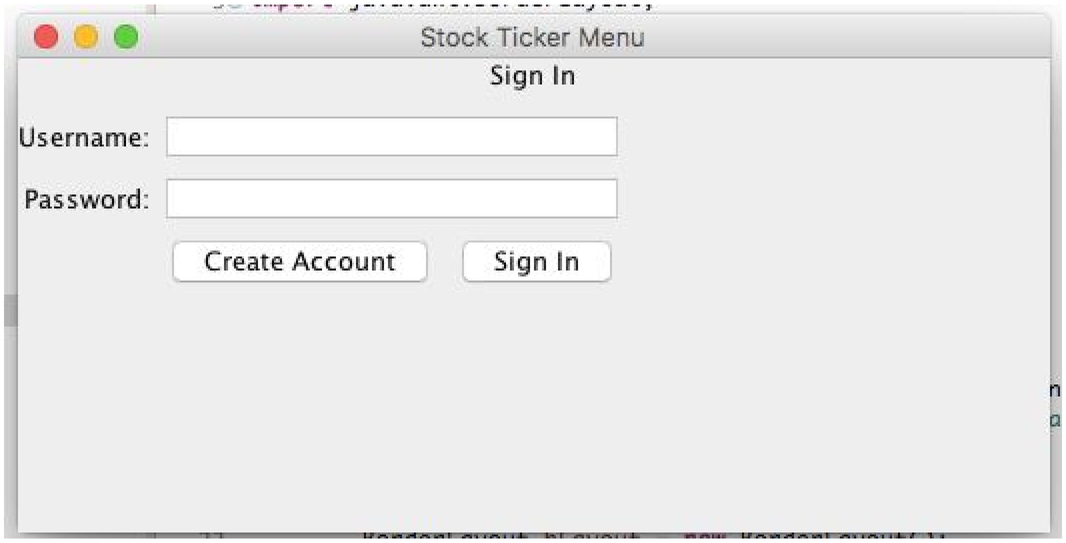
1. Final report

Goals:

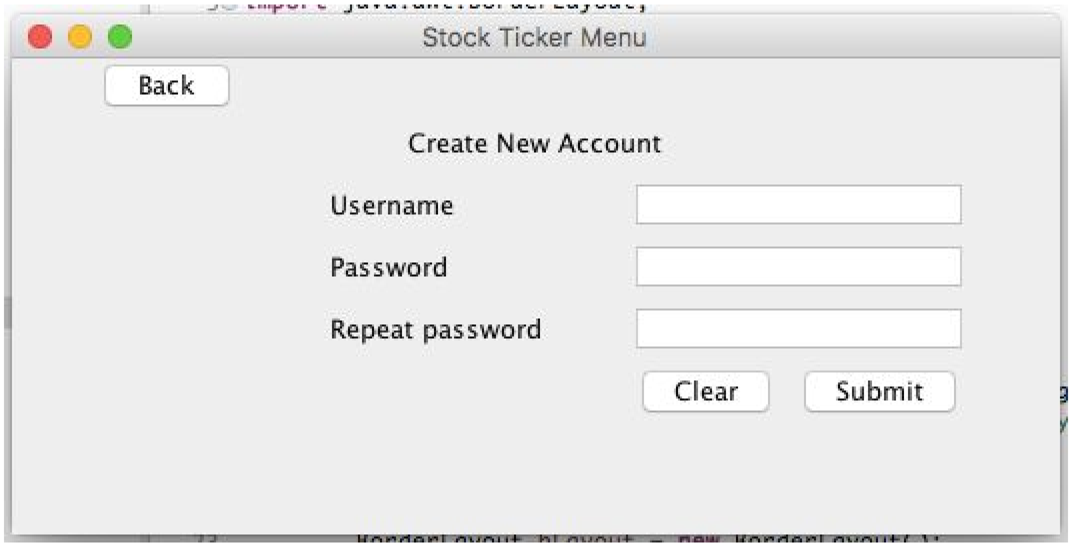
- Utilize Agile methodology to develop a working stock portfolio.
- Have an intuitive user interface.
- Have modular code to ease component testing and potential upgrades/maintenance.
- Have documentation that clearly explains workings of the code.
- Allow the application to be customizable according to every user.
- Complete the product on time (and within budget) without sacrificing features.

Description of the Demonstration along with screenshots:

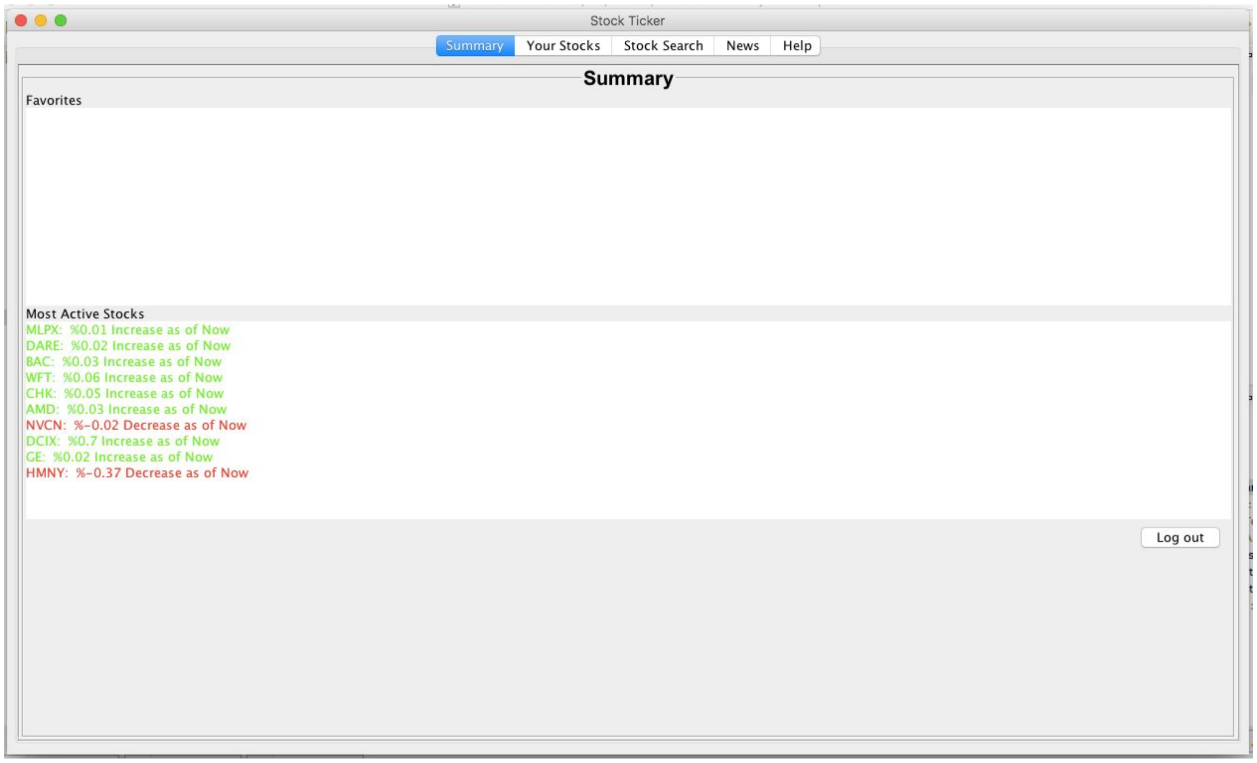
The first screen that appears when our program is run is the sign in screen.



If the user clicks on the Create Account button, the Create New Account frame appears which allows the user to create a username and password. Then it switches back to the sign in screen to allow the user to login.

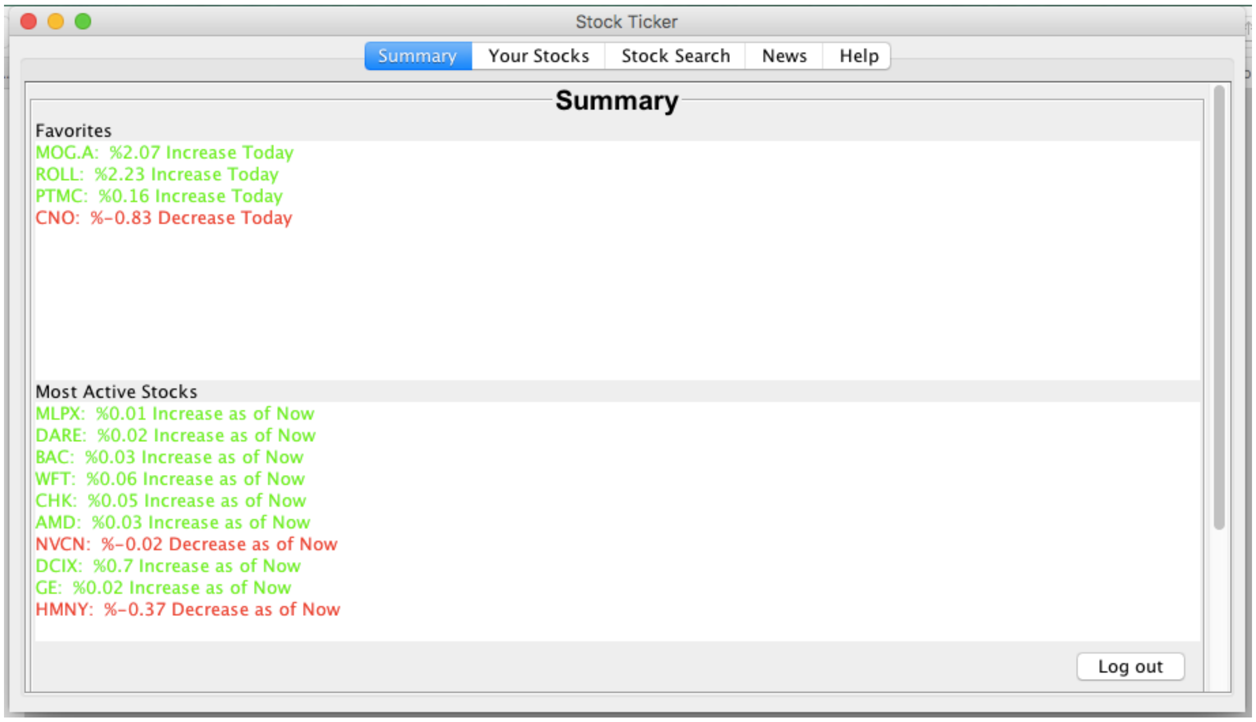


The Summary Tab is the first tab that is displayed once the user logs in, which displays the user's favorite stocks and the most active stocks for that day. The stocks are colored green if their price is increasing and red if their price is decreasing. Currently the Favorites section is empty because the user has not added any favorite stocks yet.



```
Most Active Stocks
MLPX:  %0.01 Increase as of Now
DARE:  %0.02 Increase as of Now
BAC:  %0.03 Increase as of Now
WFT:  %0.06 Increase as of Now
CHK:  %0.05 Increase as of Now
AMD:  %0.03 Increase as of Now
NVCN:  %-0.02 Decrease as of Now
DCIX:  %0.7 Increase as of Now
GE:  %0.02 Increase as of Now
HMNY:  %-0.37 Decrease as of Now
```

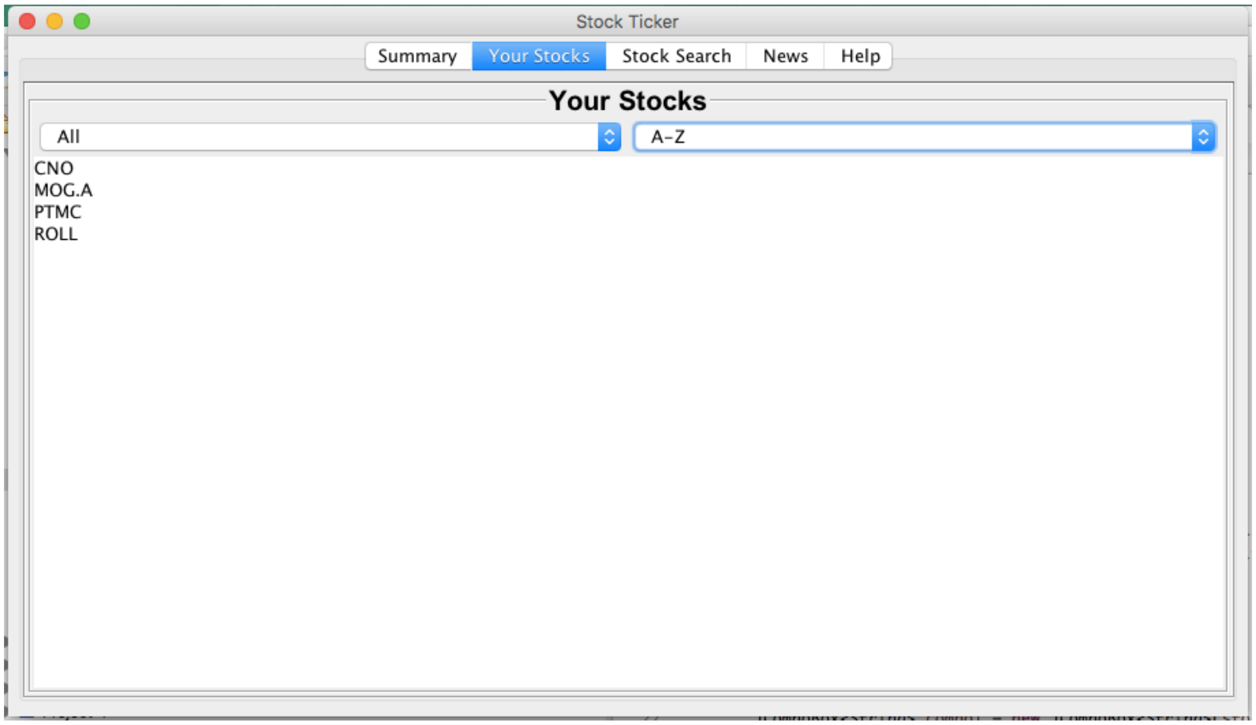
Once the user adds favorite stocks in the Stock Search tab, they appear under the favorites section automatically which can be seen below.



Favorites
MOG.A: %2.07 Increase Today
ROLL: %2.23 Increase Today
PTMC: %0.16 Increase Today
CNO: %-0.83 Decrease Today

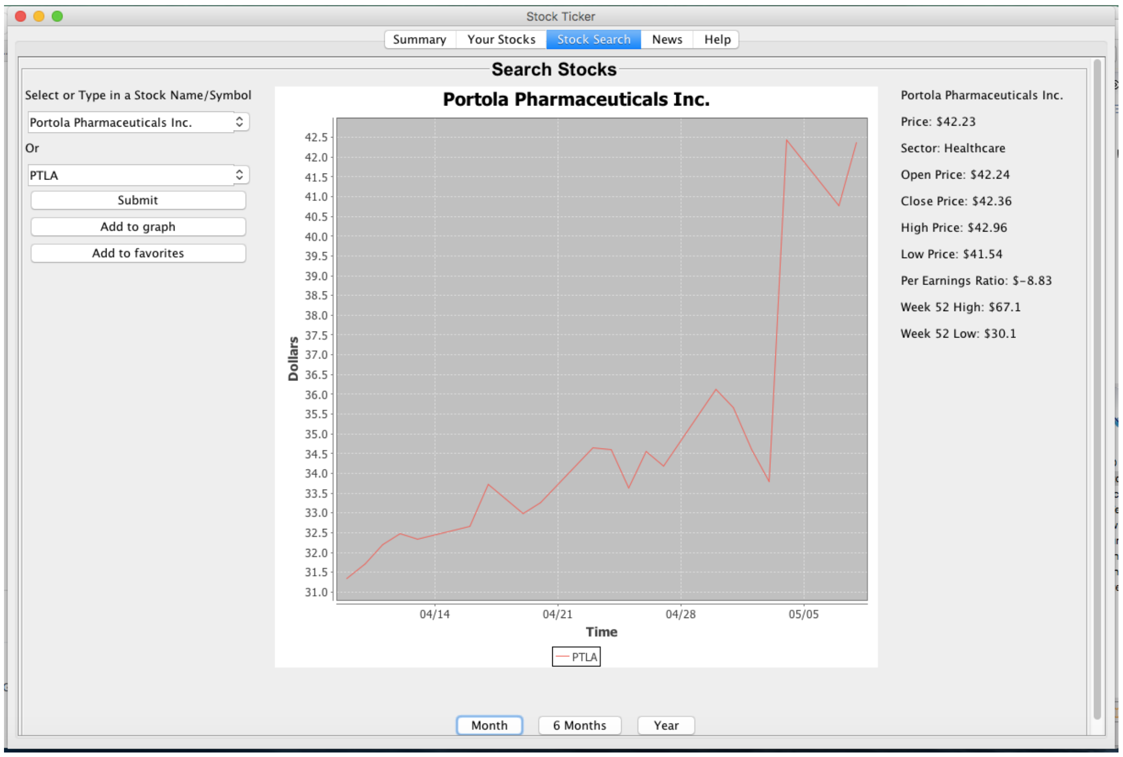
Most Active Stocks
MLPX: %0.01 Increase as of Now
DARE: %0.02 Increase as of Now
BAC: %0.03 Increase as of Now
WFT: %0.06 Increase as of Now
CHK: %0.05 Increase as of Now
AMD: %0.03 Increase as of Now
NVCN: %-0.02 Decrease as of Now
DCIX: %0.7 Increase as of Now
GE: %0.02 Increase as of Now
HMNY: %-0.37 Decrease as of Now

The second tab is the Your Stocks tab which displays the user's stocks. The combo boxes give the user the option to display all of their stocks or just their favorites. Also the user can display the stocks alphabetically or reverse alphabetically.

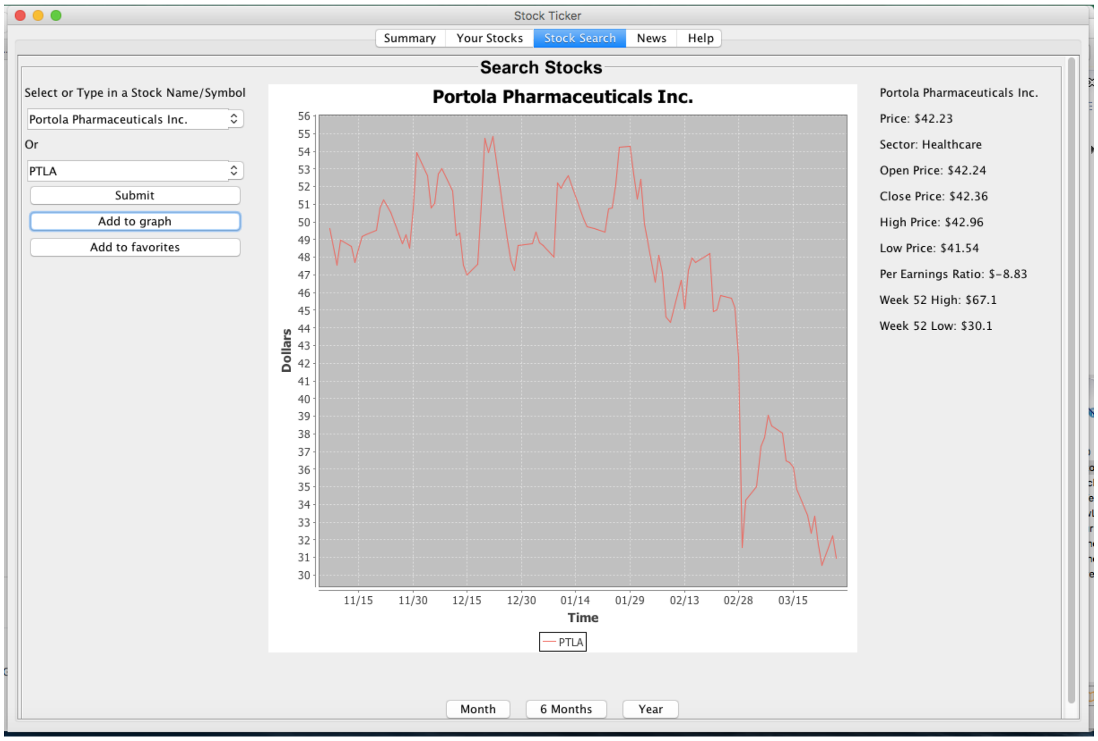


CNO
MOG.A
PTMC
ROLL

The next tab is the Search Stocks tab. This tab consists of a drop down box which has autocomplete so the user can search for a stock based on the full name or the symbol. The user then has the option to submit that stock which displays its data and a graph, the user can add it to the graph so multiple stocks' data can be displayed, or the user can add to it his favorites. For the graph, the user can select between the time ranges of month, 6 months, or year. If the API does not have sufficient data on a stock, an error message will be displayed. The following four screenshots are examples for the graph displaying for the month, 6 months, year, and multiple stocks at once.



Portola Pharmaceuticals Inc.
Price: $42.23
Sector: Healthcare
Open Price: $42.24
Close Price: $42.36
High Price: $42.96
Low Price: $41.54
Per Earnings Ratio: $-8.83
Week 52 High: $67.1
Week 52 Low: $30.1



Search Stocks

Select or Type in a Stock Name/Symbol

Portola Pharmaceuticals Inc.

Or

PTLA

Portola Pharmaceuticals Inc.

Price: $42.23

Sector: Healthcare

Open Price: $42.24

Close Price: $42.36

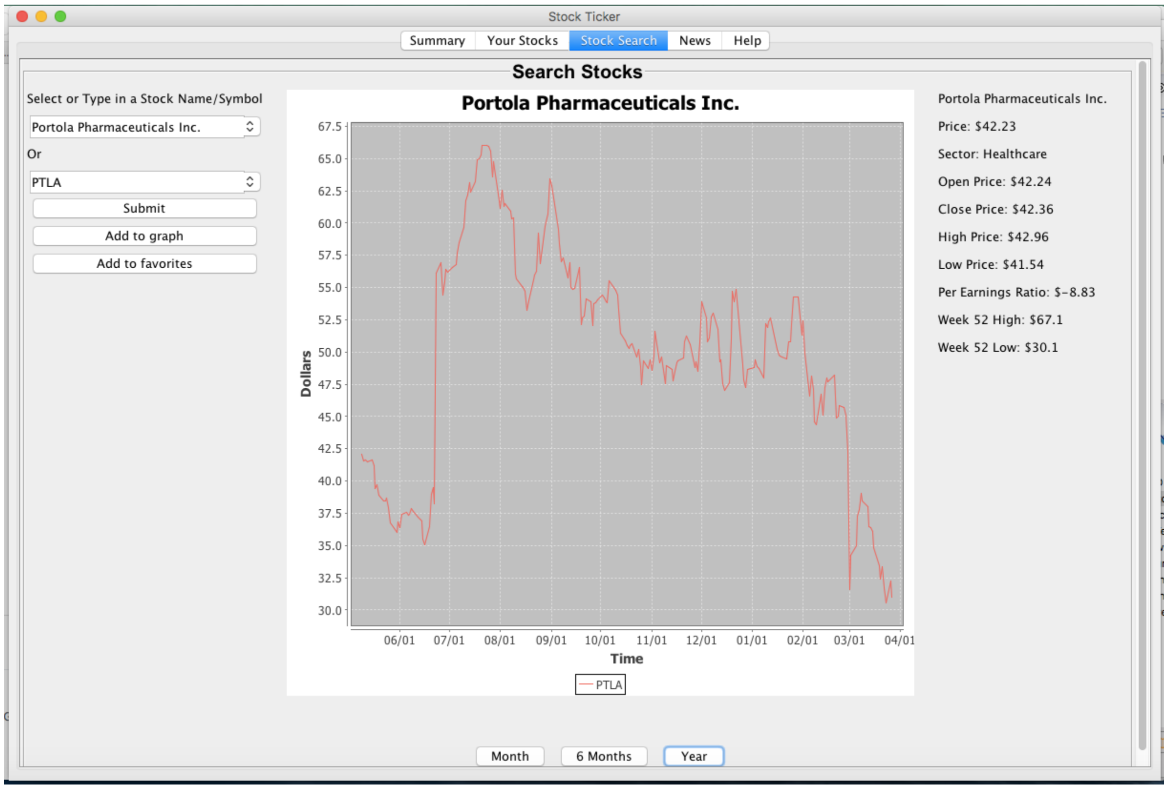
High Price: $42.96

Low Price: $41.54

Per Earnings Ratio: $-8.83

Week 52 High: $67.1

Week 52 Low: $30.1



Search Stocks

Select or Type in a Stock Name/Symbol

Portola Pharmaceuticals Inc.

Or

PTLA

Portola Pharmaceuticals Inc.

Price: $42.23

Sector: Healthcare

Open Price: $42.24

Close Price: $42.36

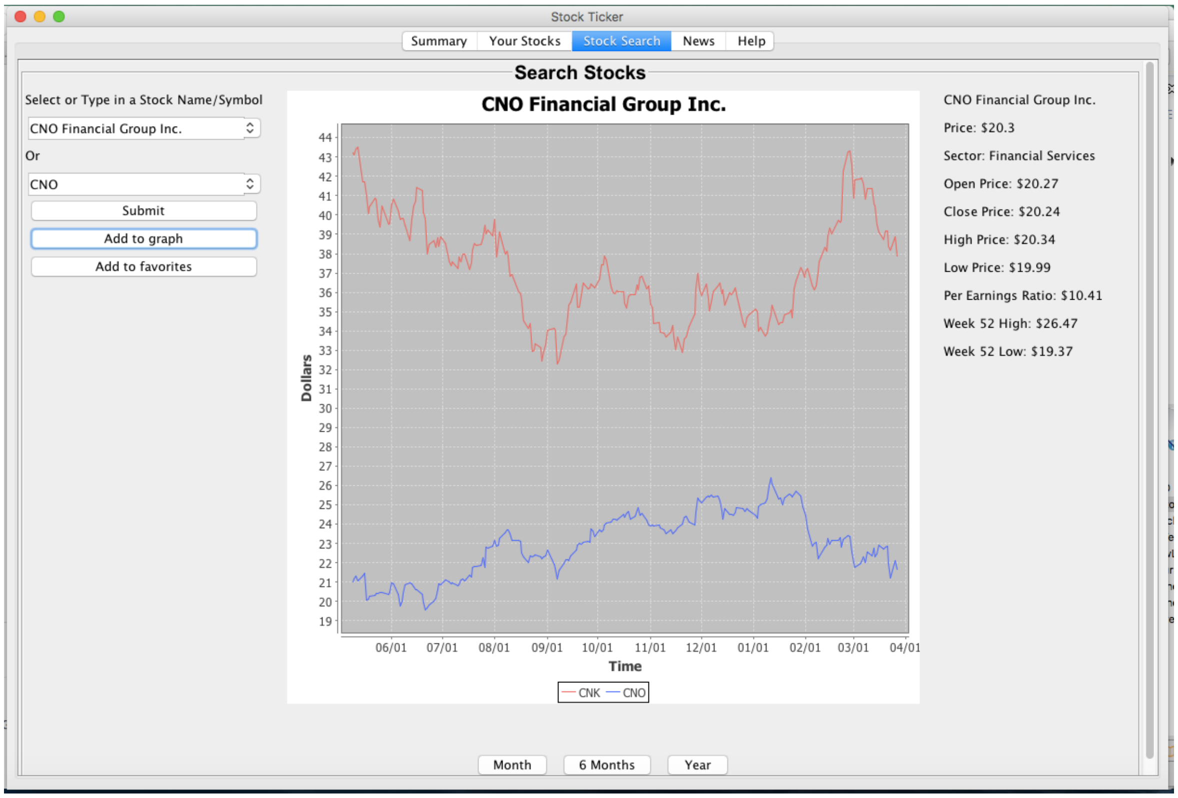
High Price: $42.96

Low Price: $41.54

Per Earnings Ratio: $-8.83

Week 52 High: $67.1

Week 52 Low: $30.1



Search Stocks

Select or Type in a Stock Name/Symbol

CNO Financial Group Inc.

Or

CNO

CNO Financial Group Inc.

Price: $20.3

Sector: Financial Services

Open Price: $20.27

Close Price: $20.24

High Price: $20.34

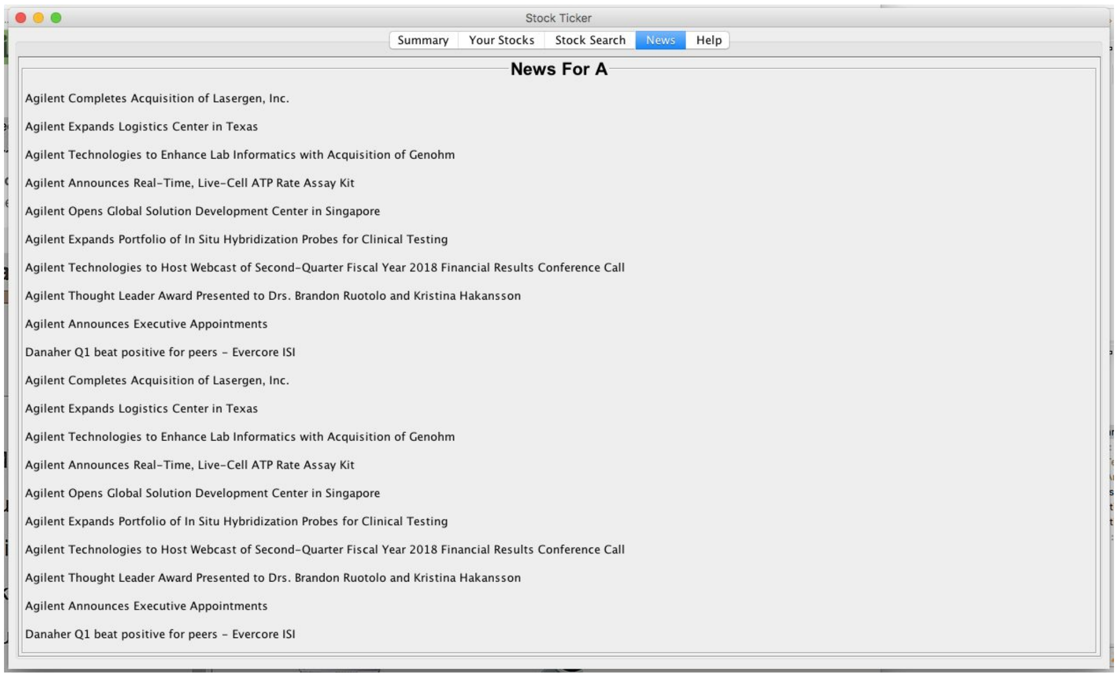
Low Price: $19.99

Per Earnings Ratio: $10.41

Week 52 High: $26.47

Week 52 Low: $19.37

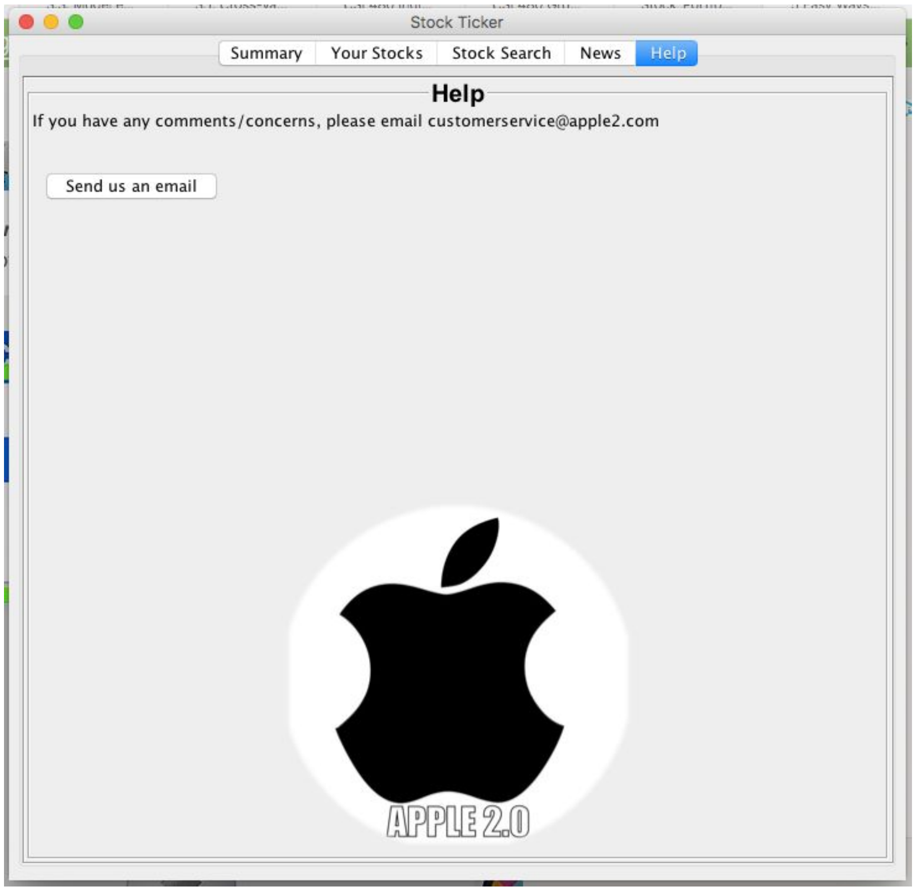
The News Panel displays hyperlinks for the stock the was most recently searched on the Stock Search tab. When the user clicks on the name of an article, it opens the article in the browser



News For A

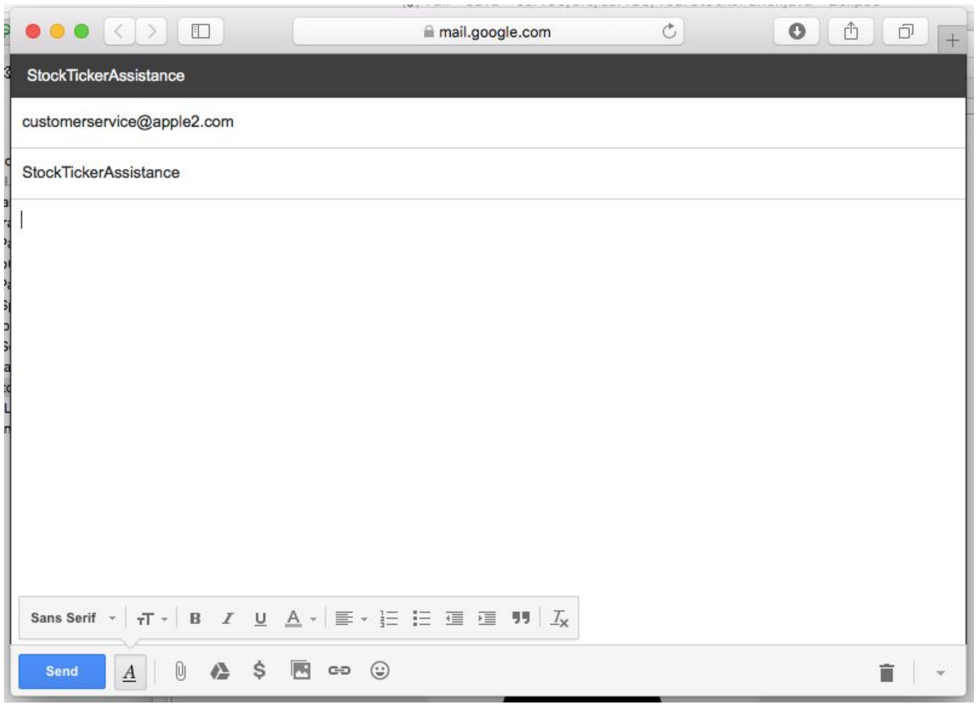
Agilent Completes Acquisition of Lasergen, Inc.
Agilent Expands Logistics Center in Texas
Agilent Technologies to Enhance Lab Informatics with Acquisition of Genohm
Agilent Announces Real-Time, Live-Cell ATP Rate Assay Kit
Agilent Opens Global Solution Development Center in Singapore
Agilent Expands Portfolio of In Situ Hybridization Probes for Clinical Testing
Agilent Technologies to Host Webcast of Second-Quarter Fiscal Year 2018 Financial Results Conference Call
Agilent Thought Leader Award Presented to Drs. Brandon Ruotolo and Kristina Hakansson
Agilent Announces Executive Appointments
Danaher Q1 beat positive for peers - Evercore ISI
Agilent Completes Acquisition of Lasergen, Inc.
Agilent Expands Logistics Center in Texas
Agilent Technologies to Enhance Lab Informatics with Acquisition of Genohm
Agilent Announces Real-Time, Live-Cell ATP Rate Assay Kit
Agilent Opens Global Solution Development Center in Singapore
Agilent Expands Portfolio of In Situ Hybridization Probes for Clinical Testing
Agilent Technologies to Host Webcast of Second-Quarter Fiscal Year 2018 Financial Results Conference Call
Agilent Thought Leader Award Presented to Drs. Brandon Ruotolo and Kristina Hakansson
Agilent Announces Executive Appointments
Danaher Q1 beat positive for peers - Evercore ISI

The last tab is the Help tab which has a button that opens up an email that is addressed to our customer service email.



Help

If you have any comments/concerns, please email customerservice@apple2.com



StockTickerAssistance

customerservice@apple2.com

StockTickerAssistance

Notes on further development

Apple2.0 will keep supporting the Stock Portfolio and fix any bugs that should appear. The users can contact our customer support to report bugs. There is user stories we would like to implement in the future, but as of right now the team is busy with other work that have higher priority.

Reflection on what the team learned as a group

Communication; while not a problem in our group we observed other groups struggle as a result of poor communication. In our team we were well organized and successfully divided tasks very efficiently, and we recognize the importance that has for our success. We learned to use git giving the team good version control and enable members to work on different features of the same project simultaneously.

The team also value the user stories and story points helping prioritizing tasks and predicting how much work can be done each sprint. Without the planning stage the organization would definitely not have been as good and the project likely would take far more time or fail.

Link to code repository on GitHub:https://github.com/Mgutkind3/Stock_Portfolio_Project

Appendix 1

Test of user interface: Click on every part of the interface and observe if the behavior is as intended
Test of user interface: Run the program on Windows and MacOS, check if the interface behave as intended/the same on both operating systems.
Test of Scrollbar: JLabels of such length that a scrollbar is necessary as not all the text fits in the frame, resizing the frame and observe if scrollbars appear/disappear as expected.
Test of Real time Information Access: Quandl’s test URL and IEX trading API, which retrieves a JSON string of real time data.
Test of Graph data to set specific time range: by using Quandl’s test URL, makes this easy to test and do.