Final Report

1. Prologue



1.1. Team Wasserfall

1.2. **Members**: Zhengqi Yang, Isaiah Lloyd, Erin Sauter, Qiuda Lyu

1.3. **Project Name**: Stock-Ticker

2. Sprint I report:

2.1.

**Scrum Master name**:Isaiah Lloyd

**Product Owner name**: Qiuda Lyu

**Team members’ names**: Zhengqi Yang, Erin Saunter

**Estimate of total person-hours spent on all aspects of job until now**: (Erin - 7 Hours)

2.2.

During the team’s first sprint, the main focus was upon user stories that defined the project’s design of the User Interface (UI). One of the two user stories selected for the sprint was that the software needed to be capable of displaying a virtual watchlist (portfolio) to keep track of a particular set of stocks showing both their ticker symbol and current price. Another user story required the software to include a graph showing both the real life prices and historical data of a stock using different time interval buttons to manipulate the display. Currently the team has chosen the APIs that they favor for use to provide the data but has been incapable of implementing. The team has managed to create a functional UI, which includes a list containing potential stock picks, buttons for different time intervals and a dynamic graph displaying potential stock trends. The team is currently at around 30% mark of the developmental process, and it will speed up as various core parts begin to fall in place.

2.3.

**Story Selection:**

The team had difficulty deciding upon the API to use to pull the stock market data, so the team decided to focus on the UI first to establish a clearer structure, so that the team would be able to more quickly implement and test the data structure at a later date.

2) A graph which is able to display real-time and historical data according to different time intervals.

**User Interface:**

**Design**:

The design of the User Interface is holding elements in chosen locations by placing panels within panels to hold steady, consistent locations and sizes. There are two primary columns of the User Interface, with the graph and buttons controlling the graph on one side, and the majority of elements dealing with stock on the other. The exception to this is the panel devoted to the purchase or sale of a stock which is beneath the graph.

**Requirements**:

A location to display a graph for stock prices, a method to manipulate the time scale of the stock graph, a method to purchase stocks and search for the stocks. The above to all be displayed in a visually appealing manner.

**Test Plan**:

Testing of of the User Interface was primarily done by running the program and visually conforming whether the design elements were in the desired locations and in the desired appearance.

**Implementation and issues**:

The implementation had the original issue of elements placed solely within a JPanel section, be it, north, south or center would expand to fill the section. At times this could merely be unpleasing on an aesthetic level, but at others it would have impairments upon program usage, such as buttons crowding others off screen. A second difficulty was the confirmation of the space for the graph due to the late integration of the scrum. A JLabel was used to verify that the JPanel to be used for the graph was inserting at the correct location.

**Outcomes**:

A usable Interface, with a preset location for graph insertion and buttons with action listeners already in place for integration with the API.

**Implementers**:

Erin Sauter, Isaiah Loyd, Qiuda Lyu

**Dynamic Data Graph**:

**Design**:

A simple XY axis chart but capable of both taking in and displaying data dynamically.

**Implementation details and issues**:

The graph uses the jFreeChart library to create a chart which includes a actionlistener to add random data item for display.

**Statement of outcomes after unit testing**:

The outcome is fairly satisfactory. The graph exhibits satisfactory characteristics for the next stage of development. The unit test was satisfactory in its ability to display a graph and intake data, with no issues upon integration.

**Implementer**: Zhengqi Yang

2.4.  **Integration Testing**:

The integration test took place once the various codes from the team members were capable of being combined together near the end of the sprint. The integration testing had difficulties due to the graph being coded outside of the overall UI, but the difficulty of the graph popping outside the UI was resolved by removal of a programming main and conversion of the graph’s class to a JPanel extension. After these changes the integration testing was successfully executed.

2.5.  **Scrum I retrospective**:

Erin Sauter was responsible for the button group and buttons that will control the graph’s time selection. Action listeners for these buttons have already been placed within the code, if without methods for activation at the current time. Sauter was also responsible for JLabels during the testing phrase of the User Interface prior to integration of other parts of the scrum to mark where the missing pieces of the interface would be. An additional button and action listener were also created by Sauter but were removed due to design development as were the JLabels upon integration.

Isaiah Lloyd was responsible for coding of the JFrame, the stock search bar, the button of the search bar, and the list of stocks results. These elements were completed with the use of a combination of JPanels, JTextFields, JLabels and a singular Scrollpane. Two action listeners were also added for future use of the button of the search bar. Alongside this Lloyd aided with the integration of the graph to the JPanel and did research towards the selection of an API for the project.

Zhengqi Yang created the dynamic chart that is able to display real-time data with the use of the jFreeChart library as well as a feature to apply random data to the graph for testing purposes.

Qiuda Lyu was responsible for the creation of the buy and sell buttons beneath the graph on the right hand side of the screen. He was also responsible for the creation of two action listeners to be used in conjunction with those buttons. Lyu attempted to add two new frames with the action listener.

**Product Owner’s statement of quality of product:**

Satisfactory.

**Scrum Master’s Statement**:

The amount of total effort worked on the project is about 32 hours. What need to done next is the creation of a landing pages, and to connect our UI our APIs for our backend.

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

3. Sprint II report:

3.1.

**Scrum Master name**:Erin Sauter

**Product Owner name**: Zhengqi Yang

**Team members’ names**: Isaiah Lloyd, Qiuda Lyu

**Estimate of total person-hours spent on all aspects of job until now**: 34

3.2.

During the team’s sprint, the main focus was upon user stories in two areas. The first of these areas was the implementation of the API and stories related to such. These stories included a usable search bar to search for stocks by name and a continuation of the last sprint’s story of the creation of a graph capable of showing a stock’s real life prices and historical data. The second area was the landing page and the implementation of a password system. The split of focus was due to an attempt to speed up the project by positioning team members inexperienced with API systems to work on a seperate portion that did not have any dependencies upon the market data from the API. Currently at this time the search bar will search for stocks by name, and the graph is capable of showing data from the current date. The landing page of the program is partially operable in that the User Interface is up. The password retrieval is still under development. The next sprint will be focused upon recording and displaying a user’s stock portfolio as well as completing unfinished objectives of this sprint. As of this time the Scrum Master would have the project at 45 percent completion.

3.3.

**Story Selection:**

Due to time constraints the team decided to split into two smaller two man teams in an attempt to speed up the project. The first of these two teams would focus upon integration of the API with code from the previous sprint while the second would focus upon the landing page as an entity that would not need information pulled from the stock market - which would implicitly require integration of the API.

**Landing Page:**

**Design**:

The design of the User Interface is holding elements in chosen locations by placing panels within panels to hold steady, consistent locations and sizes. There are five columns upon the Landing Page, which in order contain; a JLabel declaring the page title, two JPanels bearing JPasswordFields, and two JButtons beneath these JPanels.

**Requirements**:

A landing page with a hashed password login, with a method of switching between normal login mode and changing the password to access the portfolio.

**Test Plan**:

Testing of of the design of the Landing Page was primarily done by running the program and visually conforming whether the design elements were in the desired locations, and using visual elements to test whether if statements and actionlisteners were properly responding to stimuli.

**Implementation and issues**:

The implementation had the original issue of the original plan of switching JPanels not being feasible. The design team of this section switched from a BorderLayout to a GridLayout for the design of the page to allow for greater numbers of elements to be placed at once - allowing for elements to be hidden or revealed by altering the setVisible property to true or false. While functional the design has notable blank spots upon the landing page from where a Jpanel or JButton is located while being hidden from the user. Spacing also had to be adjusted to prevent the elements from crowding the top portion of the page. Passwords from the JPasswordFields were eventually extracted without complication.However hashing of the password has not been successfully implemented. The Landing Page has at this time not been integrated with the rest of the Stock Ticker.

**Outcomes**:

A usable Interface, with preset locations for JPasswordFields and Jbuttons with actionlisteners in place.

**Implementers**:

Erin Sauter, Qiuda Lyu

**Result Panel and Day Graph**:

**Design**:

Search for stock in from the API make a graph for the day.

**Implementation details and issues**:

The data had to be manually parsed because the JSON parser was not working.

**Statement of outcomes after unit testing**:

The outcome is satisfactory. The Day Graph can look a little weird by the data is accurate. The results were checked with a google finance website and it matched. Also, the result panel grab the correct data.

**Implementer**: Zhengqi Yang, Isaiah Lloyd

3.4.  **Integration Testing**:

The integration of the API has been successful with the code from the previous Scrum with code allowing functionality to the search bar, graph and one JButton associated to the graph working as intended. However, the code associated with the Landing Page has not been integrated due to the lack of functionality in reading passwords.

3.5.  **Scrum II retrospective**:

Erin Sauter was responsible for the layout of the Landing Page, from the change of design from a BorderLayout to a GridLayout, spacing of design elements upon this page, and general experimentation as to the possible ways to hide or replace JPanels for an interface that was not confusing to users. Sauter was also responsible for the creation of the actionlisteners, JPanels and JButtons upon the page. The code for the actionlisteners of JButtons to enact upon activation of the page was also the work of Sauter. Responsibility for solving the failure of the input of the JPasswordFields was originally Sauter’s but was transferred to Qiuda Lyu. Sauter was transferred to the task of implementing password hashing - a task still being performed at the end of this scrum.

Isaiah Lloyd was responsible for coding of the searching function for result panel and the get the day graph up and running. The searching function had to be manually parsed because the JSON Parser was not working. The searching function search for any stocks that start with the letter that in the Jtextfield. The Stock button upon activation will give the stock price for every minute of the day which is graphed in the day graph panel on the left side of the screen.

Zhengqi Yang was able to connect the parsed data with the display graph, and the graph shows peak prices with red indicator lines, all real-time prices with green lines, and finally low prices with blue indicator lines. The graph is currently able to display historical data. The real-time and historical data are able to be displayed through four different JButtons and after a certain stock has been searched and selected, the graph can be manipulated through the four different buttons on the top of the chart.

Qiuda Lyu took over responsibility of the task of solving the issues associated with the JPasswordFields and was successful.

**Product Owner’s statement of quality of product:**

The quality of this product is at a satisfactory level, if perhaps slightly delayed.

**Scrum Master’s Statement**:

I am personally displeased with the conclusion of this sprint due to issues in the lead up and failures in finishing tasks. The landing page is created - with the failure of any hashing protocol for password storage while also failing to be integrated with the rest of the program. The next sprint will be focused upon the creation of a page to display a user’s portfolio as well as completing unfinished objectives from this sprint. As for preparations for the next sprint: effort would likely be most rewarded by focusing upon the productivity of the team above all else.

* Erin Sauter

4. Sprint III report:

4.1.

**Scrum Master name**:Zhengqi Yang

**Product Owner name**: Erin Sauter

**Team members’ names**: Isaiah Lloyd, Qiuda Lyu

**Estimate of total person-hours spent on all aspects of job until now**: 32 Hours

4.2.

The focus of the final sprint was split three ways. The first focus was upon implementing cardlayout to allow transitioning from the landing page to the search page as well as transitioning back and forth between the search page and the portfolio page. The creation of the portfolio page was the second focus of the sprint including the layout, with the third focus being on back end work supporting features of the portfolio page. Of these objectives the transitioning between pages was fully successful while the portfolio page though created is still lacking functionality.

4.3.

**Story Selection:**

The story selections of this final sprint were based around the goal of creating a functional product. As such critical core functionalities that were seen as necessary for a final product were selected such as the ability to transition between pages, the creation of a portfolio page and support for functionalities of the support page. The broad selection of stories worked upon at this juncture was due to the breadth of work that needed to be completed.

**Cardlayout Transition System**

**Design**: Connecting three different JPanels using CardLayout

**Requirements**: Make a new on all three of the JPanels that we made or take an existing button on those JPanel. These buttons when clicked will move the user to the the major JPanels of the project which are the Landing Page, Portfolio Page, and Search Page.

**Test Plan**: Testing of of the User Interface was primarily done by running the program and visually conforming whether the right JPanel is being displaying at the right times.

**Implementation and issues**: The first attempt we had major trouble we got a fraction of the Card Layout to work, but it was slow and buggy. So we talk to Caitlin about the Card Layout and she was able to explain and teach us how to implement. The main problems was the deck was not made right and we did not pass the deck and layout to the JPanel that being displayed

**Outcomes**: All the JPanel are connected and we are able to move through the major JPanels

**Implementers**: Isaiah Lloyd, Zhengqi Yang

**Data Storage System:**

**Design**: Two different sections, one being a text file to pull and store information from the API, and a Record Class for iterative functions and calculations. The Record Class’s design includes a dictionary to allow simplistic access to a Stock’s designation within the array system of the class due to the array position being tied to a stock’s name within the dictionary.

**Requirements**: The ability to store and access information from the API as to perform calculations as well as record the number of stocks purchased as well as which stocks. Functionality to allow access to the information to perform calculations as to whether the user has sufficient money to purchase a stock as well as calculating the value of currently owned stock.

**Test Plan**: Each section was tested separately, with the Record Class in particular being tested with each complete function.

**Implementation and issues**: The primary issue of the Data Storage System is based upon a miscommunication between implemementers leading to the two sections of the Data Storage System being unconnected due to the format of information to be passed up and to be accepted being incompatible. A secondary issue for the Record Class was due to its positioning within the system for accessing information, in part due to simplicity of the organizational system for that information, further functionalities were inputted as need for the information arose elsewhere

**Outcomes**:

The completed functionalities of the Record system is operational, such as the ability to remove or add stocks to the dictionary. However, key functionalities such as pulling information from the other section of the Storage system is inoperable and not implemented.

**Implementers**: Erin Sauter**,** Isaiah Lloyd

**Portfolio Page:**

**Design**: The design of the portfolio page is fairly similar to that of the search page. There are two main panels dividing the scrollpane and the labels with the display graph.

**Requirements**: The portfolio page has to show a watchlist containing a list of stocks that the user is interested and the user is also able to buy a certain amount of stocks so that it simulates real-time stock trade. The user should be able to see a display of the total value of the portfolio and the percentage change based on the original value. Historical data should be displayed in a graph on its alone.

**Test Plan**: Testing of the portfolio page is separated into two parts as well just like the design of the page. The testing of the watchlist is associated with the data storage of the search page, so that part of the testing is done independently. The other part containing indicators and graph was done later.

**Implementation and issues**: The implementation was also done in two main parts. After creating different panels, different implementations were done according the two main panels. One implementation involving data storage and connection with search page was done on one side, and the other part of the portfolio page was implemented fairly easily. There were several issues associated with the implementation of the watchlist. For instance, the watchlist was not able to be refreshed when we first implemented it, but later on, many of the issues were resolved.

**Outcomes**: The outcome is fairly satisfactory. The design is largely completed but there are minor issues that were not able to be resolved. However, the large picture was materialized, and the page looks crisp and contains the necessary functionalities intended.

**Implementers**: Zhengqi Yang, Isaiah Lloyd, Erin Sauter, Qiuda Lyu.

4.4.  **Integration Testing**:

There were not many integration problems in this sprint. This is because all of the code write was on the Portfolio page which mainly use to display data. The biggest problem that we had was an error that was caused by a merge the files in github. This error made a duplicate for the Portfolio Page class constructors in the code causing problems making us have to find and remove the duplicated code. After successfully implementing the CardLayout to connect all of the JPanels and finishing up the portfolio page, the integration was fairly straightforward.

4.5.  **Scrum III retrospective**:

Erin Sauter was originally tasked with hashing of passwords to finish work left from the last scrum. Due to time expenditures upon this task becoming unmaintainable Sauter was moved from this to other duties with the password hashing being scrubbed from the project. After the cancellation of the password hash Sauter worked upon the coding of the backend. Specifically, the Records class, which included the design of the class, the functions and the logic behind the updates for addition or removal of stocks. With the addition of the Budget class - the design of the class, functions and the decision of tying the Records class as a supporting element. This included going back to the Records class and applying additional functions to support the Budget class. Sauter was also partly responsible for code writing for the button creation for the stock list upon the Portfolio page.

Isaiah Lloyd was responsible for coding to fix the JfreeChart labeling issue of the X axis, help work on the card layout system, and made a simple storage system to hold stocks that User want to store for later. Isaiah had to go ask for help with this from Catlin Croke for the card layout system. Isaiah also talk to Mike Gutkind about making the x-axis in the Jfreechart. Those two tasks were successfully completed. The simple storage works but only for interested stock. The simple data storage uses a text file to store data and the program will not allow duplicate of stock in interested stock folder.

Zhengqi Yang was mainly responsible for implementing the CardLayout, and with the help of Caitlin and Isaiah, he was able connect all three panels together. He was also able to connect able to connect the parsed data with the display graph, and the graph shows peak prices with red indicator lines, all real-time prices with green lines, and finally low prices with blue indicator lines. The graph is currently able to display historical data. The real-time and historical data are able to be displayed through four different JButtons and after a certain stock has been searched and selected, the graph can be manipulated through the four different buttons on the top of the chart. Zhengqi also contributed the design and math calculation behind the display of several important indicators on the portfolio page. The change percentage now has a color distinction associated with it (green means positive change and red means negative).

Qiuda Lyu designed the whole Portfolio Page frame that there are two part which are leftPart JLabel, rightPart JLabel. In leftPart JLabel, there is an JScrollPane which will hold the stock button which will be added from the search page. In the rightPart JLabel which is GradeLayout(2,1), there is a top JLabel which is GradeLayout(3,1) and holds three small JLabel that will be show total property, total percentage change and total gain. Qiuda also worked on the Favorite Stock store. In this part, the main work is that Isaiah make a way to store the user’s favorite stock name into a file line by line. Then, Qiuda got the way to read every line from the file. As for each line will be the name of the stock button’s name. Those stock button will be store in the JScrollPane.

**Product Owner’s statement of quality of product:**

I am personally displeased with the product, due to in part the incomplete nature of several functionalities, the lack of connection between the record keeping classes and the information draw from the API for example. I am also displeased with the form of the program as I find while functioning components are functioning the form of those components are visually rough.

* **Erin Sauter**

**Scrum Master’s Statement**:

I am personally very satisfied with the progress that we have made during this particular sprint. It was certainly the most difficult sprint that we have experienced so far due to the close proximity to the final exams and projects. I have delegated different tasks to different team members and those tasks were largely completed with a very little exception. Agreeing with the product manager, I do not think that the current state of the product is sufficient or satisfactory compared to the vision we had at the start of the project, but I truly believed that one sprint after another, we as a team have made tremendous progress. This sprint, despite some lack of communication, achieved things that we would have liked to achieve in a timely manner. I believe that perhaps with one more sprint like this one, we could complete and improve the product to the state where the product manager and I would consider to be sufficient and satisfactory. -

- **Zhengqi**

5. **Goals**

5.1. Description of Demonstration:

There are three parts in our final project what is landing part, searching part, and portfolio part.

In landing part, users will get an initial password when they open the application on first time. They can type the password on the JPasswordfield to get into next main page directly or they can choose to click the changePassword JButton to change the password in the changePassword JFrame. After the password is changed, the background will record the new password. After users type the password correctly, they will get into the portfolio part.

In this part, user will get into the portfolio JFrame. In the left part, there is a JScrollPane which will hold the stock button which will be added from the search page. In the right part, There is a big JLabel which will hold top and bottom JLabels. The top JLabel holds three small JLabels that will be show total property, total percentage change and total gain. If users gain money, the values of percentage change and total gain will be green. If they lose, they will be red. In the bottom of the right part, there is a graph display JLabel which will shows the portfolio trend chart to users. On the upper right of portfolio JFrame, there is an next JButton. If users click that, they will be transferred into the last part that search part.

In search part, there are still two part. In the left part, there is also a JScrollPane. Above of the JScrollPane, there is a JTextfield and a searching JButton. User can type the name of the code name of the stock which they want to see in the JTextfield. After the typing, they can click the search JButton. The stock will will be shown as JButton in the JScrollPane. The right part is made up by a JLabel which is in Borderlayout. In the South, there are four JButton which are Day, One month, six months and Year. In the center, there is a graph display JLabel. If user click the stock button in the JScrollPane and click one of those four JButton, that stock’s trend chart in those four time interval will be shown on the display JLabel. In the North, there is an Add JButton. When user want to store the stocks which they like, they can clicked the stock JButton and the Add JButton. The name of that stock will be stored in the background text file. User can find a back JButton on the upper left which can transfer user back to the portfolio page. After user go back, the JScrollPane that in portfolio JFrame will read the user-like stock’s name and generate the stock JButton for that.

For images of the application it will be on the Github repository under in the Pics folder in the information

5.2. Further Development:

Further development would predominantly focus upon the completion of the portfolio page to be functional. Such completion would be marked among other things with the transfer of information from the txt file storing information in a usable manner to other parts of the programming. A secondary and less intrinsic to the success of the project is proper implementation of password hashing and implementation of a card layout for the two panels with login elements and password changing elements for the loading page respectively.

5.3. Team Reflection: One of the primary lessons that the team learned as a whole was the importance of time management and prioritization of critical product components. The other central lesson was the importance of communication to prevent the need for code to be rewritten due to misunderstandings between team members.

5.4. Code Repository: https://github.com/IsaiahL12/Stock-Ticker-Project