 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**

**Signature of Product Owner:**

X\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Signature of Scrum Master:**

X\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Signatures of Team Members:**

X\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

X\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_