Scrum Report Format

1. Prologue
   1. Team Name: Nird
   2. Team Slogan: Nirds of a feather
   3. Team Members: Patrick Gallagher, Spencer Russell, Tianyi Zhang, and Hank Zhu
   4. Project Name: Stock Ticker
2. Sprint 1 Report:

2.1 Teammates Roles

* 1. Scrum Master: Patrick Gallagher
  2. Product Owner: Tianyi Zhang
  3. Team Members: Spencer Russell and Hank Zhu
  4. Combined Hours on the job: ~15 hours

2.2 User Stories

The first sprint we worked on the connection of the API and the GUI for the program. The API that we are using is AlphaVantage and the project is wrapped in Gradle to help us connect the API with AlphaVantage. The GUI has three frames that connects to each other when searching for a stock, clicking on the searched stock, and adding the stock buttons. The user stories that we chose were:

* As a programmer, I want the program to be in a simple GUI interface. So that the program can be easily used by everyone. – 1pt.
* . As a programmer, I want the project to have an API to disconnect and connect automatically. So that the application can display the stocks in real time. – 8pts.

We have connection with the API, also the GUI can pop up all three frames when clicking on the buttons that revolves around the other frames.

2.3 Design, Requirements, Test Plan (GUI User Story)

The GUI user story’s design had three frames, one for the menu, a frame for the searched stocks, and the third frame about the description of the selected stock. The requirements for the GUI were the panels, buttons, and a few of different functions for the frames, like the buttons going to different frames when clicked. The menu frame has only one requirement that when the user hits enter it goes to the second frame. The second frame has an array and a for loop that makes buttons for each item in the array. When clicking on one of the buttons, the third frame will show up with the description of the stock and an option of adding the stock. The testing in this sprint was testing for what each frame looks like and the testing on for each button that connects to another frame.

2.3 Implementation details and issues (GUI User Story)

We have implemented frames, labels, buttons, panels, and other java applications in make the frame. The issues when we implement the java.awt’s is the positions of where the panels go into the frame. The other issue is that when we do implement the ActionListener, some classes will not co-operate with each other and we had to fix some of the code.

2.3 Statement of outcomes (GUI User Story)

* The outcome after testing is that all the frames work and finally connect between each other.

2.3 Names of Implementer(s) of that story (GUI User Story)

* Patrick Gallagher has worked on the menu frame, the first frame, Hank Zhu worked on the second frame, and Spencer Russell worked on the third frame.

2.3 Design, Requirements, Test Plan (API User Story)

* The second user story was working on the connection of the API. The design for the API is that it calls the stock numbers for the day. Also, the API grabs the stocks information and the stock name. The requirement for the user story is that we just want to have a connection for the API. TEST CASES

2.3 Implementation details and issues (API User Story)

* The implementations on API is using patriques and we had to wrap the project in Gradle to get Project and External Dependencies jar files. Wrapping the project in Gradle will make project able to connect to AlphaVantage. The only issue is that the project needs to be wrapped in a Gradle project to make the project to work.

2.3 Statement of outcomes and Names of Implementer(s) (API User Story)

* The outcome of the testing is that the description of the stocks and how high or low the stock numbers. Tianyi Zhang implemented the API to the project.

2.4 Integration Testing

* There is only one integration test that the project had was the connection between all three frames. All three classes used ActionListener, but only two classes use the other classes frames names to call the class frame. The third class, stockName, uses a constructor in the display class called display.run() to run the display frame. The quality of the product is that all the buttons makes a new frame from a different class shows up. Everything works in the project that is supposed to work and there are no errors.

2.5 Scrum 1 Retrospective

* Hanks’ perspective: I made a panel, which services as a container for favorite stocks that user added personally. I created string list that contains all the stock names that user added, and I made a “for” loop to show these stock names in the panel vertically. My plan is to make a favorite stocks container that user can either add or remove the stock, when user click on the stock, different box pops up with detailed information about the stock that user clicked. For elaboration, I want user had a preview of what their favorite looks like without click on it, and user can also customize the stocks order by clicking and dragging it.
* Spencer’s Perspective: My panel has been designed and cannot currently be improved due to the lack of API. This panel is supposed to help with the one user story involving understanding the history of stocks by displaying all relevant information. Another possible story relation may be to the given functionality of adding stocks to a personalized list.

The panel I [Spencer] designed is going to be implemented as window for presents information related to the stock including the stock listing abbreviation, full business name, a brief description of the company, and a graph that displays the rise and fall in stock value for the past day, week, month, and year of information, and a button for adding the stock to the user’s collection. Currently, the window is lacking functionality as the API has yet to be implemented so no information is being displayed. I need to better apply my time, possibly by assisting others or seeking assistance for myself.

* Tianyi’s Perspective: In this Sprint, I oversaw working with the API. I spent most of the time try to understand how to get the JSON data become Java beans. I found this API service of stocks called “Alphavantage” and a wrapper called “alphavantage4j”, with them I can have all kinds of stock information. Then I spent around three hours making the daily graph of a single stock. It can display the price and volume of a stock in one day from 10am to 16pm in every 60 minutes. This graph can be called with the command StockHourlyGraph.run(Lday,Lday.get(0).getDateTime().toString()), where Lday is a list that contains information objects of a single stock
* Patrick’s Perspective: In this sprint I have made the menu frame. The frame has four different panels. One panel for the user information, the user name and other information. The second panel is the search function that allows the user to search what stock they want to look for. The third panel is all the added stocks that the user has bought or favorited, so far, we put random items in the array to show that the array works in the early version of this project. The last panel is the information of an added stock that the user has clicked on. The search bar, when hitting enter, Hank’s frame will show up.

2.5 Product Owner Statement

* By the end of the Sprint I, the team members have finished the design of the GUI of the Stock Ticker. The frames are runnable and are linked with each other. But they cannot work with the functionality of saving the stock preference or display the graphs, which will be the goal to achieve in next Sprint. Also, the GUI does not look aesthetic, we expect to improve its design after having it works functionally. The team has the first draft of the one-day graph of a stock made, it successfully presents the key information, but the team will need a discussion on the standard of all the graphs. The API works fine with displaying the stock data within a day. But sometimes the API does not connect until several runs. The cause of the error is still unknown.

2.5 Scrum Master Statement

* Everyone has worked on this project around 15 hours combined in this sprint. We would have the API connected to the entire project instead of having just a connection that does not work along with the GUI. There are going to be arrays that have all the items in the API and another array that has the stocks name that the user has searched. Also, the group will start making graphs and descriptions for stocks.

2.5 Set up for Sprint 2:

* As a user, I want multiple data displays when I am looking at a stock. So that I can have an easier time looking at a stock. – 4pts.
* As a user, I want a search function when looking for stocks to add, so that I can find stocks easier. – 2pts.
* As an investor, I want to keep hold of my personal shares. So that I can monitor my own financial gain. – 4pts

Signatures:

Scrum Master Name (Printed) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Scrum Master Name (Signature) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Product Owner Name (Printed) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Product Owner Name (Signature) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Team Member Name (Printed) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Team Member Name (Signature) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Team Member Name (Printed) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Team Member Name (Signature) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_