Project Report: Stock Tracker

1. Introduction

This a solution for a customer who wants to be able to track specific stocks on the New York Stock Exchange, by customizing their experience. The user will be able to select their stock of interest, input their required upper and lower threshold values and monitor it through a user-friendly dynamic graph. They now will not have to search through multiple stocks and can easily be notified when desired values are reached, making their investment journey more efficient, helping them make better, well-informed, and timely financial decisions.

Some things that have been utilized to build the requirements of this project are asking potential customers for input, doing a competitive analysis of current similar tracking tools, and reviewing business and policy procedures (to make sure everything is in accordance with trading regulations). Doing these things will give us a strong base on how to build our project. Next, diagrams (use-case or activity) will be used to show these requirements and actors will interact with each other, like the data from NYSE, customer, and system. An agile driven (possibly incremental) approach is used.

1. Project Goals and Problem Definition

The Goal of our project is to create a web application that allows users to track price fluctuations in their stocks of interest through an interactive user-friendly interface that displays a dynamic graph of stock prices in real time, with a notification system that notifies the users when their desired values are reached.

To attain our project goals, we must first form a well-defined problem definition. We must thus:

* Implement a feature that allows a new user to input their username and password to sign up, after which a notification pops-up indicating that the user has signed up
* Implement a feature that allows the user to input their username and password to login, after which a notification pops-up indicating that the user has logged in
* integrate a functionality that allows users to choose the stocks they wish to monitor
* input a feature that allows users to input their pre-set threshold and loss-stop values.
* Implement a checker in place to ensure values inputted into the system are valid (no letters, special characters, etc.)
* Implement a notification system in place that will inform the user if the pre-set threshold and stop-loss value is reached.
* Implement a dynamic graph that will display the past week/month price trends of the selected stocks

1. Project Deliverables

This section documents our team’s goals for the product release. Our project deliverables will be divided into the following points.

1. business operations, performance, and requirements analysis reports: a brief report on the overall performance of the management team and product delivery team.
2. software specification documents and systems architecture: has a list of user, business, and functional features required by the stakeholders that are to be implemented on our web app. for example, our application must

- be able to import data from NYSE in real-time

- display a login and sign-up page

- display a search bar for the user's entry of the specific stock they want to track

- output the user input’s stock's chart

- notify the user of abnormal fluctuations in stock price and when the right time to invest is

1. final product(external deliverable): investment tracker software source code and project binder.

final product: investment tracker software source code

user input: name of stock and user account

system output: dynamic graph that displays stock prices in real-time and sends notifications during abnormal stock price fluctuations

1. prototypes and preliminary models that serve as a framework for the whole project and their progress reports and error logs can help estimate the trajectory of the entire project. We will make our prototypes with working code and use case, sequence and activity diagrams.
2. risk management, refactoring, and version control report: this report consists of performance logs measured against baseline reports and previous system efficiency logs. This is essential as it will keep track of problems faced during our project and their solutions which might come in handy at later stages in the process. This log also serves as a way to communicate within the team when in need of problem-solving strategies and troubleshooting. It also keeps a log of the various versions of the source code and helps us improve the efficiency of later versions.

Our deliverable will be released in the form of the following project binder:

|  |
| --- |
| project binder |
| 1. the project report |
| 2. all versions of the software source code |
| 3. test cases and test data and return reports |
| 4. PowerPoint slides with overall project description |
| 5. video presentation of proposed solution |
| 6. two use case descriptions and diagrams: for brokers and investors |
| 7. system architecture |
| 8. sequence diagram |
| 9. activity diagram |

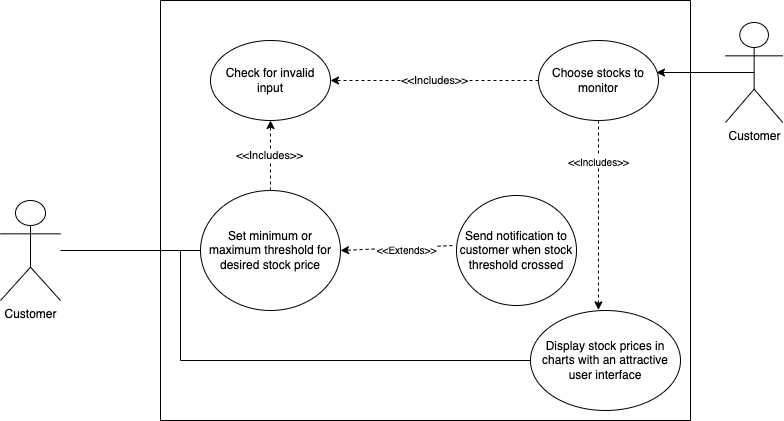
**UML Diagrams**

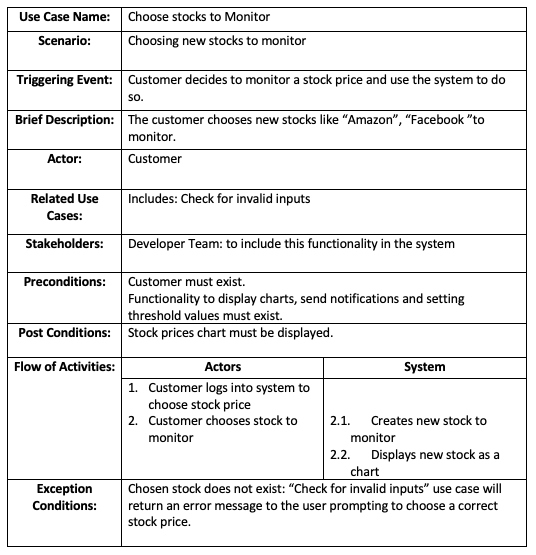
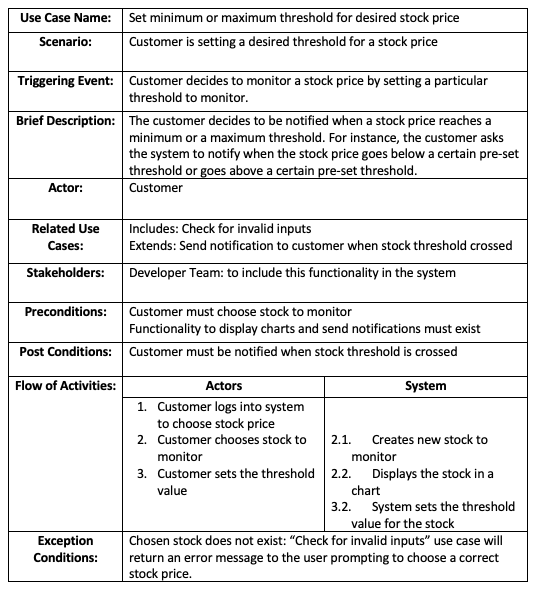
Making these during our planning stages was very helpful to our project team as it allowed us to think through the intricacies of the project before jumping into the code.

4. Use Case Diagrams and Descriptions

The following use case diagram and descriptions were constructed in order to help us document and visualize the interactions between the user and the system we are creating. The interactions between the system and the user (actors) are shown with the help of a directional graph that makes use of <<extends>> and <<includes>> arrows.

The customer has to be able to choose a stock to display its chart and be able to choose a threshold stock price. When the client chooses a stock, the stock of that particular company needs to be displayed in our user interface. When the customer chooses a threshold value, the customer should be notified when that threshold value is breached by the stock





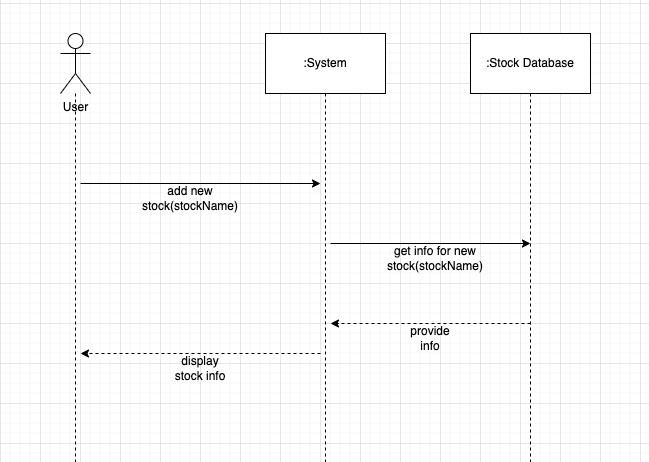
5. Sequence Diagrams

The following sequence diagram and descriptions were constructed in order to help us document and visualize different processes that the system performs. The processes are arranged in the order of a time sequence and are syntactically shown with the help of asynchronous, return and synchronous arrows.

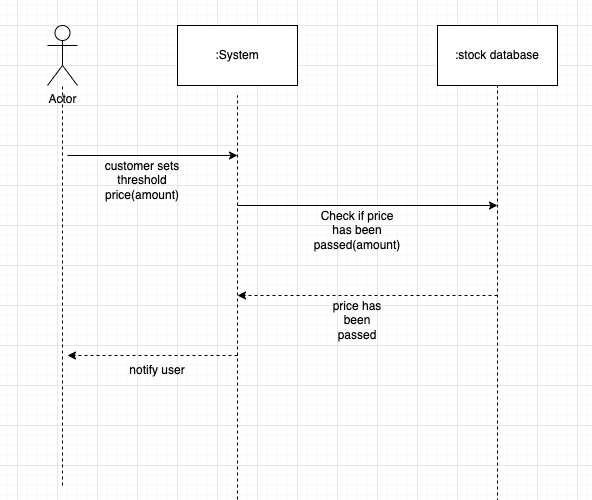
It highlights the users ability to search up and add a stock. The user inputs a stock name to the System. The System makes an API call to the Stock database, which in turn fetches the stock requested and its corresponding data. The API used for our application is Alpha Vantage. The data is then returned to the system and displayed for the user.

We have two sequence diagrams for this system:

1. The diagram for adding new stock:



1. Diagram for checking price threshold:



6. System Architecture

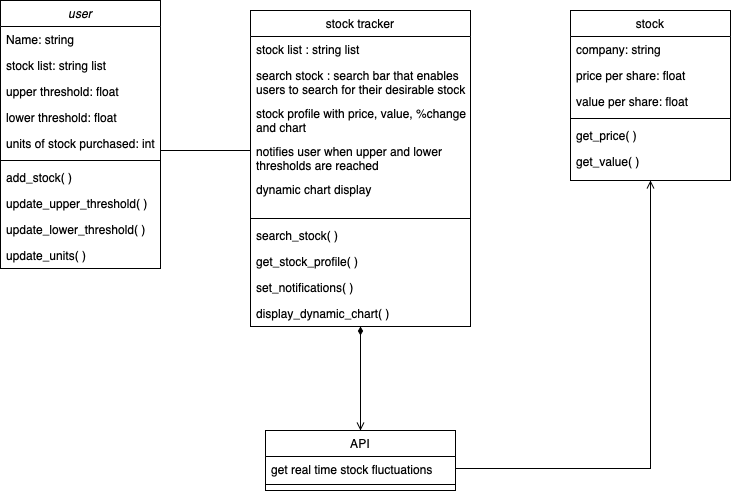
For our application architecture, we made the choice of using the Model-View-Controller Architecture.

This is because:

* It is easier to pull data from a model of the stock market.
* The controller aspect is integrated because we make decisions based on user actions and breakpoints to decide what to display on view.

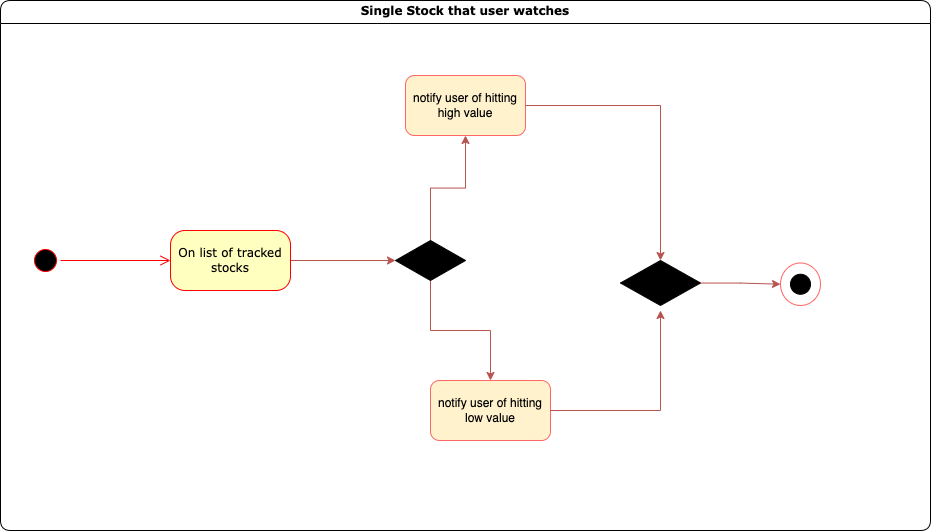
7. Detailed Class Diagram

For further clarity on the system’s classes, attributes, operations and their functionalities; we make use of a detailed class diagram that gives us a static structural description on the relationships among objects in the system. The three main classes are user stock tracker and stock. The user class gets user input for the given attributes. The stock tracker calls the API to get the stock values and chart. This is then passed to the stock class.



8. State Machine Diagram

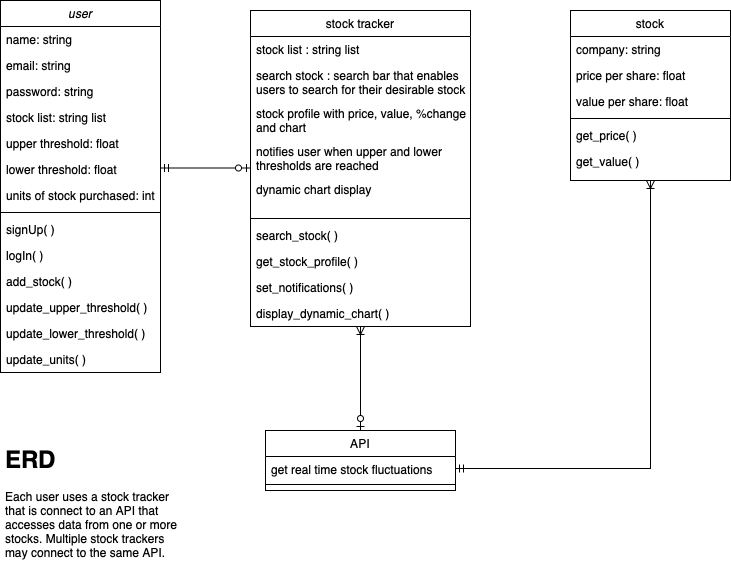
We make use of a state machine diagram in order to show the transition between states in an object. An object is seen to change its state as a result of a change in past/current input.



The black filled dot displays the start-state and circled dot the end-stage. This specific diagram represents the notification states that the System implements. Once a list of stocks to track has been administered, the System can notify the user when the set highpoint or low point have been reached. On successfully notifying the user, the state terminates.

9. ER Diagram (Data modeling)

The ER diagrams show the modality and cardinality of the relationships between objects in the entity. We chose to use an ER Diagram as our App Data Model. This model represents the 4 entities in our application. The user, the stock tracker itself, the stock inputted by the user, and the API used. As seen by the relationship symbols used, each user has access to one stock tracker application. The stock tracker application has 1 user and 0 or 1 API. Each API has 1 or many stocks and each stock is connected to 1 API.

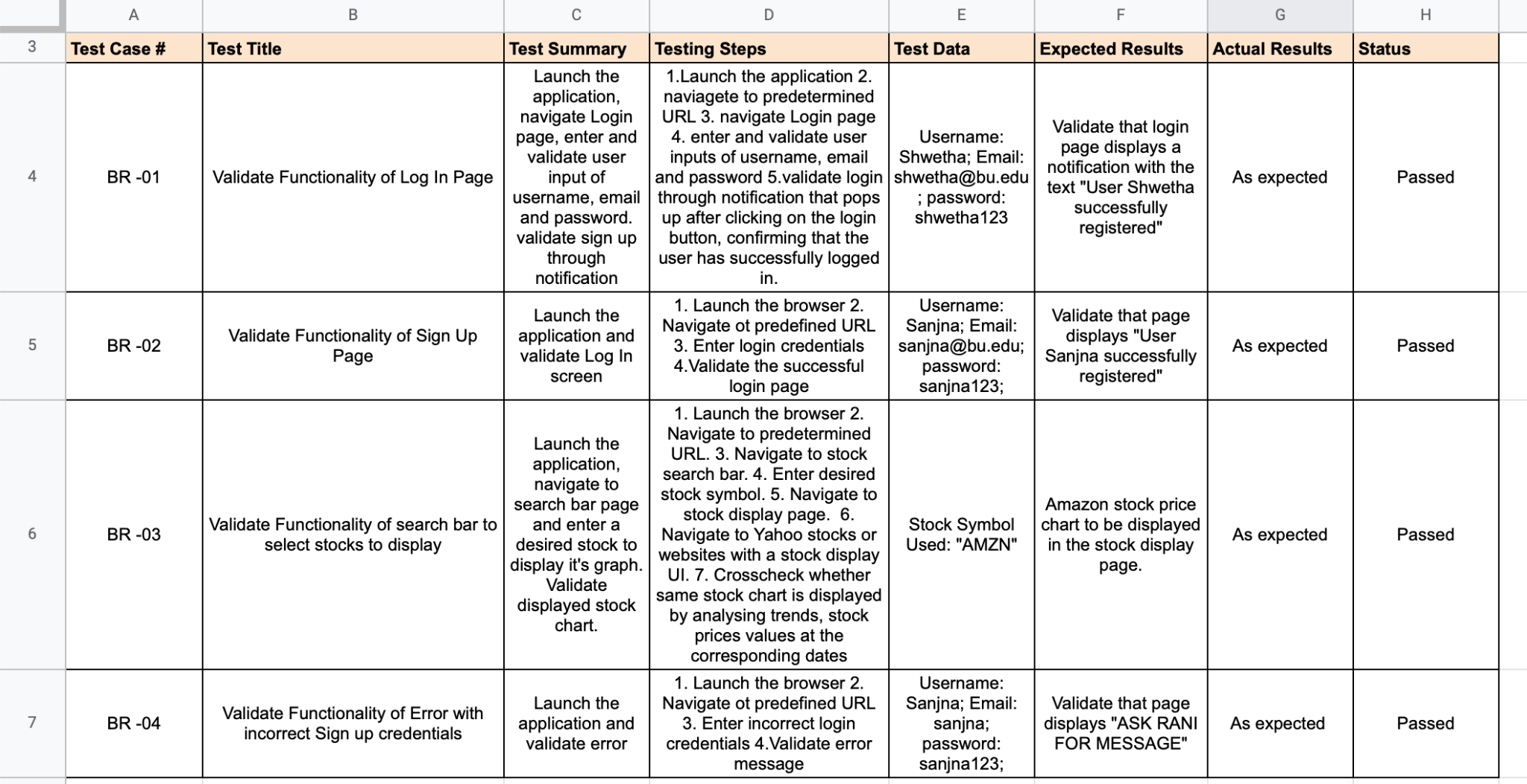


10. Test Cases

Testing is essential to our project for the following reasons:

* Product validity : To show that our product’s functionalities are implemented in the right way, and the program does what it is intended to do. This inspires confidence in both the customers and the development team.
* Defect discovery: To help the project team recognise defects and inefficiencies in the program by using artificial data to test for errors and anomalies

The testing method we chose to do is the Black-box-testing method. This technique helps us test the functionality of the program without getting into the depths of the code, internal structures and workings. It is fast, efficient and easily comprehensible for both the customers and development team.



11. GitHub Link to Project Source Code

<https://github.com/ranishah0402/cs411-stock-tracker.git>

12. Conclusion (Lessons Learned)

Our group learned many valuable lessons during the life-cycle of this project, both technical and project management skills.

A summary of the most important things we learnt are the following:

1. Communication is key

- We realized that we needed to be flexible, and work through all of our ideas to reach one conclusion on what we thought best for the project, a decision made by the whole group.

- We learned how best to communicate between our members by updating each other on the state of our project as a whole and our individual assigned work

- Being receptive to everyone’s feedback and asking for help when needed

2. Planning and being organized is essential

- We learned that planning is essential and we learned how to plan well with the help of various diagrams that helped us understand our product better: such as the system diagrams and app structure and state diagrams.

- We also used a Gantt chart to plan, track and schedule meetings and attain our project goals through stepwise tasks

- We also learned the importance of documentation

3. Project assessment and monitoring, and testing is crucial

- Through Peer-reviews and testing our product multiple times

4. Research is important

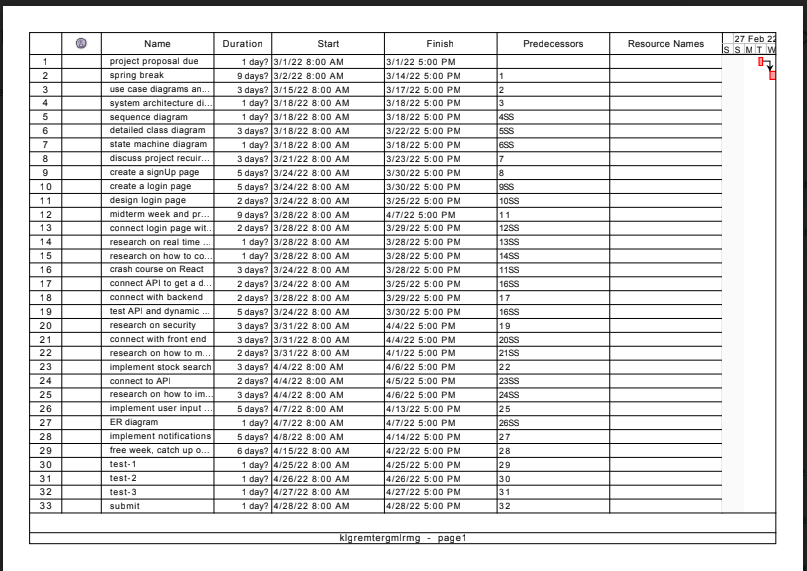
* We read through a variety of articles, looked at various other projects, and asked other people to decide what would work best for our case.

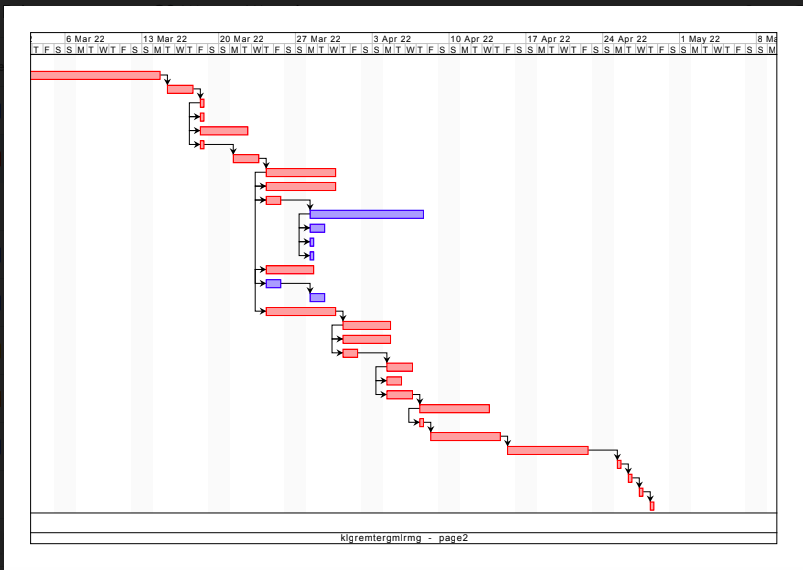
5. Other technical skills

- How to connect the front and back end, how to use an API, etc.

13. Project Work Breakdown Structure: Gantt Chart and Task Assignment Matrix.

For project planning, we made use of a Gantt chart using the *project libre* software. The Gantt chart helps us plan, track and schedule meetings and attain our project goals through stepwise tasks by illustrating a project plan, outlining a list of tasks with start and end dates and displaying a timeline with schedule bar charts showing the critical path of the project and dependencies between the tasks.





We all worked on each of the assigned tasks and coordinated with each other with updates on the state of the project. Most of our meetings were in person, and we did our work as a collaborative effort with everyone being very receptive to feedback and asking for and giving help when needed.

14. Project Monitoring and Project Closure

The Project was monitored throughout by strictly following the schedule in the Gantt chart and starting each of our meetings by reviewing the progress we made since the previous meetings and working on the feedback given.

Even though our project life cycle was well planned, we did hit a roadblock while implementing the notification system and did not have enough time to get the threshold notification to work. We thus discussed our process metrics in-order to explore how we could have implemented it if we had the time.

16. References

* <https://serverless-stack.com/chapters/create-the-signup-form.html>
* <https://www.knowledgehut.com/blog/web-development/building-a-sign-up-form-using-react>
* <https://www.alphavantage.co/documentation/>.
* <https://medium.com/bb-tutorials-and-thoughts/how-to-develop-and-build-react-app-with-java-backend-c1e6c5c93ae>.
* <https://danielmjung.medium.com/getting-started-with-alphavantage-b0efb3140a61>
* <https://docs.gitlab.com/ee/gitlab-basics/start-using-git.html>
* <https://github.com/aqd14/stock-tracker>
* <https://www.educative.io/edpresso/the-fatal-refusing-to-merge-unrelated-histories-git-error>
* <https://www.emgoto.com/react-search-bar/>