Real-time monitoring of volatile, retail investor-driven price fluctuations in the equity market

Sprint 2

Group 10

Huei-Che Chang, Lanye Pu, Wen-fang Lu, Wan-yi Mao

A. Demo Working Product System

The screenshots below are our current project

We can select a certain company to see its company information

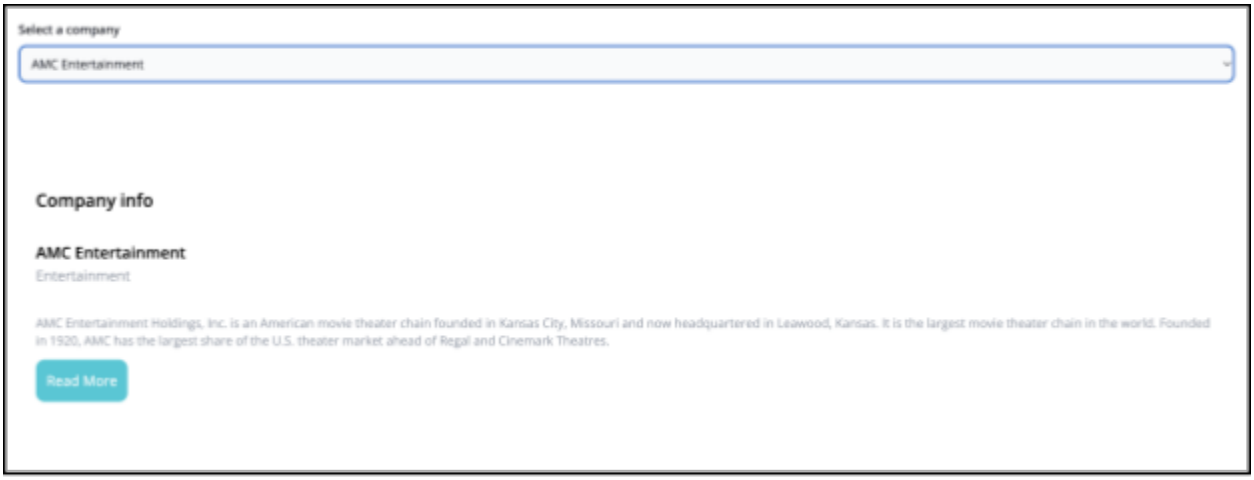

Fig. 1 User interface : company dropdown list and company information

In basic information, we can see a certain company's instrumentType, yearChange, exchangeName and other related information. Users can resize the columns to see the detailed data.

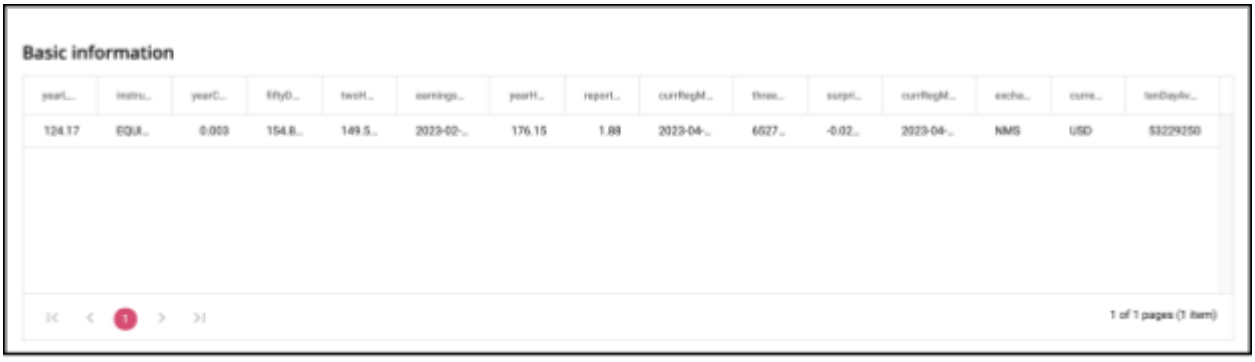

Fig. 2 User interface : basic information

In real-time reddit posts, users can see the real-time reddit posts. This part has not yet been connected to back-end api service.

Fig. 3 User interface : real-time reddit post

Users can see historical stock prices through the line chart.

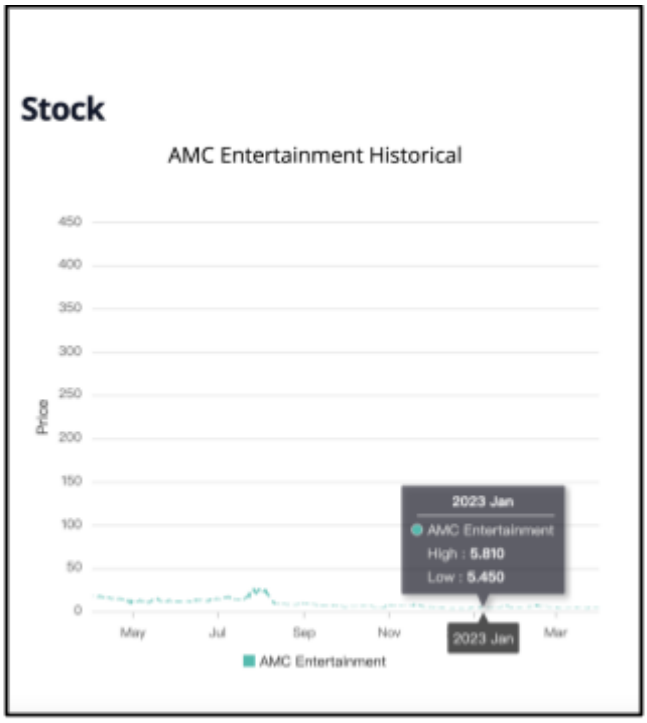

Fig. 4 User interface : stock price line chart

B. Scrum

Sprint 2 Backlog and Burndown Chart

	Task Name	ideal	actual
1	Priority-1-1 (basic stock info)	24 hrs	10 hrs
2	Priority-1-2 (yfinance API, MySQL)	48 hrs	70 hrs
3	Priority-1-3 (Maxwell)	48 hrs	72 hrs
4	Priority-1-4 (Reddit PRAW, Kafka, Spark)	96 hrs	110 hrs
5	Priority-1-5 (SpringBoot: producer, service, mapper)	72 hrs	70 hrs
6	Priority-1-6 (front end: history stock price)	72 hrs	80 hrs
7	Priority-2-1 (front end: search bar, filter)	24 hrs	24 hrs
8	Priority-2-2 (reddit post Kafka producer, early split topics)	48 hrs	30 hrs
9	Priority-2-3 (Spark Dstream with Scala)	48 hrs	45 hrs
10	Priority-2-4 (SpringBoot: Restful API)	72 hrs	90 hrs
11	Priority-2-5 (front end: reddit post display section)	72 hrs	60 hrs
12	Priority-2-6 (live update of reddit posts)	72 hrs	50 hrs
13	Priority-2-7 (keyword filter)	48 hrs	70 hrs
14	Priority-3-1 (sentiment analysis result for each posts)	24 hrs	24 hrs
15	Priority-3-2 (Kafka producer send to aggregated topics)	48 hrs	48 hrs
16	Priority-3-3 (Data cleaning, sentiment analysis)	48 hrs	40 hrs
17	Priority-3-4 (Elasticsearch APIs)	48 hrs	120 hrs
18	Priority-3-5 (front end: Pie chart for analyzed results)	24 hrs	24 hrs
19	Priority-4-1 (front end: historical data)	24 hrs	24 hrs
20	Priority-4-2 (yfinance polling to MySQL)	24 hrs	10 hrs
21	Priority-4-3 (Read/Write splitting with Maxwell)	48 hrs	48 hrs
22	Priority-4-4 (Data cleaning, sentiment analysis)	96 hrs	
23	Priority-4-5 (Finalize the SpringBoot - Elasticsearch)	72 hrs	
24	Priority-4-6 (Finalize the API endpoints)	72 hrs	
25	Priority-4-7 (Finalize all the charts: line, pie)	72 hrs	
26	Buffer	112 hrs	

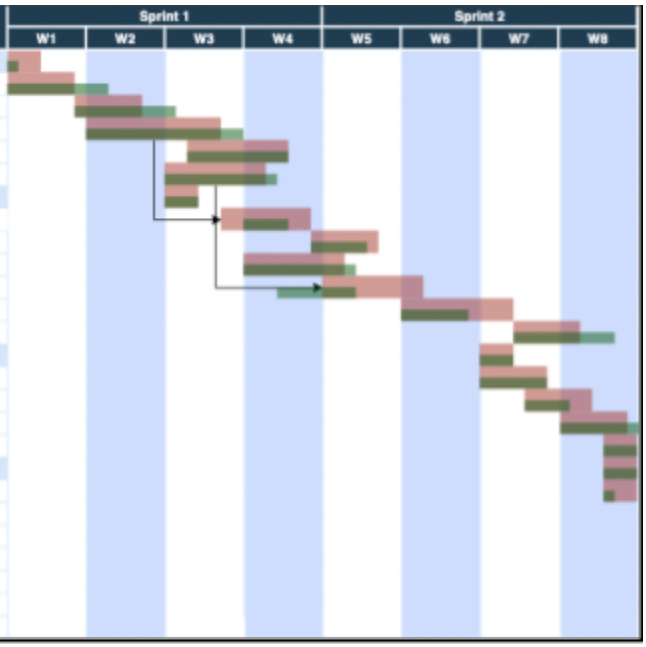

Fig. 5 Backlog

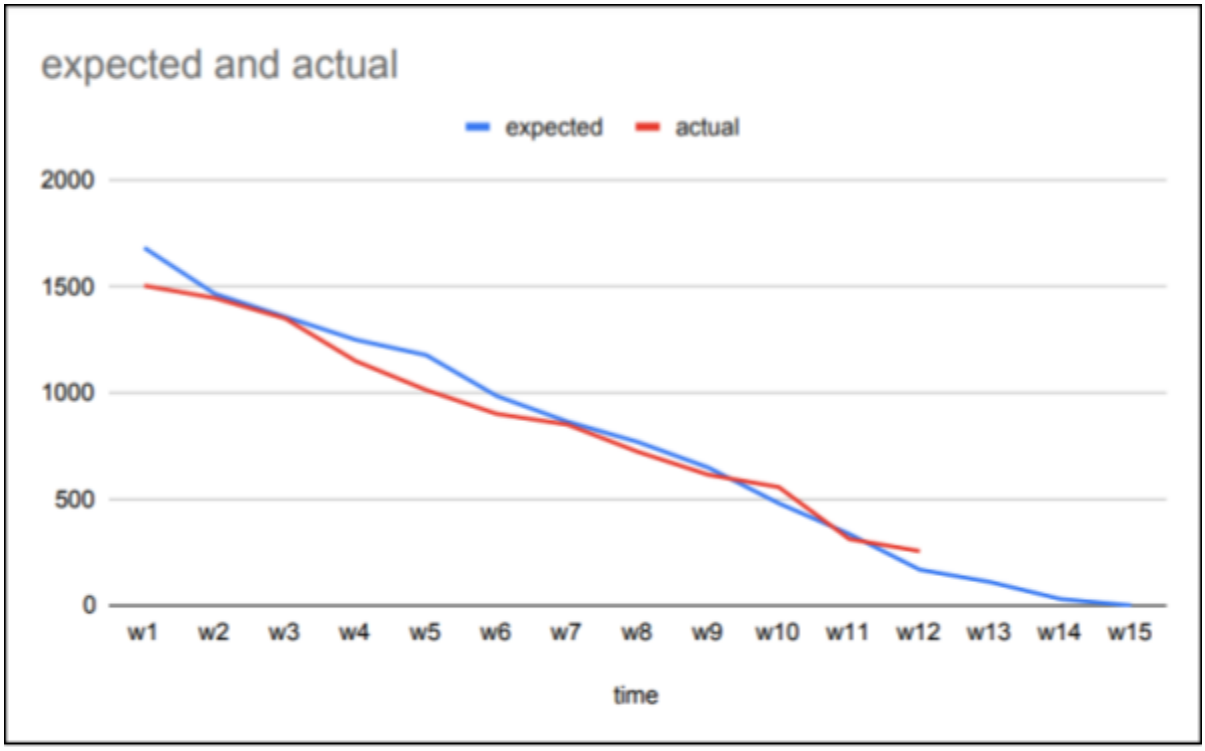

Fig. 6 Burn Down Chart

Kanban Screenshots

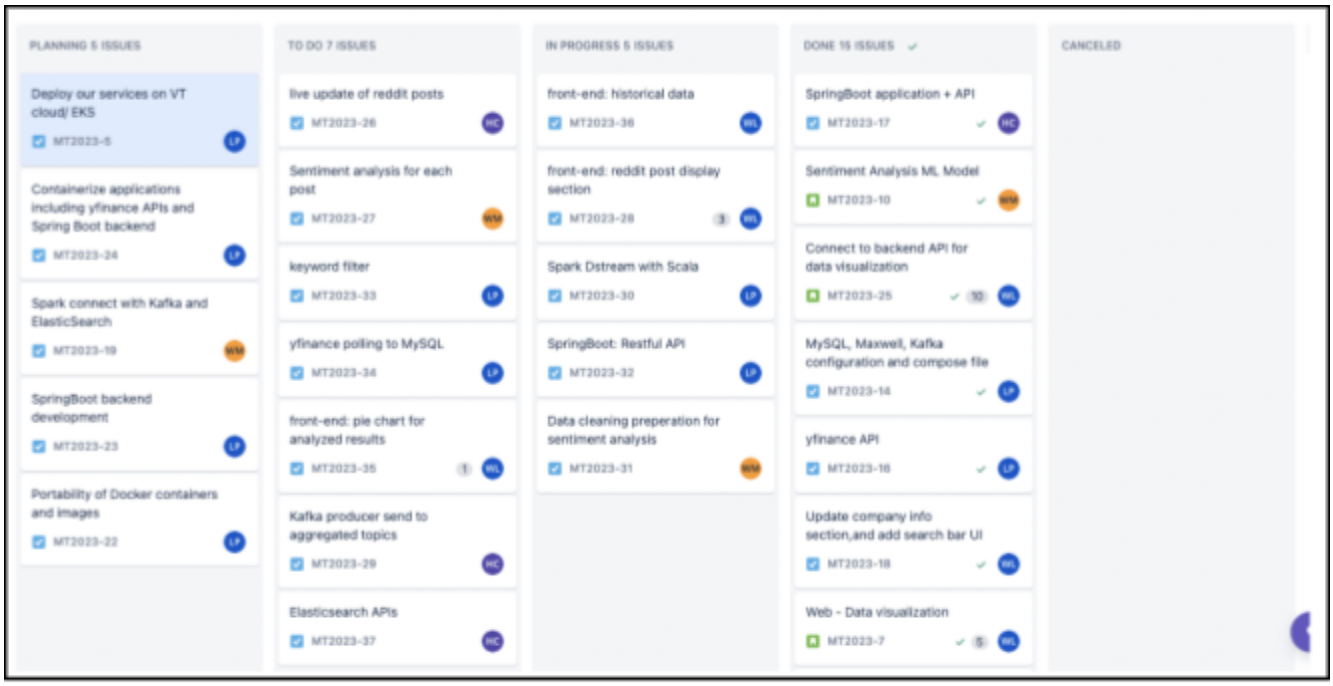

Fig. 7 The beginning of the sprint

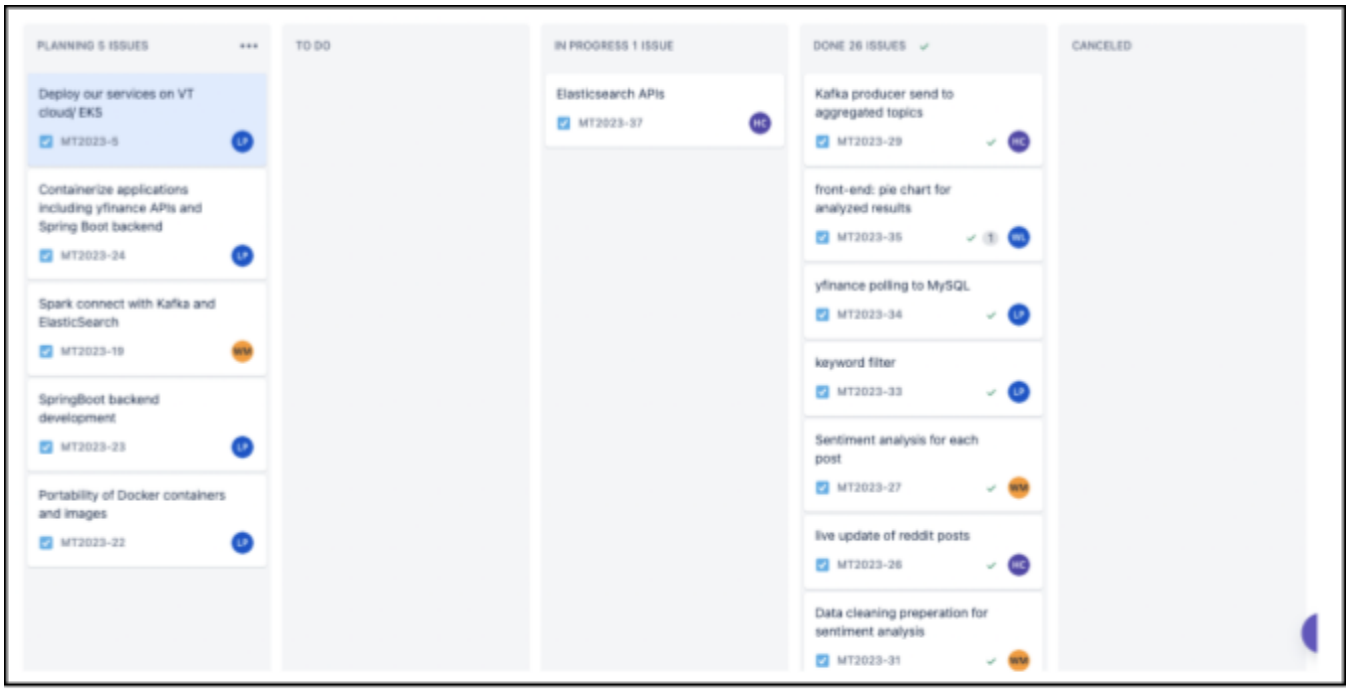

Fig. 8 The end of the sprint

Sprint retrospective

In this sprint, we have completed almost all the tasks as planned, except for the elasticsearch API, which requires additional time to learn the DSL language, so it takes more time than we expected.

Throughout the sprint, we learned the following three points:

1. When facing a problem, it's important to reach out to your teammates as soon as possible. By doing so, you can leverage the collective knowledge of the group and potentially find a better solution. For instance, determining the optimal storage option or project structure can be challenging, especially when new information emerges during the development process. Collaborating with others and sharing ideas can help overcome these obstacles. If you have an idea, you can also test it out and see how it performs before implementing it. Overall, teamwork and collaboration can lead to more effective problem-solving and better project outcomes.

2. At first, we spent much time discussing the UI, but we didn’t come to a conclusion. Before designing the UI, it would be better to start with what kind of project we want to do, what data we have, and what we expect our users to gain information from the website. Once we can clearly know what information we want to deliver, it would be easier to design the UI.

3. Regarding sentiment analysis on stock-related posts from Reddit, our team was unsure about how to present the results. We were faced with the challenge of determining how to weigh the importance of each post. For example, if a post expressed a positive opinion about a particular stock, but many other users disagreed, should we consider it a valid bullish opinion? It was difficult to determine the validity of a view. Ultimately, we decided to filter out posts based on upvote ratio. However, we believe that this topic warrants further exploration in the future.

Product backlog (updated)

Often, the first attempt at a user story does not contain enough detail to implement, or you identify additional user stories based on that original user story. The Sprint retrospective provides an opportunity to update your product backlog to account for these additions instructions. Review your user stories from the previous deliverable. List any new user stories you identified during the sprint, as well as any user stories you no longer plan to implement, and why you no longer plan to implement them.

We ultimately decided not to implement the pie chart feature for displaying analyzed results on the front end. We encountered some delays while setting up elasticSearch and working on keyword filters, and spent a significant amount of time writing the necessary API code. Due to these factors, we felt it was best to prioritize other aspects of the project and forgo the pie chart feature in this iteration.

Issue Tracking

Make sure to update your issues list in your GitLab project for the additions and changes to your user stories. Refer to the page on using GitLab for agile projects if you're unclear as to what this means. Simply putting an issue when is not a collaboration. I want to evidence you are using the tools to manage the project, not just make lists.

Following are how we manage issues of our project during the sprint. Fig. 9 and Fig. 10 show the open issue and closed issue respectively. Fig. 11 is the content of one of the solved issues. Root cause and solutions are attached for team members. For example the issue in Fig. 11 was solved by this PR. https://github.com/2023-VT-Spring-Capstone/spark-realtime/pull/15. All the PR related issue were written in the description.

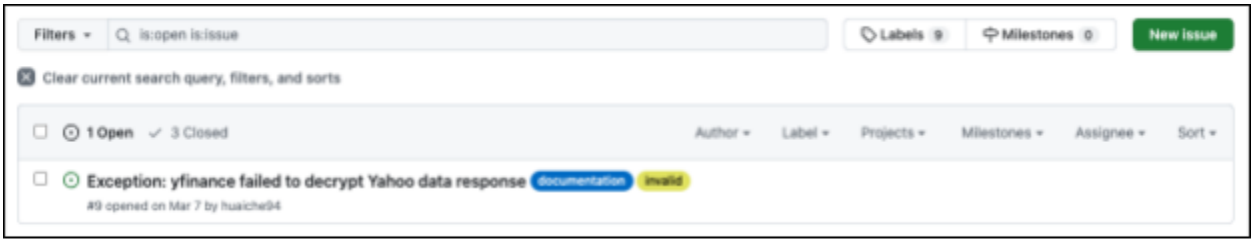

Fig. 9 Open Issues

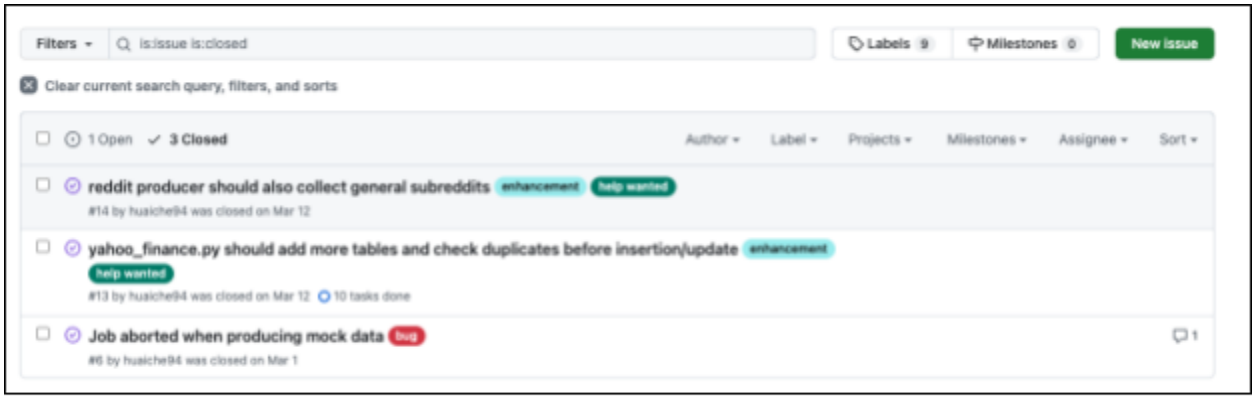

Fig. 10 Closed Issues

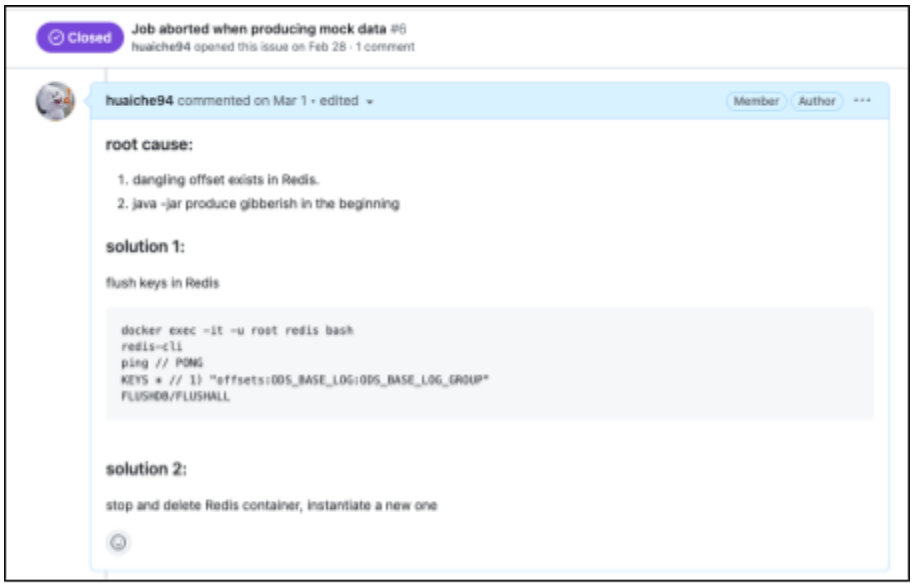

Fig. 11 root cause and solution

Sprint 3 Backlog

	Task Name	Ideal	Sprint 3			
			W9	W10	W11	W12
17	Priority-3-4 (Elasticsearch APIs)	48 hrs				
21	Priority-4-3 (Read/Write splitting with Maxwell)	48 hrs				
22	Priority-4-4 (Data cleaning, sentiment analysis)	96 hrs				
23	Priority-4-5 (Finalize the SpringBoot - Elasticsearch)	72 hrs				
24	Priority-4-6 (Finalize the API endpoints)	72 hrs				
25	Priority-4-7 (Finalize all the charts: line, pie)	72 hrs				
26	Buffer	112 hrs				

Fig. 12 Sprint 3 Backlog

Fig. 12 demonstrates the minor tasks in Sprint 3. We firstly developed the Elasticsearch API which translates from DSL to Java code for Elasticsearch High Level Rest Client. The minimum viable functionality is listing all the details of analyzed reddit posts. Read/ Write splitting with Maxwell is also finalized and we are finding a better way to transmit finance data to backend services. In this sprint we are exploring a BERT-based model for sentiment analysis and will look into more specific keywords for the Reddit ecosystem. We scheduled priority 4-7 as the last one because it is currently blocked by priority 3-4.

C. Documentation and code for completed tasks

For this capstone project, we use Github to manage our code and docs. Github Wiki page is used to put useful information and resources so members don't have to build the same wheel again. The API design document is subjected to changes from time to time for the frontend and backend development.

1. Sentiment analysis : https://github.com/2023-VT-Spring-Capstone/Sentiment-Analysis
2. capstone-front-end : https://github.com/2023-VT-Spring-Capstone/capstone-front-end
3. tooling : https://github.com/2023-VT-Spring-Capstone/tooling
4. spark-realtime (backend service) : https://github.com/2023-VT-Spring-Capstone/spark-realtime
5. Documents : https://github.com/2023-VT-Spring-Capstone/Documents
6. API design document: https://docs.google.com/document/d/1mWEAYTPcO65BwUhhgiWTj5d4KgeQlb1ihTFvoDbZ03I/edit?usp=sharing

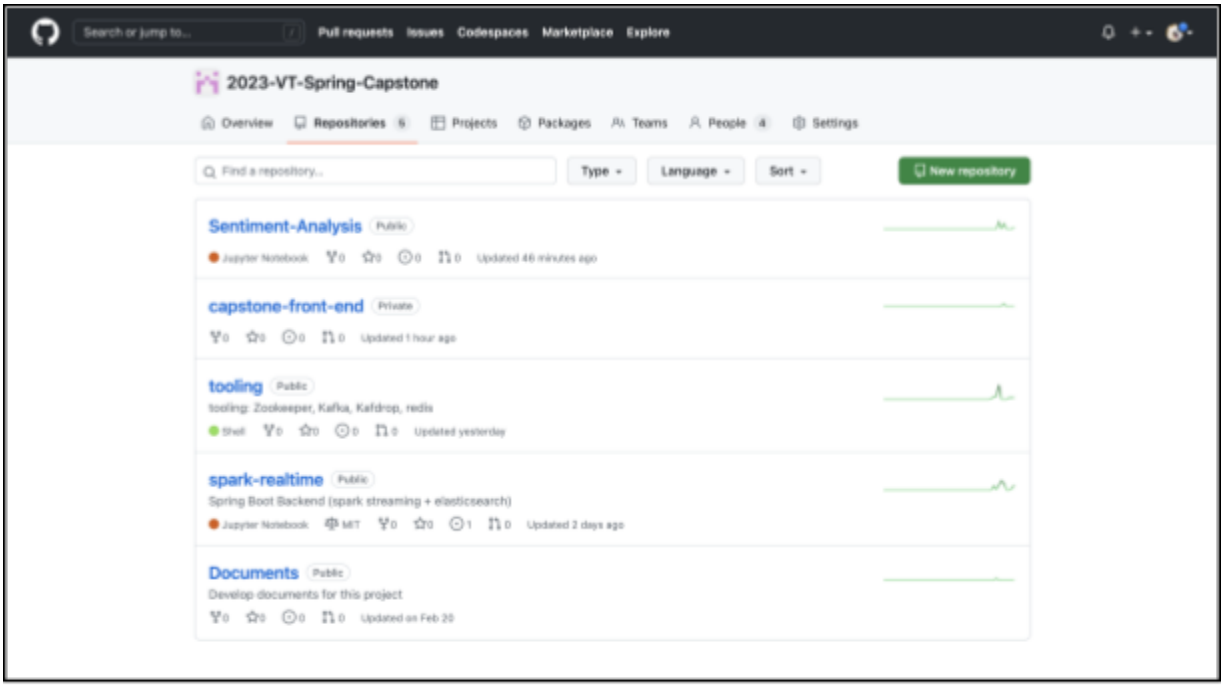

Fig. 12 Github repositories for capstone project

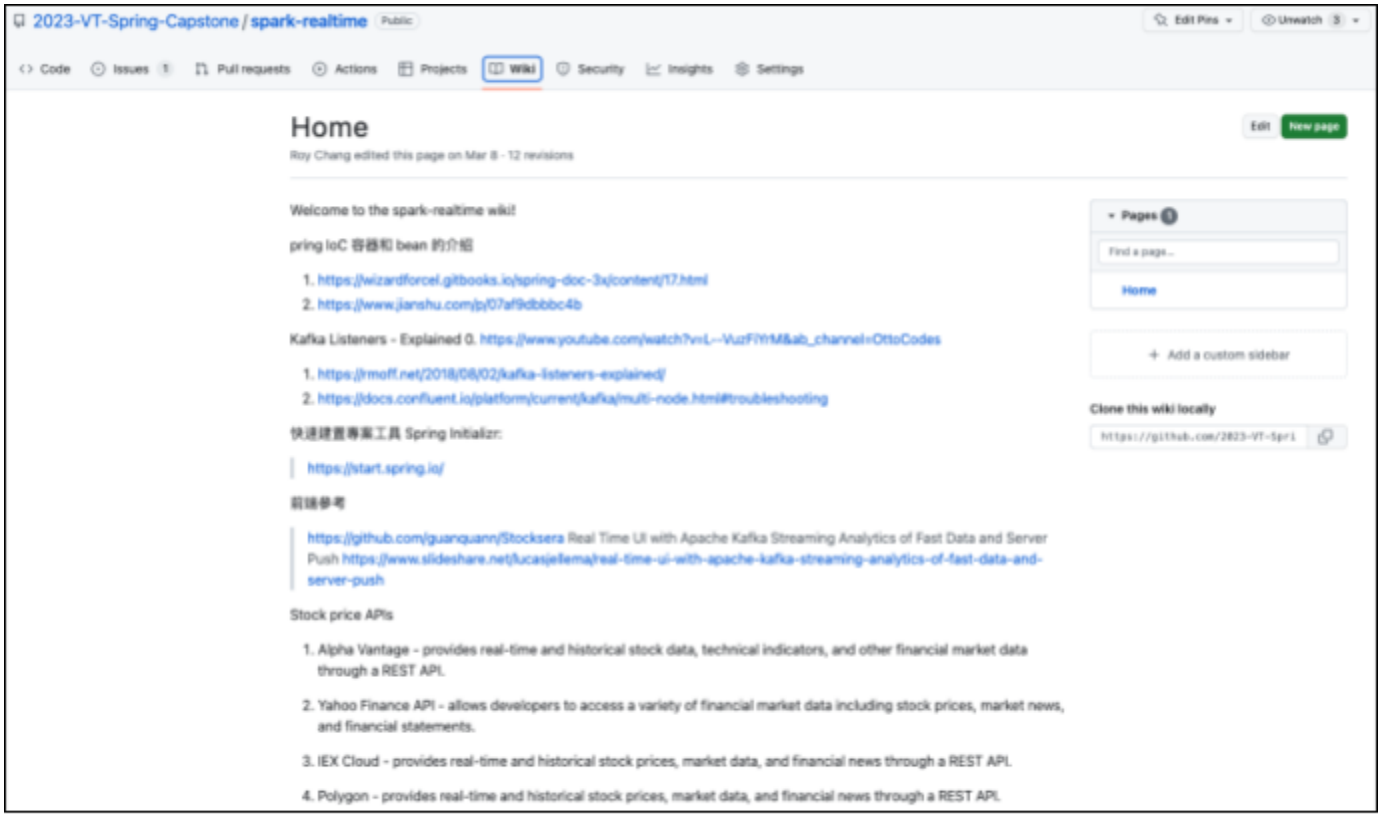

Fig. 13 Github repo Wiki