System Design Document (SSD)

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

0. Purpose of the document

This System Design Document (SDD) presents the technical details of the stock platform system design. Specific features and the motivation for this platform can be found in Purpose of the System and Proposed software architecture. This document starts with an introduction to the architecture and the design goals to be considered. Then it presents the proposed system architecture by describing the subsystem decomposition and the subsystem services. The hardware/software mapping is defined and the management of persistent data is explained. Access control and security issues are addressed. The global software control and boundary controls are described.

Version Control

Version Number	Name(title)	Author	Date	Authorization
1.0	SDD		02/26/2023	

Document Properties

Item	Details
Document Title	System Design Document (SSD) - Real-time monitoring of volatile, retail investor-driven price fluctuations in the equity market
Author	Huai-Che Chang Wen-Fang Lu Wan-Yi Mao Layne Pu
Creation Date	02/19/2023
Last Updated	02/25/2023

1. Introduction

This project would be a user-friendly and scalable platform that integrates with stock information from social media. It provides users insight of stock markets by sentiment analysis and interactive graphs and charts. The detailed design goals and software architecture are illustrated in the following sections.

1.1 Purpose of the System

Stock market is the aggregation of buyers and sellers of stocks, which represent ownership claims on businesses. Stocks go up and down everyday because of the law of supply and demand. With the rise of social media, the influence of social media on financial markets is here to stay, as younger generations start saving and investing. This carries both opportunities and risks. Information sharing and discussion on internet platforms can improve market transparency and efficiency. On the other hand, social media platforms are known vehicles of disinformation and manipulation of human behavior.

In order to help the investors observe the stock market , we would like to build a real-time and user-friendly stock platform. In this platform, people could have access to the newest and valuable information retrieved from social media. In the end, people could use this platform as a means to explore the stock market and make their own decisions to invest the money on stocks.

1.2 Design goals

There are three goals we would like to achieve in this project :

- A seamless interface for stock price and reddit
    - Users can access interactive charts and graphs, and customize their dashboards.
- Customized data exploration:
    - In this platform, we provide the deep insight of the stock information such as sentiment analysis and stock prices. This feature can help users to explore the stock market from diverse perspectives. Selected date range and keywords can be filters for which data should be presented.
- Fault tolerance and high throughput
    - A single topic log is partitioned into multiple logs, each of them can live on a separate node in the Kafka cluster. In this way the processing of existing messages can be split among brokers and increase the throughput. Redis can help with data caching for higher QPS.

1.3 Definitions, acronyms, and abbreviations

- Fact data
    - Raw, data, or data to be analyzed.

- Dimension data
    - Descriptive information about data, or filtered data.

- QPS
    - The number of queries or requests per second. If you have more QPS capacity when making API calls, the API performance will be higher and more stable.

- Exactly-once :
    - A data processing guarantee which ensures that each message is processed only once and in the right order. With exactly-once semantics, Kafka guarantees that messages are not lost or duplicated during processing, even in the presence of failures.

- Bootstrap Server:
    - In this project the bootstrap server is used interchangeably with the kafka cluster with 3 brokers.

- Spark streaming :
    - A micro-batch processing model where streaming data is divided into small batches and processed using Spark's batch processing engine. This allows our project to achieve high throughput and low-latency.

- Bullish
    - In the context of finance and investments, the term bullish refers to a positive or optimistic outlook on the future performance of a particular asset or market.
    - A bullish sentiment can lead to increased demand for an asset, which in turn can drive up its price. However it does not guarantee that a stock price will rise in value, and investors should always consider the risks associated with any investment decision.

- Elasticsearch indexes
    - In Elasticsearch, an index is a logical namespace that maps to one or more physical shards (i.e., a subset of the data) that hold the indexed data. Elasticsearch indexes are used to organize and store large amounts of data, making it easy to search and retrieve specific pieces of information.

- Full-text queries
    - A full-text query is a type of search query that looks for words or phrases within the content of a document or a database. It allows users to search for specific words or phrases within a larger body of text, rather than just searching for a specific term or keyword.

1.4 References

There are many stock platforms that provide services for trading and information about stocks. However, the influence of social media on the stock market is undeniable. In many ways, it's a double-edged sword: while social media can promote stocks and make them go viral, it can also lead to their downfall when investors start shorting them or betting against them. As a result, we want to provide a stock platform that integrates with the information from social media. Following are the platforms and models that have the features we want to include in this platform, providing investors with comprehensive and up-to-date information.

- HootSuite Insights
  Hootsuite Insights is a social media monitoring platform that allows companies to track and analyze motions of their brand and products on social media. It can also be used to track mentions of stocks and monitor sentiment around specific companies or industries.

- Yahoo Finance
  Yahoo Finance is a web-based platform that provides users with financial news, real-time stock quotes, stock charts and research tools.

- Sentiment Analysis Pre-trained Models
  Pre-trained sentiment analysis models offer a resource-efficient solution to analyzing new text data without training a model from scratch. This is particularly useful in domains where there is limited labeled data available. Popular pre-trained models include BERT, RoBERTa, and DistilBERT, which have been trained on large datasets and provide state-of-the-art performance on various sentiment analysis tasks.

1.5 Overview

There are several stock platforms currently available, including TradingView, Investing.com, and Yahoo Finance. These platforms offer news and financial reports that show the information of the stock market. However, social media platforms are also playing a significant role in showing investors reaction as well as shaping investor’s decisions.

Our platform aims to provide more information about a particular stock by analyzing public views of its potential. While this approach has both pros and cons, it can help investors to quickly understand how other people feel about a stock without having to spend a lot of time collecting information. One of the benefits of using social media to analyze stock potential is that it provides a broader perspective. By collecting a large enough dataset, the information can be subjected to statistical analysis. This approach can provide valuable insights into public sentiment towards a stock, which can be used to guide investment decisions. Social media information cannot always be accurate or objective. It is mostly subjective and can be influenced by factors beyond the stock's fundamental performance. Investors must also be cautious and exercise critical thinking when using social media as a source of information.

2. Current software architecture

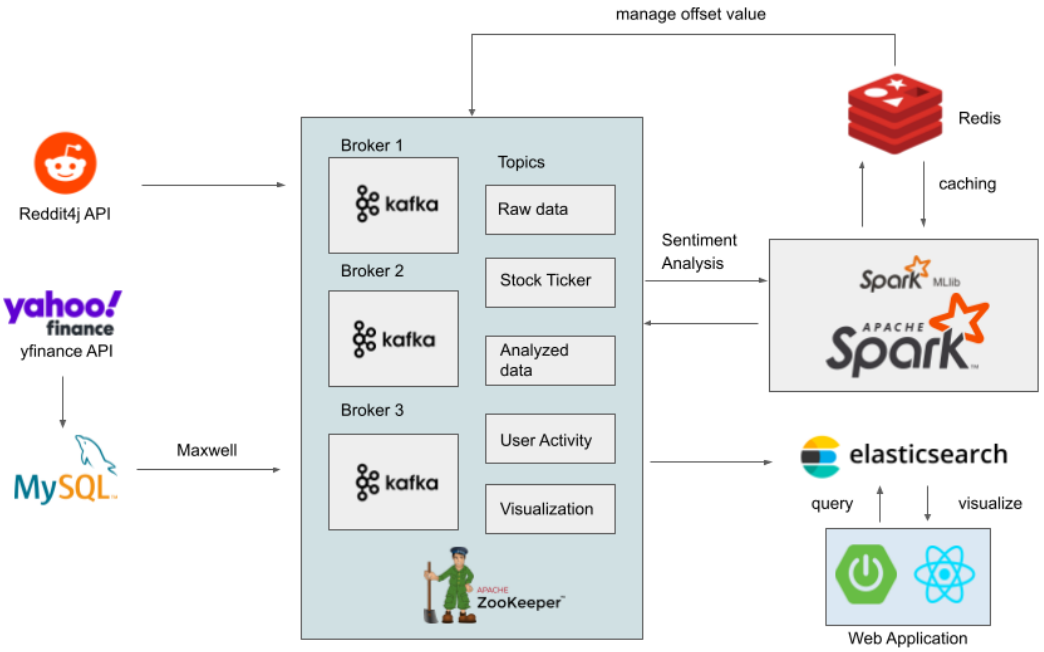
Systems Overview

There are various stock platforms available, including Yahoo Finance and StockTwits, that use different software architecture to provide financial data and insights to users. Yahoo Finance's software architecture includes a front-end web application, data ingestion and storage system, data processing and analytics tools, search and indexing capabilities, and data visualization features. In contrast, StockTwits also includes sentiment analysis on social media data in addition to a similar system structure.

Our aim is to integrate the real-time streaming and sentiment analysis features of platforms like StockTwits with the system structure of platforms like Yahoo Finance. This would involve designing a software architecture that can efficiently ingest, store, process, and analyze large volumes of financial and social media data in real-time, and provide insights to users through an intuitive front-end interface. The architecture would need to be scalable, reliable, and able to handle diverse data sources, including social media platforms like Reddit, in order to provide users with valuable insights on stock market trends.

3. Proposed software architecture

3.1 Overview



3.2 Subsystem decomposition

Reddit: Data source for the dynamic, real-time data

Yahoo finance: Data source for static type of data

MySQL: store the data fetched from yahoo finance,

Redis: manage Kafka offsets, cache dimension data

Zookeeper: manage 3 Kafka brokers. A topic will be cut in 4 partitions and distributed.

Kafka: A distributed stream-processing platform for better throughput of real-time data.

Topics: data are separate into different topics for producer/consumer. This design decouples the entangled processes of data from data source to data sink.

Spark: A Data processing engine. Spark MLlib is used for sentiment analysis.

Elasticsearch: provides text search and that enables users to filter out data with keywords.

Web application: handles incoming HTTP requests from frontend and visualizes data on webpage.

3.3 Hardware/software mapping

	Software	**Hardware/ Platform**
Development/ Testing	Docker/ Docker compose: with unified yaml file we can develop and test on local machine with the same environment Kafdrop: In early phases we use this open source project to monitor the Kafka clusters. Grafana: In later stages we plan to use this tool to monitor the throughput of each broker.	Mac OS ARM 64 architecture
Deployment	Strimzi solution for Kafka on Kubernetes Terraform to automate infrastructure tasks	VT CS Cloud AWS EKS

3.4 Persistent data management

Our project uses a MySQL database as the persistent storage for handling large volumes of data. MySQL is a proven database management system that provides reliable, persistent storage capabilities, and it can efficiently scale horizontally by adding more servers if needed. This means that MySQL is well-suited for our data streaming project that requires handling very large amounts of data. Additionally, MySQL is relatively easy to set up and configure, which enables us to deal with real-time data that requires quick and efficient access to the data being stored.

The persistent data storage contains the following types of data:

1. Static data

   This type of data refers to historical stock pricing data collected from the yfinance API. It provides information on the price and trading volume of a particular stock over a period of time in the past, including the opening, high, low, and closing prices, as well as the volume of shares traded during each time interval such as minute, hour, day, or week.

   Data schema:
   {
   Symbol: String
   Open: float
   High: float
   Low: float
   Close: float
   Date: date
   Volume: integer
   }

   Explanations:
   - Symbol: a unique series of letters assigned to publicly traded companies that are listed on a stock exchange
   - Open: the price at which a particular stock starts trading at the beginning of a trading day
   - High: the highest price that a stock has reached during a trading day
   - Low: the lowest price that a stock has reached during a trading day
   - Close: the final price at which a stock is traded at the end of a trading day
   - Date: the specific date on which a particular transaction takes place in the stock market
   - Volume: the total number of shares of a particular stock that are traded during a trading day

2. Dynamic data

   This type of data refers to the Reddit content data collected from Reddit API, including posts and comments from specific subreddits, user profiles and activity,

subreddit metadata, such as subscribers, moderators, and tags, and finally the time series data on the number of posts and comments over time.

Sources:

- r/Investing: This subreddit is focused on investment-related discussions and analysis.
- r/StockMarket: This subreddit is dedicated to discussions about the stock market, investing, and trading.
- r/Daytrading: This subreddit is for discussions about day trading and short-term trading strategies.
- r/WallStreetBets: This subreddit gained popularity in recent years due to its focus on high-risk, high-reward trades and investments.
- r/Stocks: This subreddit is a general forum for stock market discussions, news, and analysis.
- r/Options: This subreddit is focused on options trading and strategies.
- r/PennyStocks: This subreddit is dedicated to discussions about penny stocks, which are stocks that trade for less than $5 per share.
- r/Robinhood: This subreddit is focused on discussions about the Robinhood trading platform and related topics.
- r/SecurityAnalysis: This subreddit is dedicated to discussions about security analysis and valuation.

Data schema:

```
{
        id: string
        author: string
        created_utc: integer
        body: string
        score: integer
        permalink: string
        title: string
        num_comments: integer
}
```

Explanation:

- id: A unique identifier for the post or comment on Reddit
- author: The username of the user who submitted the post or comment
- created_utc: The UTC timestamp of when the post or comment was created
- body: The text of the post or comment
- score: The net score (upvotes minus downvotes) of the post or comment
- permalink: A URL that links directly to the post or comment on Reddit
- title: The title of the post, if applicable (only applicable to posts, not comments)
- num_comments: The number of comments on the post or comment, if applicable (only applicable to posts, not comments)

3.5 Access control and security

Due to the constraints of project scale and development time, Access Control and Security are currently not within the scope of our development considerations. However, there are several topics worth considering for future versions:

1. Implementing a login system to verify and authenticate users, ensuring that only authorized users can access the system.
2. Creating different user roles with varying permissions to provide different levels of access to website functions.
3. Encrypting data to protect against unauthorized access or data breaches would be a valuable addition to the system.

By addressing these topics in future versions, we can significantly improve access control and security in our project. Such improvements can enhance network security, offer greater control over data access, and safeguard against potential security threats.

3.6 Global software control

For the external flow control: Dynamic type of data is fetched through polling and dispatched to brokers. The static type data is fetched by event-driven callbacks and saved into database services. Services communicate with each other through HTTP/1.X protocol. If time permits, gRPC with HTTP/2 is also considered. For the internal flow control: each service has its own thread for communication, which communicates with the service manager and other services.

3.7 Boundary conditions

The following steps outline the procedure to start the system successfully:

1. Start the zookeeper server first
2. Start the Kafka server
3. Start the Maxwell server that connects to Kafka and MySQL
4. Start the Elasticsearch server
5. Run the core streaming application
6. Run the applications that scrape data from the Reddit API and finance API
7. Launch the web application, enabling users to access data from the data stream line.

It's important to follow this sequence when starting the system to ensure that all services are initialized correctly and that the system operates seamlessly. If you need to shut down the whole system, simply close each component in the inverted sequence, starting from step 6 and moving down to step 1. This will help ensure that the system is shut down properly and that no data or processes are lost.

4. Subsystem services

Data collection

- Reddit API: retrieves data from Reddit.
  https://github.com/masecla22/Reddit4J
  https://www.reddit.com/dev/api/
  https://praw.readthedocs.io/en/stable/

- yfinance API: retrieves data from Yahoo finance.
  https://github.com/ranaroussi/yfinance

MySQL Database storage

- Stores stock price data from Yahoo finance.

Redis Database

- Manages the offset value of kafka to ensure no data loses in the presence of failures.
- Cache the dimension data for query optimization.
- The offset data in Kafka is in key to value structure, where key is group ID, topic, and partition, and the value is offset. To map this data into Redis, we need to use hset API and further map group id + topic to the key of structure, and map partition + offset to the value.

Data streaming and processing

- Maxwell: listens for changes to the MySQL database and forwards them to Kafka.
- Spark: splits the stream into sub-streams and inject into different topics
- Kafka: 3 brokers in a single cluster, each topic is partitioned into 3~4 sections with 2~3 replicas
  - Producer: sends data to the bootstrap server (Kafka cluster)
  - Consumer: subscribes to Kafka topics and consumes the data for later processing or analysis.

Spark MLlib

- Analyzes the content of the stock posts from Reddit and calculate the sentiment scores of bullish.

Full-text queries

- Elasticsearch indexes the text data to enable fast searching and querying, which in turn enables the web server to efficiently retrieve and serve text data to the user.

Web server

- Handles incoming HTTP requests from the frontend and responds with the corresponding data using Java Spring boot.
- Interacts with Elasticsearch to search and query the indexed data.

Web client (user interface)

- Serves the frontend application to the user's browser.
- Visualize the data in a user-friendly way and allow the user to interact with the system.

- Sends requests to the backend API server and handles the responses to update the user interface.

A little wrap up:

The back-end application

This platform has two data pipelines: real-time data streaming and static data. Real-time data is obtained through Reddit APIs and processed using Apache Kafka and Spark for sentiment analysis. On the other hand, the static data is retrieved from the finance API and stored in a MySQL database. The historical data is updated daily in the background.

1. Sentiment Analysis for real-time reddit posts
2. Full-text search
3. Historical stock data

The front-end application

The application would be a dashboard that provides data visualization and searching functions in stock. Users could interact with the stock graphs, searching and see the real-time sentiment analysis results of the posts from Reddit. Moreover, the searching function allows users to explore the stock they are interested in.