IE498 HFT FINAL REPORT - GROUP2

Title

A web application to visualize historical lookup and streaming data in financial market.

Team:

Jundong (Ted) Chen - Leader: tedjdbusiness@gmail.com; Qiyuan Cheng: qiyuanc3@illinois.edu; Sparshdeep Singh: ss85@illinois.edu; Fang (Kevin) Li: fangli3@illinois.edu;

Project Description

This document is the final report of a semester long project for "IE498 - High Frequency Trading" under Professor David Lariviere.

The web application utilizes `Django Rest API Framework` and `Cors Header` to connect the `React` frontend and the `Django` backend. Upon frontend, we use `antd` library for UI implementation and we use Django default `Sqlite` database to store historical lookup data from external API in backend.

There are three major external API in this application: `Alpha Vantage` (Historical Lookup Data), `Polygon` (Streaming Options Quote and Trade Data), and `Investpy` (Economic Calendar Data). Users will be able to visualize and interact with both historical lookup data and streaming options data in frontend.

The historical lookup data includes:

Macro Economic Data:

- Economic Calendar
- Real GDP Quarterly/Annual
- Real GDP per Capita
- Daily/Weekly/Monthly Treasury Yield 3M/2Y/5Y/7Y/10Y/30Y
- Daily/Weekly/Monthly Federal Funds Rate
- Monthly/Semiannual CPI
- Inflation
- Inflation Expectation
- Consumer Sentiment
- Retail Sales
- Durables
- Unemployment
- Nonfarm Payroll

Company Fundamental Data:

- Quarterly/Annual Income Statement
- Quarterly/Annual Balance Sheet
- Quarterly/Annual Cash Flow
- Earning History
- Company Overview
- Related News
- Earning Calendar
- Options Chain

The streaming data includes:

Options Data:

- Options Quote

- Options Trade

We maintain the connections among frontend, backend, database, and external API. For historical data that is constantly unchanged, users will query the database before making external API calls to store new instances to database. For historical data, such as economic calendar and options chain, that are constantly changed, users will make directly API calls in frontend. For streaming options trade and quote data, we establish websocket connection to Polygon API so that we can get updates precisely to nanoseconds timestamps. Users will be able to select and interact with data.

Technologies

API

- Polygon for news, options chain, real-time options quote, and real-time options trade data (Premium subscription required)
- AlphaVantage for lookup data (Academic data access required)
- Investpy for economic calendar data (No subscription required)
- Future developers can substitute their API keys in the backend files titled as the name of the API.

Included

- React Frontend Framework:
  - A prevalent frontend framework with strong support libraries for UI components
- NodeJS:
  - Languages used in React frontend framework
- Django Backend Framework:
  - A powerful open-source framework for backend web applications
- Python
  - Languages used in Django backend framework
- Django Rest API Framework:
  - Powerful toolkit for building Web REST APIs
- Cors Header:
  - Cross-Origin Resource Sharing for connection between front and backend
- Antd:
  - React UI Library
- Django Sqlite Database:
  - Relational Database to store constantly unchanged historical lookup data
- Websocket:
  - Two-way interactive communication session between user's browser and server to receive real-time data update.
- HTTP requests:
  - Messages sent by the client to initiate an action on the server
- Serializer:
  - Converting objects into data types understandable by javascrpt and frontend.
- React Router Dom
  - Dynamic routing in a web application
- Vagrant VM:
  - Management to create, configure, and manage boxes of virtual machines
- Alpha Vantage API
  - Good documentation
- Polygon API:
  - Low latency streaming data precisely to nanosecond timestamps
  - Good documentation
- Investpy API
  - Good documentation

Excluded DTN Iqfeed API: - Unsuitable for web application - High Latency for streaming data - Poor documentation for implementation

Components

Application Hierarchy: enter image description here

Application Components: enter image description here

Git Repo Layout

`frontend` contains all the files for frontend. `others` contains all the non-coding files during projects. `web-dev` contains all the files for backend. enter image description here

Navigate to `../frontend`. `public` contains all public components like icons. `src` contains all important files for frontend. enter image description here

Navigate to `../frontend/src`. `components` contains the small components used in other pages. `pages` contains the design of homepage. `visualization` contains the design of other pages and functionality implementation. `App` contains the router setup. enter image description here

Navigate to `../web-dev`. `backend` contains django application files. `lookup` contains django project files for a project named "lookup". **Note: An application can have multiple projects.** enter image description here

Instructions for using our project

This project may not work properly in the future because it relies heavily on external APIs. Some API keys are only limited for individual use and will expire in the future.

VM uses vagrant and VirtualBox. The VM is set to use 1024mb memory and 1 cpu.

- Install Vagrant (2.2.19) and VirtualBox (6.1.34)
- In shell, navigate to project directory
  - (ex. cd C:\Users[UserName]\Documents\GitHub\group_02_project
- Run `vagrant init generic/centos7`
- Run `vagrant up`
- To connect to VM, run `vagrant ssh`
- To logout of VM, run `logout`
- To halt VM, run `vagrant halt`
- To destroy VM, run `vagrant destroy"`

In case the VM is broken, the following is the environment we use for development.

- Python 3.10.4
- Node.js 16.15.0 LTS
- npm 8.6.0
- pip 22.0.4
- Django 4.0.4
- djangorestframework 3.13.1
- markdown 3.3.7
- django-cors-headers 3.12.0
- investpy 1.0.8
- datetime 4.4
- requests 2.27.1

You will have to start both the backend server and the frontend server. Navigate to `web-dev/backend`, run `python manage.py runserver` to start backend Navigate to `frontend`, run `npm install` and `npm start` to start frontend The localhost for backend server is `http://localhost:8000/` and you can modify the url to check different data endpoints according to the http messages. The localhost for frontend server is `http://localhost:3000/` and you can visualize and interact with data by clicking on different functionalities in sidebar.

The detailed instruction of how to use our project can be found in this demo video. https://gitlab.engr.illinois.edu/ie598_high_frequency_trading_spring_2022/ie498_hft_spring_2022_group_02/group_02_project/-/blob/main/group2project%20demo.mp4

Testing

Polygon requires advanced options subscription which gives access to streaming options trade and quote data. You can substitute your API key to the existing one in `web-dev/backend/lookup/polygonApi.py`.

Alpha Vantage requires different API key. You can substitute your API key to the existing one in `web-dev/backend/lookup/alphaVantageApi.py`.

Other API does not require individual API key.

All the API keys in this project are restricted to only individual use and they are imposed connection restrictions by API servers, which cannot support deployment testing.

Project Result

This project provides us a chance to participate in the development process of a full-stack web application. There are many technical issues we encountered when trying to develop the app from scratch. It's a very different process between theories and practices. We all know what a relational database looks like but none of us has real experience on how to work with it. We all know what websocket is from lecture but none of us has real experience on implementing it. It's a completely different experience between knowing it and practicing it.

This is a well-functional web application with many functionalities available from visualizing historical lookup data to handling streaming data. We're very impressed and happy to see the final results exceed our initial expectation given the little web development background we had at the beginning of this project. We do not choose to skip the difficulties. Instead, we embrace them and try to include as many different functionalities as possible. However, given the expenditure to include other streaming data in our API subscription plan, we only select options data. But once we figure out how to handle the streaming options quote and trade data, the same techniques can be applied to other streaming data such as stocks, cryptos, and forex.

Overall, the final results exceed our initial expectation of the project.

Screenshots and Example Outputs

Access `http://localhost:8000/lookup/` for all backend endpoints enter image description here

Economic Calendar Table enter image description here

Chart Visualization enter image description here

Chart Interaction enter image description here

Chart Interaction enter image description here

Search the symbol enter image description here

Earning History enter image description here

Company Overview Table enter image description here

Related News with Clickable URLs enter image description here

Earning Calendar for all companies and a specific symbol enter image description here

Options Chain enter image description here

Streaming Quote Data enter image description here

Streaming Trade Data enter image description here

Postmortem Summary

Jundong (Ted) Chen: *1. What did you specifically do individually for this project?*

- Almost everything as I've spent several hours writting this final report and recording demo. I set up the connections among frontend, backend, database, and APIs at the beginning of the project. I designed all functionalities and researched all new APIs that may be suitable for this project. I wrote entire backend and 1/3 frontend (`News`, `EconomicCalendarList`, `EarningCalendar`, `OptionsChain`, `OptionsQuote`). I tested the final application version and fixed all the issues I encountered. As team leader, I assigned tasks to everyone and make sure that the tasks are managable for each individual. I maintained the communication in discord chat, frequently updated progress, ask for others' feedbacks to improve quality, and put pressure to ensure some teammates who are behind to be in the right track.

2. What did you learn as a result of doing your project?

- Before this project, I only had very limited background in developing a full-stack web application. By doing this project, I learned all the technical details used in this project and I would even be able to replicate a new one myself. I also learned the cooperation and the skills to manage the team as a team manager,

3. If you were to continue working on this project, what would you continue to do to improve it, how, and why?

- This project is mainly for learning purposes. I try to cover as many functionalities as possible without considering if the functionalities are truly useful. Even some functionalities may not be that useful, I still learn a great deal during this process. In the future, this web application will be more user-centric and the functionalities will be re-designed to fit the real user needs. The visualization will also be improved to look more like a commercial web application.

4. What advice do you offer to future students taking this course and working on their semester long project (be sides "start earlier"... everyone ALWAYS says that). Providing detailed thoughtful advice to future students will be weighed heavily in evaluating your responses.

- The project should be from small to big. It should start from a small idea and scale up to become a big one. For example, we want to build a web application for streaming stock data visualization. After we, as a team, achieve the goal, we can start to incorporate the streaming options data, streaming forex data, and etc. If we still have more time left, we can further improve the project by implementing better GUI and user interaction. Because individuals and teams have different learning curve and progress speed, the project should start from small to big to adapt different situations.
- Providing feedbacks, whether suggestions, agreement, or appreciation, to other teammates in a timely manner is very important to maintain good communication and cooperation. Do not let only one person talk in the chat alone.

Qiyuan Cheng: *1. What did you specifically do individually for this project?*

- I was responsible for the layout of the entire front-end, used Table and Description component in ant design and Echart data visualization tools to achieve the visualization of the data stored in database and streaming data. Implemented the search function for different company data. Also implemented the user login, logout and register functions on the front end.

2. What did you learn as a result of doing your project?

- Learned Html, CSS, Javascript from scratch. Learned how to use third party library to make a clear and concise front-end layout. Learned how to use react framework, how to use its core concept to achieve different functionalities. Learned how to use Echart library to achieve data visualization. Learned how to handle the data passed from the backend to the frontend and how to make them appear on the frontend according to different needs. Learned some basic concepts about high frequency trading.

3. If you had a time machine and could go back to the beginning, what would you have done differently?

- Nothing different.

4. If you were to continue working on this project, what would you continue to do to improve it, how, and why?

- Make the front end more beautiful, and add functionalities according to the needs of the back-end.

5. What advice do you offer to future students taking this course and working on their semester long project (be sides "start earlier"... everyone ALWAYS says that). Providing detailed thoughtful advice to future students will be weighed heavily in evaluating your responses.

- Communicate with teammates in a timely manner and give feedback on problems and difficulties.

Sparshdeep Singh: *1. What did you specifically do individually for this project?*

- I worked primarily on the backend handling tasks assigned by our project leader. One of my major tasks was to implement the datafeed. We researched different APIs and settled on polygon API. I created the python files and methods to access and manipulate the datafeed for news and options contracts data. Then I worked with Ted to integrate the datafeed into the Django framework to be able to lookup using endpoints.
- I also worked on hosting the project on Vagrant and VirtualBox.
- Finally, I integrated an economic calendar in the backend and frontend with data on important economic events and their forecasted and actual results. I did this using investpy's API and getting data from their calendar.

2. What did you learn as a result of doing your project?

- I learned a great deal from our project. I had never used a market API to get data before, so the great amount of data we were able to get shocked me. Manipulating that data and visualizing it on the frontend was also new to me. I had never worked on backend for this type of project before, so I gained a new perspective implementing the backend and then using the data on the frontend. Also, this was my first time using the Django framework. It is very clean and I can see myself using it in the future to set up other projects.

3. If you had a time machine and could go back to the beginning, what would you have done Differently?

- I would create a more detailed outline of the deliverables we want for the semester. I feel like we lost track of what we wanted and began implementing as many features as we could get our hands on. This isn't necessarily a bad thing since in the end we have a project that is way more impressive than we had originally planned, but setting an end goal is important to a healthy team environment.

4. If you were to continue working on this project, what would you continue to do to improve it, how, and why?

- I would begin adding user friendly visualizations. Right now we are just displaying data and basic manipulations of it. Adding graphs and technical indicators of the data and maybe even our recommendations would be a big boost to our projects frontend appeal, but we were not able to implement such features in time. However, what we have is a good foundation for a better app, so given more time, I believe we can make this project even more impressive and professional than it already is.

5. What advice do you offer to future students taking this course and working on their semester long project (be sides "start earlier"... everyone ALWAYS says that). Providing detailed thoughtful advice to future students will be weighed heavily in evaluating your responses.

- I would recommend setting clear expectations for what the project should look like by the due date. Planning is as important, if not more, than implementing the project itself. If you have no direction, development is going to be messy and there can be miscommunication between team members, further delaying the timeline.
- Weekly meetings are a must. Checking in on everyone's progress and whether they need help or not is one of the major obstacles I believe groups face. Getting blocked is okay, but you need to be able to reach out and ask for help so you can solve the problem efficiently. This leads into communication again, but it's important to keep communication clear and effective, so everyone is on the same page and you can root out issues before they begin delaying the timeline and project's success.
- Research. There are many resources available online that can streamline your project's deliverables. Spending a whole week developing a script for a task just to find out later the same functionality could have been accomplished using a library is demoralizing. Before beginning development, make sure to research all avenues of the project to ensure you are using all of your available resources effectively, giving you more time to work on personalizing the project to your liking.

Fang Li: At the first half part of this project. I was assigned to scrape the data containing symbols, price and so on from Nasdaq and yahoo finance for the establishment of the project. Because we choose to use the stable data firstly as the sample for the establishment. Then I was assigned to find out how to get the real time data. Ted also worked on it, and he chose to use a new one and tell me to test all the api to see if they are useful. He then let me get the sample data from each api to see the result after using each api and go on to find which columns of data can be transformed into diagrams meaningfully.

At the last half part of this project. I was assigned to work with Qiyuan to implement the data visualization and the establishment of the frontend. Because Qiyuan is expert at frontend by JavaScript, but I do not know how to code in JavaScript, so I can only do the 'data part' and go on to help him to match each api of different functionalities with the corresponding data. I think in this part, Qiyuan did more than me. In the end of this semester, Ted assigned me to do the deployment test and the final test of whole project.

I really learnt a lot by doing this project. For example, we need to cooperate with each other. It is true that we might be emotional in this process, but we need to face it and tackle it instead of avoiding it. In this project, I learnt how to scrape data by myself and grasp some knowledge in stock market which I never be involved in before. And I also know some basic knowledge about the frontend while assigned to work with Qiyuan.

I will try to make our goal much clearer. Because at the last half part of this semester, I think we already have done most of work decided by our discussion at the beginning of this semester. But I think we are addicted to add more functionalities instead of making sure what we have done is right. If our goal was much clearer at the beginning, then we could check what we have done first and go on to develop more. That should be much less riskly.

I will continue to use ML and DL to try to predict the fluctuations of the price of stocks.

You should make you goal of the project clear so that you can know what you have done. Do not be addicted to make the project more and more complicated, you should make sure your goal was achieved correctly and go on to add more into it. Otherwise, it will be not meaningful.