Project Selection: Ajay Gupta

1. What problem is the product aimed at solving?

This project aims to compare historical prices of different stocks over a long period, and visualize them on a single time series chart spanning several years (2 Decades). I have yet to see any easy service that compares stocks and other economic indicators in a really simple way. So I thought it would be interesting to make one.

2. Who is the product geared towards (targeted audience)?

This project is aimed at regular working-class passive investors who need a bit more insight and compare stocks quickly to take a wise investing decision.

3. How is the product unique?

It will be super easy to use: For this project, I have just brought stock data but I would continue building it to add more economic indicators like income equality, unemployment, GDP, and Climate data which you can superimpose on a single time series graph.

Highly scalable. With the current software design, it will be really easy to bring more and more datasets.

Techstack

- **Database:** MongoDB is used as a persistence store. It's a no SQL database so requires minimal deployment efforts and is ideal for data science projects where the data structure of external services is unknown and unpredictable.
- **External API:** For this application, I am using Alpha Vantage (https://www.alphavantage.co) a free public API to get the historical stock prices of a particular stock by Ticker (Example:TSLA). Their documentation is pretty simple and has API key-based access.
- **Frontend:** React web app interacting with a backend service. This will be hosted on AWS Amplify with full CI/CD using amplify.yml file (In Progress)
- **Backend:** There are two separate serverless backend services that you can run in parallel. Written in Python using FastApi framework and deployed on a Highly Scalable **Modal Serverless Platform** with full CI/CD pipeline managed through Github Actions. It is similar to AWS Lambda but with minimal deployment effort. https://modal.com
  - Collector Service: To collect data and save it to MongoDB upon request. It will check if data exists or is stale and then update it on demand. The update trigger will be managed by a simple messaging queue.
  - Analyzer Service: This is a Rest Api service connected to the web application and has Api to display data on the front-end web application.

Workflow and System Architecture

Analyzer Service retrieves stock data from MongoDB, preloaded by a **Collector Service**. If the data is unavailable in MongoDB, it queries the external Vantage API and immediately responds to the user (to avoid user wait time). In parallel, it sends a message to the collector service to load the data into MongoDB for future requests asynchronously.
If multiple users access the same stock data, the data is loaded only once by whoever requests it first.