2017 Engineering Essentials

Stock Visualization Application Requirements

Overall Objective

Capital markets play an important role in helping drive job creation, innovation and financial security. They enable people to save for retirement, afford to buy homes, finance their education, and grow their businesses, and they enable communities to get funding to provide necessary services. If you haven't done so already, learn more about Capital Markets with this interactive visualization: http://www.goldmansachs.com/s/interactive-guide-to-capital-markets/.

Financial intermediaries, buyers and sellers are key players in these markets – they need to be able to understand large amounts of data to accurately make decisions. Your task is to create an application that can help these parties visualize and analyze stock data. Through the process, everybody on the team should get the opportunity to work on all layers of the stack and the presentation.

Data Source

- You will be provided with 2 flat files that contain data on which you will build services:
    - Company data: caseStudy/services/src/main/resources/data/companyInfo.json
    - Historical stock data: caseStudy/services/src/main/resources/data/historicalStockData.json*

*Please note that the historical stock data represented in this file is fake and does not represent the actual market values.

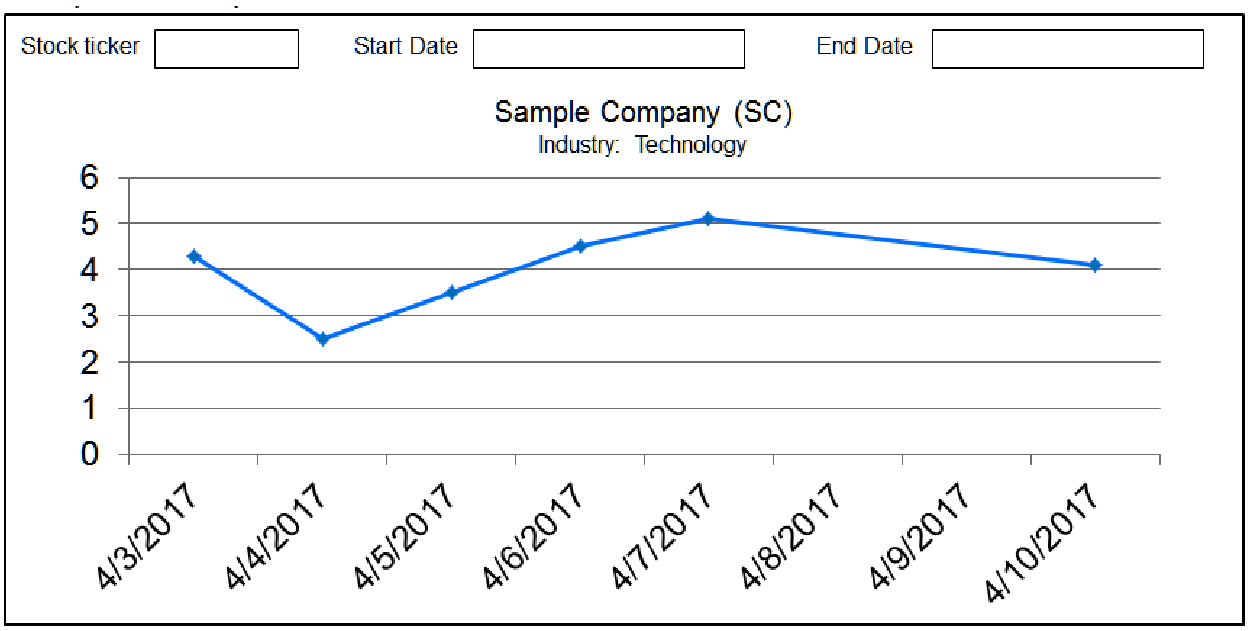
Service

- You will be writing 2 RESTful services that retrieve the data from the flat file and return it to the user
- Don't forget to validate the input files and input values before using them in your service! (e.g. stock ticker != null)
    - Company service (GET)
        - Input: stock ticker
        - Shell structure: Case Study/services/company
    - Stock price service (GET)
        - Input: stock ticker, start date, end date
        - Shell structure: Case Study/services/stock

UI

- The user interface for your application should include the following features:
  – Filters for company stock ticker, start date, and end date
  – Display the company's metadata (name, stock ticker, industry, etc.)
  – A line chart to represent the data retrieved, according to the filters selected
    - Title: Company name
    - X-axis: date
    - Y-axis: stock value

Simple Example



Final Presentation

- The case study will culminate with a final team presentation, in which you will present your application to panelists, describe the technical implementation and discuss any challenges faced. We will have a more detailed session on the best practices around presenting a technical solution.

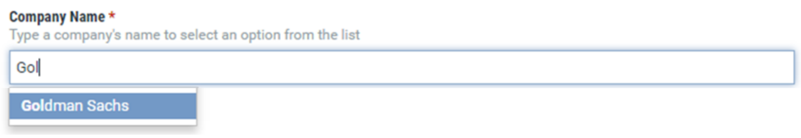
2017 Engineering Essentials

Stock Visualization Application Requirements

Additional Features

If you have extra time, here are a few potential enhancements for your application that you can design. Feel free to come up with your own ideas as well!

1) Add a filter option for company name with a predictive type ahead



2) Create a single type ahead component that can search on both stock ticker and company name
3) Add inputs to your services for each of the attributes in the data source to get more precise data (e.g. add an industry input to your company service to only get companies in a particular industry)
4) Add additional filters in the UI to match the inputs you added for the company service in the prior step.
5) Support the view of multiple stock tickers on the same page. This involves 2 parts:
    a) Add functionality to select multiple stock tickers – there are some existing components that you can leverage for this, or you can implement your own (e.g. https://github.com/JedWatson/react-select)
    b) You can choose one or more of the below options to display the data:
        i) Show all line charts for chosen tickers within the same chart
        ii) Show separate charts per chosen tickers
        iii) Give the user an option to choose their preferable way of seeing the multiple stock tickers (using a dropdown selector): whether they want to see them all in the same chart or individually
6) Provide the user with a pre-defined time period to be displayed in the chart by adding a dropdown that has multiple options: Past 24 hours, Yesterday and Past week. The displayed data should filter based on the option selected
7) Provide the user an option to see the data using a different type of chart (e.g. bar chart, pie chart, scatter plot, etc.)
8) Save the state (i.e. filters, etc.) in which the user closed the application and reload it with the same data when the user returns