CSS 481 Final Project: Stock Website

By Phuong Vu

A) Part 1: Project Proposal

1. **User Interaction:**

   a. User Scenario #1: Search Company's Information

   - Users click on "Explore Company" button on the Homepage or "Company Information" link on the navigation bar to navigate to "Business Profile" page
   - The "Business Profile" page offers Apple's information as default.
   - Users type in their desired stock's ticker (ex: AMZN, META, TSLA, …) to the search bar on the top right of the navigation bar.
   - Then, users click on the search button to submit their request
   - The page would be updated with information about the company that the users requested
   - The "Business Profile" page provides the following information about the input company:
     + Name
     + Description
     + Address
     + Phone Number
     + Website
     + Total Employee Number
     + Sectors that the company is in
     + Stock Exchange Ticker
     + Primary Listing Exchange
     + Date that the company was first publicly listed on the stock market
     + Market Capitalization
     + Outstanding Shares Number
     + Weighted Outstanding Shares Number

   b. User Scenario #2: See News in Stock Market

   - Users click on the "Stock News" link on the navigation bar to navigate to the "Stock Market News" page.
   - The "Stock Market News" page offers a set of 10 preset news articles as default.

- Users click on the button with the newspaper icon on the top right of the navigation bar to request the most recent news articles
- The page would be updated with the most recent set of 10 news articles (based on published time).
- Users click on the image or title of the news article.
- The page would direct the users to the website/link of the news article to read the article.
- On the "Stock Market News" page, each article provides the following information:
  + Image for the article, an attached link that directs to the article URL
  + Title for the article, an attached link that directs to the article URL
  + Description/Summary of the article
  + Tags of stock tickers, which associates or relates to the article
  + Publisher
  + Author
  + Publish time

c. User Scenario #3: Search Stock's Daily Trading Statistic/Price

- Users click on "Explore Stock" button on the Homepage or "Stock Price" link on the navigation bar to navigate to "Stock Quote Report" page
- The "Stock Quote Report" page offers Apple's information as default.
- Users type in their desired stock's ticker (ex: AMZN, META, TSLA, …) to the search bar on the top right of the navigation bar.
- Then, users click on the search button to submit their request
- The page would be updated with daily trading statistics/prices of the company's stock that the users requested
- The "Stock Quote Report" page provides the following daily trading statistics about the input company's stock:
  + Stock Exchange Ticker
  + Opening Price at the beginning of the trading session
  + Closing Price by the end of the trading session
  + Highest Price of the stock throughout the trading session
  + Lowest Price of the stock throughout the trading session
  + Number of Transaction made within the trading session
  + Trading volume of the trading session
  + Volume weighted average price of the whole trading session

d. User Scenario #4: Check Stock's Price History Chart
   - On the "Stock Quote Report" page, which can be accessed by clicking on "Explore Stock" button on the Homepage or "Stock Price" link on the navigation bar, users click on the graph button at the top right of the navigation bar
   - The "Stock Price Chart" page offers Apple's graph as default.
   - Users click in the drop-down box to choose the company they want to investigate stock price history
   - The page would be updated with the graph of the chosen company's stock price history
   - The graph on the "Stock Price Chart" page provides the following information about the Company's stock:

     + Company name

     + Most recent date that the data was collected for the chart; an indicator of whether the stock price has increased or decreased; as well as opening, closing, highest and lowest price of the stock on that day

     + The x-axis is the stock price(US $) and the y-axis is the timeline for 1 year (03/07/2022 – 03/07/2023)

     + Candle sticks to illustrate the stock's price and its trend. When the users hover over each candle stick, the company's price information (date, company, opening, closing, highest and lowest price) will be displayed

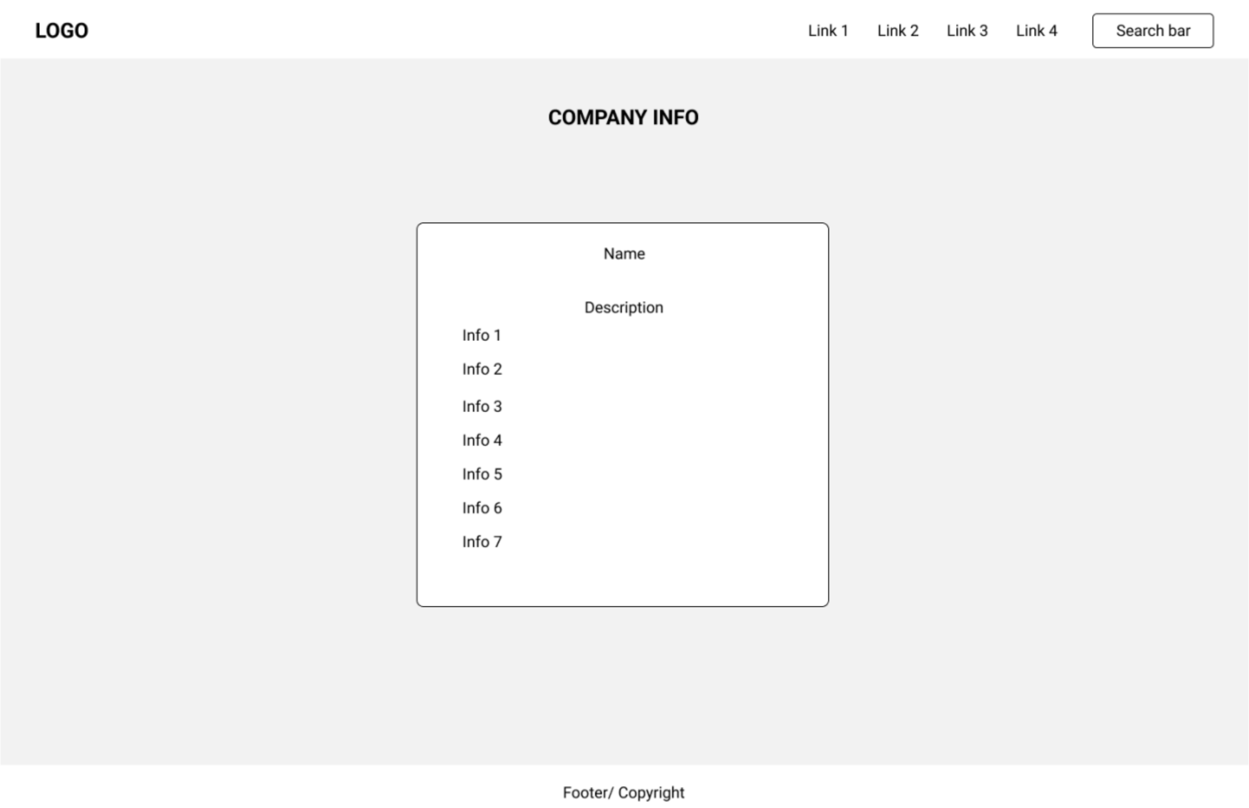
     + A timespan at the bottom of the graph

     + At the bottom left of the page, there is a time input that the users can put in the range of time they want to investigate the stock price

     + At the bottom right of the page, there is a time duration, which can be used as a zoom tool for the graph. The users can see the stock's price for the latest 3 months, 1 month, 10 days, 1 year,…

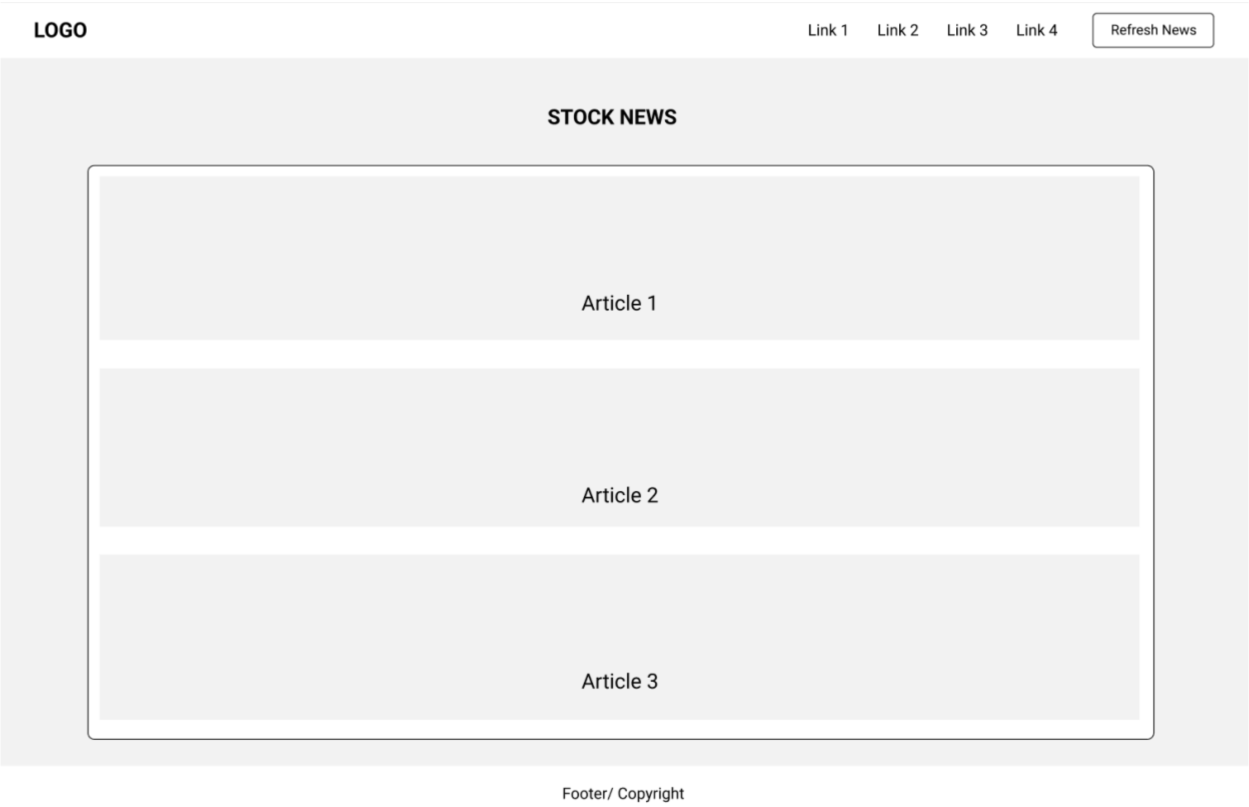
2. **Wireframe Diagram:**

   a. User Scenario #1: Search Company's Information



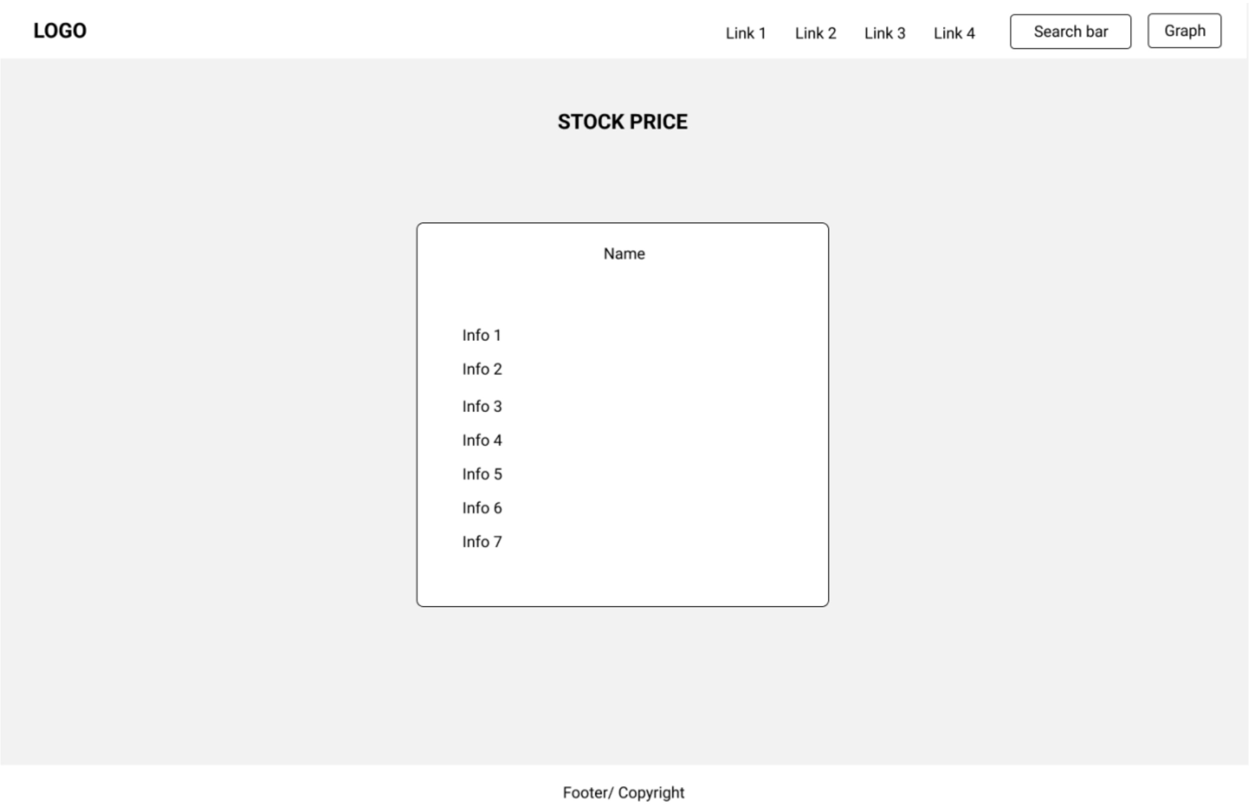
- The click target in this diagram is the Search bar. When the user enters a stock ticker in the search bar and clicks on the search button, the user would be able to see the company information of the stock ticker that they entered earlier.

b. User Scenario #2: See News in Stock Market



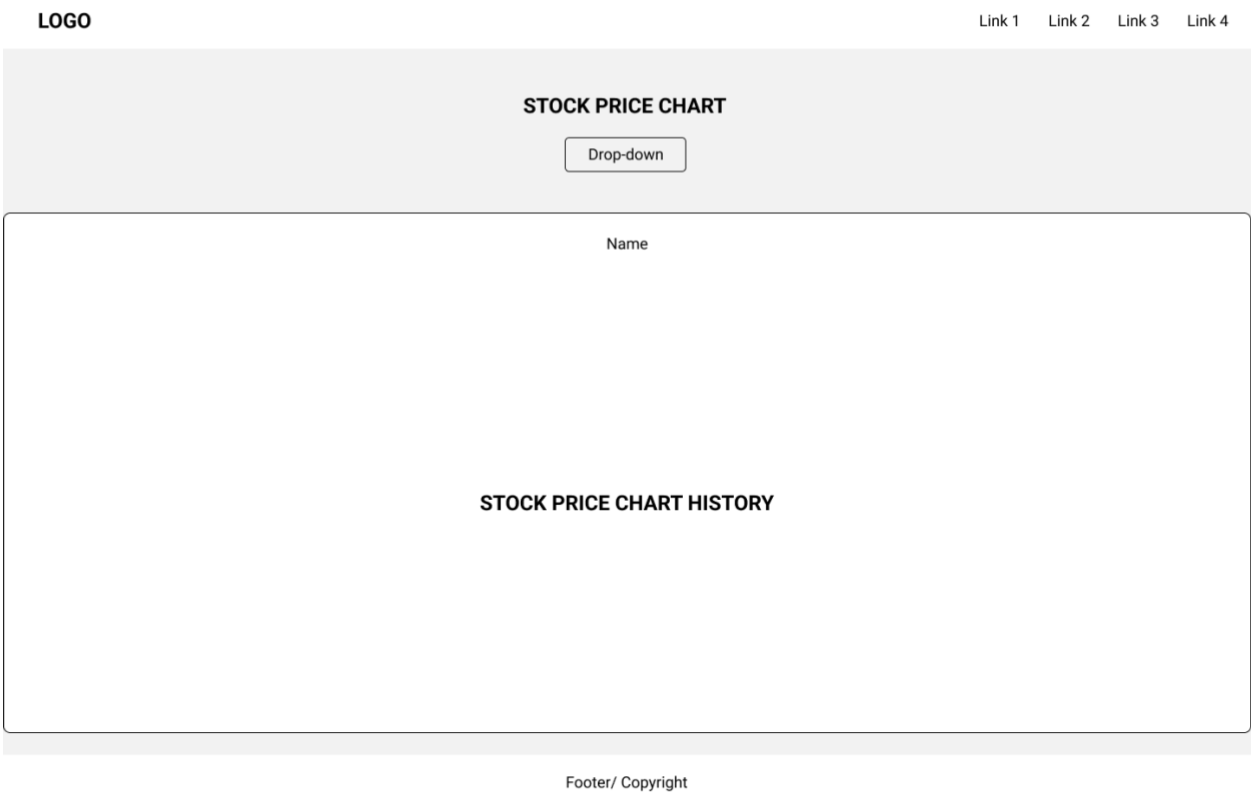
- The click target in this diagram is the Refresh News button. When the user clicks on the refresh news button, the user would be able to see the 10 most recent news in the stock market based on publication time.
- Another click target is the either image or title of the article. When the user clicks on the image or title, they will be directed to the original URL of the article.

c. User Scenario #3: Search Stock’s Daily Trading Statistic/Price



- The click target in this diagram is the Search bar. When the user enters a stock ticker in the search bar and clicks on the search button, the user would be able to see the daily price of the stock ticker that they entered earlier.
- Another click target is the Graph button. When the user clicks on the graph button, the user would be redirected to the Stock Price Chart page to investigate the stock price history.

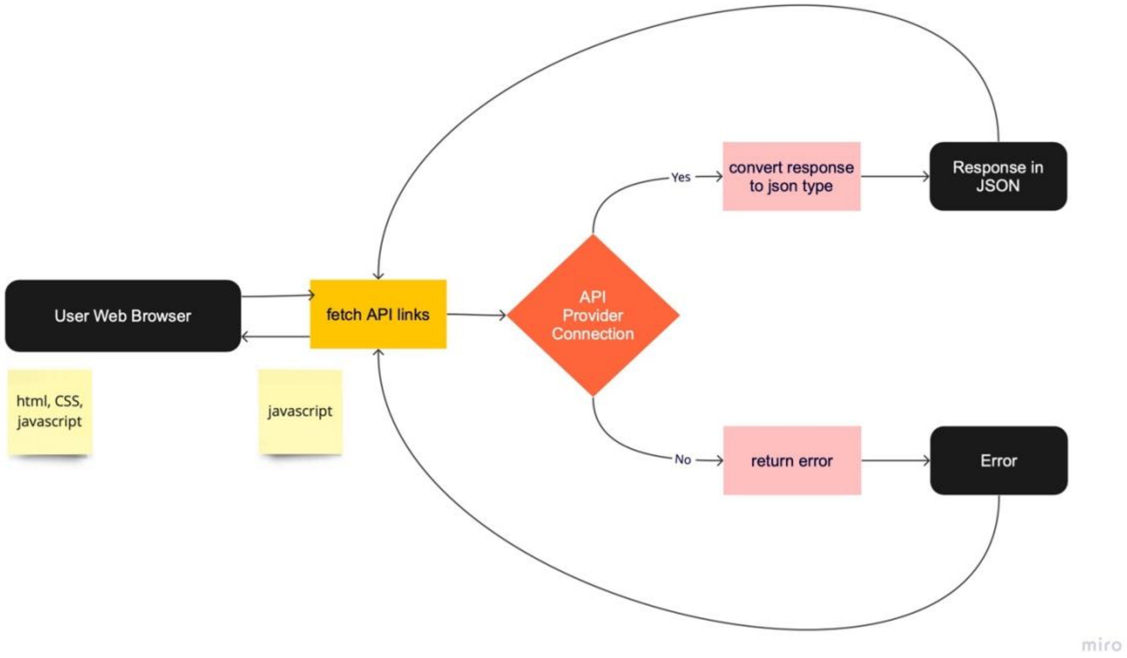
d. User Scenario #4: Check Stock's Price History Chart



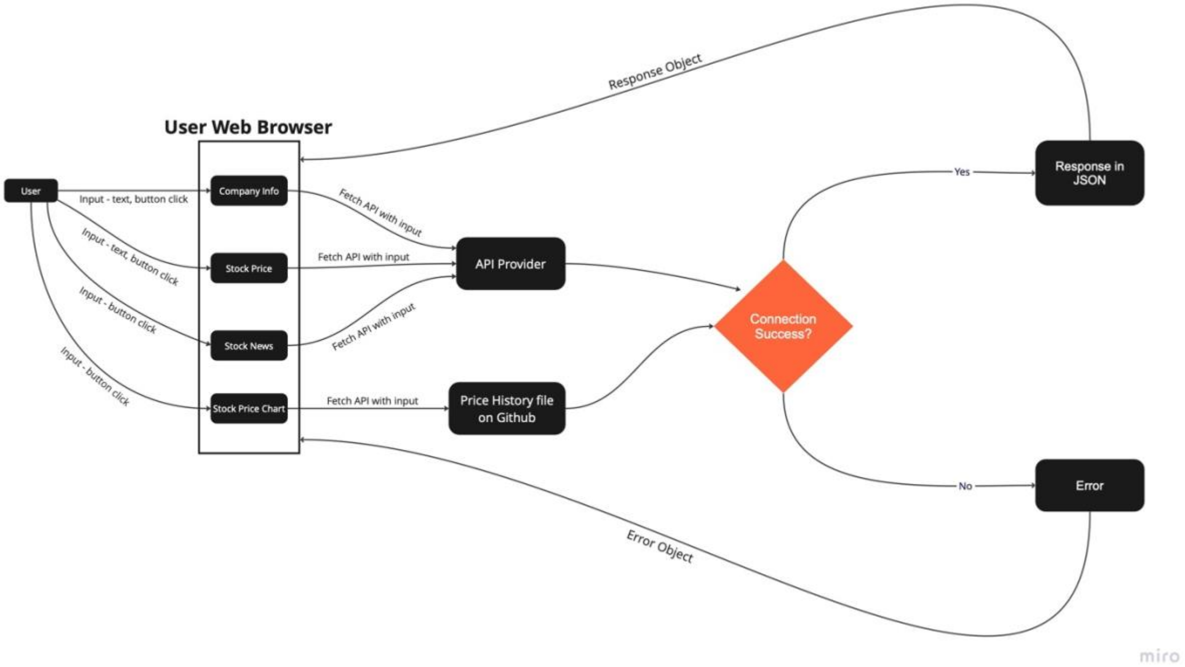
- The click target in this diagram is the Drop-down. When the user clicks on the drop-down button, the user would be able to choose a stock ticker that they are interested in. After the user clicks on the stock ticker that they want, the user would see a chart of the stock ticker's price history over the past year.

3. Architecture Diagram:

a. Primary object:

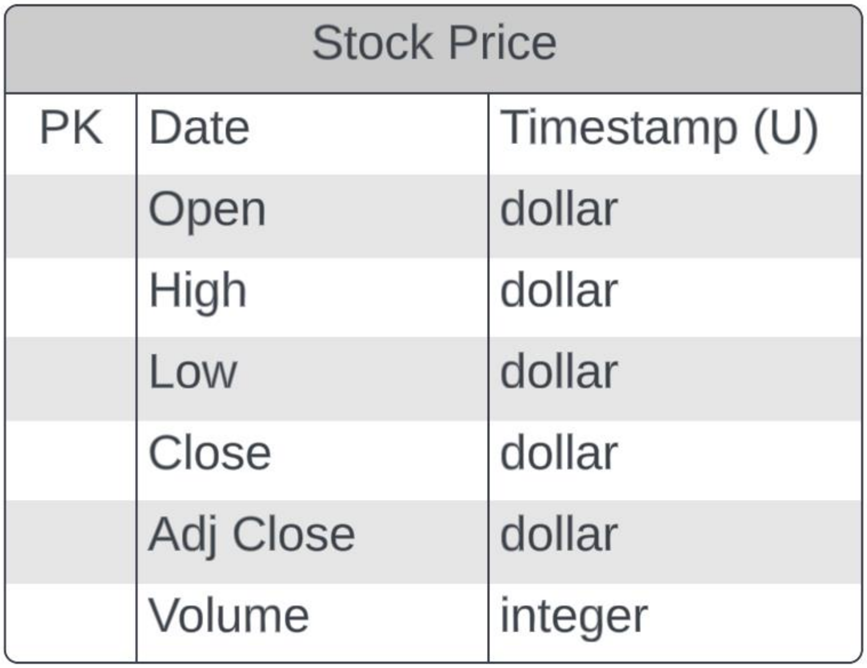


b. Data Flow:



4. Database:

- Yahoo Finance: I download the price history of different companies in form of CSV files from Yahoo Finance. Then, I uploaded those data files to GitHub for links to request data in my JavaScript file



Stock Price		
PK	Date	Timestamp (U)
	Open	dollar
	High	dollar
	Low	dollar
	Close	dollar
	Adj Close	dollar
	Volume	integer

5. **Comparison with other competing applications:**
   a. Competing Application #1: Robinhood
      - Similarity:
        + Stock Website
        + Offer Stock Price History and Stock Daily Prices

      - Difference:
        + Stock Portfolio offers news in the stock market, as well as company information. My website also does not require users to sign up to utilize the website's functions.
        + Robinhood is a stock trading app. It requires users to either sign in or create an account to navigate to the app's homepage and access its functionalities. Robinhood does not offer a page for

stock news, but instead, it offers news related to the company on the company's page.

b. Competing Application #2: Etrade

- Similarity:

  + Stock Website

  + Offers Stock Price History, Stock Daily Prices, and Stock News

- Difference:

  + Stock Portfolio is a free website for hobby purposes, and the users of my website do not need to sign in or create an account to use it. My website's user interface is straightforward and easy to navigate, even for new incoming stock players.

  + Etrade is heavily a trading website. It mainly focuses on trading techniques and analysis. As a result, it requires users to either sign in or create an account with their subscription to access all their functionalities. In addition, their website is difficult to navigate for a new person who is interested in the stock market.

6. Testing:

- Postman: I will use Postman to test API fetch and connection for my website. I will put the API link into Postman and generate information. Then, I will compare it against the information that the fetched API request in my JavaScript code to make sure that my website is fetching the API link correctly.
- Chrome's Dev Tool: I use Chrome's Dev Tool to inspect and print out testing statements to the console to ensure my website is behaving the way I expected it to. I also use Chrome's Dev Tool to inspect my website's user interface.

7. 3rd Party Components:

- Font Awesome
- NPM (live-server)
- Github
- W3School

8. List of Tooling:

- IDE: Visual Studio Code
- Website Hosting: W3School
- Github: Hosting data file from Yahoo Finance
- Source Repo:

  + Follow this tutorial for building the stock price history chart: Stock Chart Tutorial

9. Data Providers:

- Polygon.io
- Yahoo Finance

10. List of Assets:

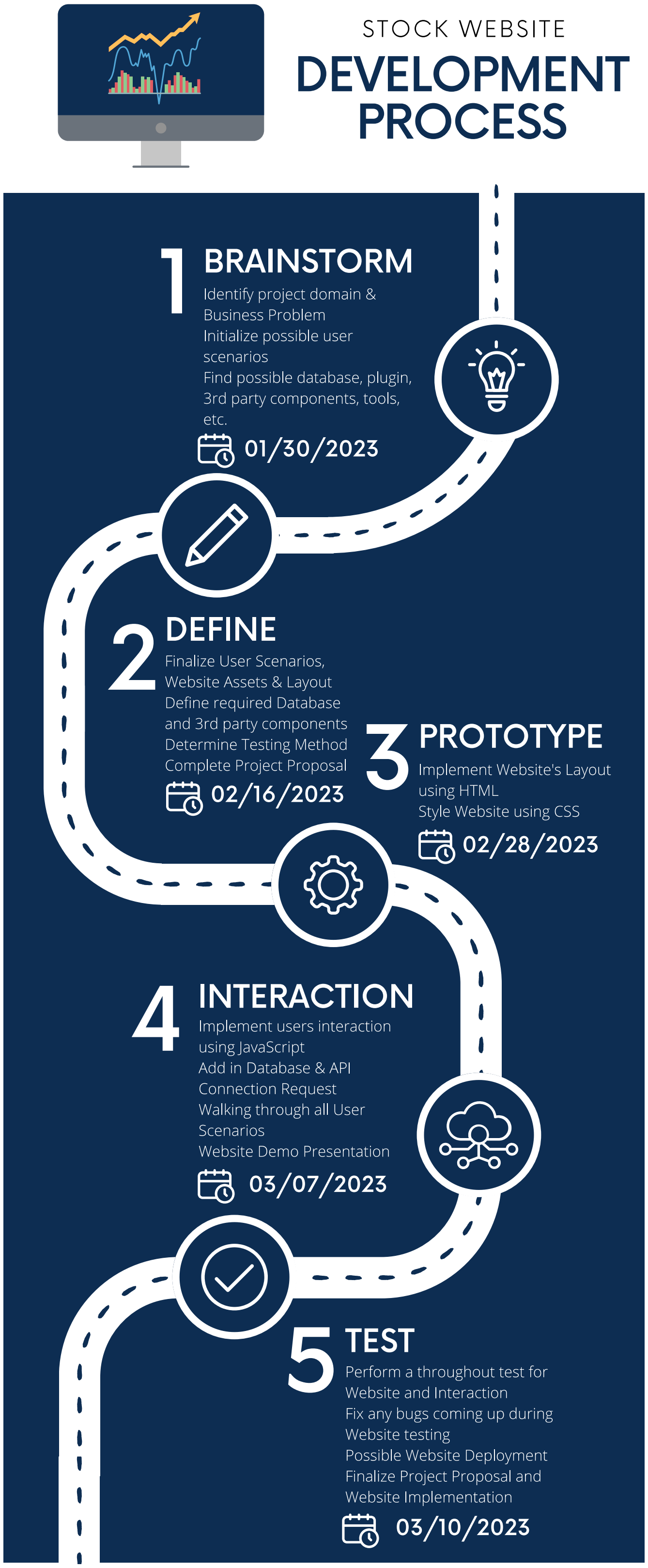
a. Color Palette:
   - #E0E0E0
   - #080808
   - #547793
   - #FFFFFF
   - #645D5D
   - #C1C6CC
   - #807D7D
   - #F4FAF9
   - #96A4A9
   - #E7E7E7
   - #5A64D3
   - #1274ED
   - #000000
   - #C6CDD5
   - #333333

b. Front List:
   - Courier
   - Poppins
   - Sans-serif

c. Images:

CHARLES VU

11.Development Timeline:



STOCK WEBSITE

DEVELOPMENT PROCESS

1 BRAINSTORM

Identify project domain & Business Problem
Initialize possible user scenarios
Find possible database, plugin, 3rd party components, tools, etc.

01/30/2023

2 DEFINE

Finalize User Scenarios, Website Assets & Layout
Define required Database and 3rd party components
Determine Testing Method
Complete Project Proposal

02/16/2023

3 PROTOTYPE

Implement Website's Layout using HTML
Style Website using CSS

02/28/2023

4 INTERACTION

Implement users interaction using JavaScript
Add in Database & API Connection Request
Walking through all User Scenarios
Website Demo Presentation

03/07/2023

5 TEST

Perform a throughout test for Website and Interaction
Fix any bugs coming up during Website testing
Possible Website Deployment
Finalize Project Proposal and Website Implementation

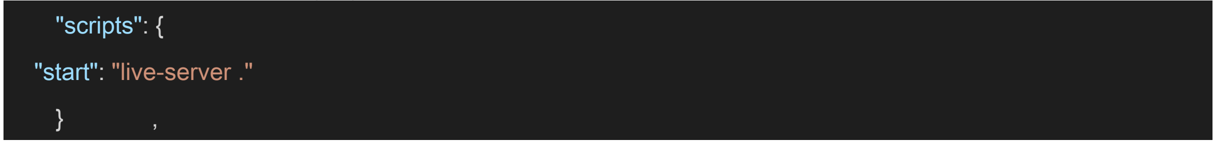
03/10/2023

B) Part 2: Required Dependencies

1. **Font Awesome:** icon for website
   - Sign up for an account with Font Awesome
   - Create your own account
   - Once your account is created, create a Font Awesome Kit.
   - Replace the Javascript Embed Code for your kit in the Script tag in the “head” tag of all html files

2. **Polygon.io:** Stock data API
   - Sign up for an account with Polygon.io
   - Create your own account
   - Once your account is created, choose basic free plan
   - Get your own secret API Key
   - Insert your API in the URL in the index file

3. Install **live-server** for localhost:
   - “npm install live-server --save-dev” at the project folder Terminal
   - In “package.json” file, add in:



```
  "scripts": {
"start": "live-server ."
  }        ,
```

   - Then, in the Terminal, type in “npm start” to start hosting the website on localhost

4. **Building Stock chart:**
   - Follow the instructions in this link

5. **Website Hosting:**
   - Create an account at W3school here
   - After creating an account, create a Space
   - Upload all the code to the space
   - The website link is generated at the top left of the project’s space page
   - The website is host at this link: https://stock-portfolio-web.w3spaces.com

C) Part 3: Development Instruction

Notes: Please have all the dependencies stated above installed and register before moving on to this instruction section

1. **Writing static website with HTML and CSS**
    a. Set up project:
        - Create a repository on Github
        - Name the repository project name.
        - On desired folder/path of computer, using Terminal to git clone the project link.
        - Project is ready in the folder to begin with.
        - Create a folder called images to store images for the website
        - Create a folder called data to store datafiles for publishing to Github later
        - Create a html folder to store html files of different pages for the website

    b. HTML:
        - Create an index.html file with all the required tags in the head tag including meta, title, link, and script. The meta and title tags are required for html file setup. The link and script tags are required to link with stylesheet, favicon image, external package for icon, and JavaScript files.
        - Create a div of header part, which include the navigation bar and welcome page div.
        - Create the navigation bar with all necessary link to different pages. The href of the a tag should be the path of current page to other pages. Include in the navigation bar two icons: one for closing and one for opening navigation when screen is too small to show all the navigation contents.
        - Insert background image into the front page of index.html.
        - Put in welcome message with short cut button on top of the background image.
        - Create a footer for copyright information.
        - In the html folder, create 4 different html file for different pages of the website: companyinfo.html, price-chart.html, stocknews.html, and stockprice.html
        - Copy the content in the head tag, navigation div and copyright/footer div from index.html into the 4 newly created html files. Correct the path to different pages/files accordingly to current file location.

- Companyinfo.html:
  + Create a div in the body of the page between the navigation bar and footer for page content
  + Create a search bar to put into the navigation bar
  + Create a heading for the page
  + Create a div to store the company information
  + Put in Apple's information as default on the page
  + Inside the information div, there should be company name, description and 11 other information to be implemented.
  + For each information, it should have an id to be altered based on users' search later

- Stockprice.html:
  + Create a div in the body of the page between the navigation bar and footer for page content
  + Create a search bar and graph button(linked to the price-chart.html) to put into the navigation bar
  + Create a heading for the page
  + Create a div to store the company information
  + Put in Apple's information as default on the page
  + Inside the information div, there should be company name, open, close, highest, lowest price, number of transaction, trading volume, volume weighted average price.
  + For each information, it should have an id to be altered based on users' search later

- Stocknews.html:
  + Create a div in the body of the page between the navigation bar and footer for page content
  + Create a refresh button to put into the navigation bar
  + Create a heading for the page
  + Create a div to store all the article
  + Put in 10 news article as default on the page
  + Create a div for each article. Each div should have image, link to the article as title, description, span tag to display ticker name, and additional information like publisher, author, publication date
  + For each information, it should have an id to be altered based on users' search later

- Price-chart.html: Follow the tutorial in the Dependencies #4 stated above for this section
  + Create a div in the body of the page between the navigation bar and footer for page content
  + Create a heading for the page
  + Create a label and select tag to create a list of drop down company for user to choose from
  + Create a div to store the graph
  + Put in Apple's graph as default on the page
  + For each information, it should have an id to be altered based on users' search later
  + At the end of body tag, create a script tag to put in javascript code to fetch information and create the graph accordingly
  + Set the default value for company and stock ticker to be Apple and AAPL.
  + As the document load, draw the graph. When there is a change to the drop down button, update the value of company and stock ticker variables to be user's choice. Then, redraw the graph with user's input
  + Follow the tutorial stated above to draw graph.

c. CSS:
  - Set general styling rules for all the pages including margin, padding, font-family and box-sizing.
  - Set general styling rules for all the head, html body tag
  - Set general styling rules for all the navigation bar
  - Set styling rules for the front page including background image and welcome page
  - Set general styling rules for all html files in the html folder for their body-container div and heading inside the div
  - Set styling rules for the rest of the components inside the body-container to stay in the center of the page with proportional size.
  - After all the styling for all pages in full screen is complete, open Chrome's Dev Tool to inspect screen on the smaller screen
  - Alter styling rules to better accommodate smaller screen sizes.
    + Hide two opening and closing menu icon when the screen is big enough to show the navigation bar
    + These two icons would be shown when screen is too small to show all the items in navigation bar. Add styling so that the menu content in small screens goes in vertical order.

2. **User interaction with JavaScript**
    - Create the index.js file in strict mode
    - Add event listener for window when the dom is load and attach behaving function to be domLoaded()
    - Inside domLoaded() is all the behaviors would be executed when the page is loaded.

      + Add event listener and handling function for opening and closing menu button, which would be visible when the screen is small like mobile screen.

      + Add event listener and handling functions for users' input on the search bar and refresh button (stock news page) if the current page has the respective element
    - Create handling function to close and open menu for small screen size
    - Create handling functions to handle users' input on the search bar of company information and stock price page. In each of these function will call to another function to fetch API links
    - Create functions to fetch API links for the handling functions above, then render the received information to the page corresponding
    - Create handling function for the stock news page when users click on the refresh button. Inside this function, fetch API links and get the 10 newest article and render these article to the stock news page.

3. **Hosting Website**
    - In the terminal of the folder, use npm to install live-server. Testing the website interation on local host and ensure all the render information matches with Postman's result for testing purposes.
    - Upload all the code files to project's space on W3School. The website is hosted automatically with link at the top of the project's space page.
    - The website is hosted at: https://stock-portfolio-web.w3spaces.com/index.html