**Date**: 04/25/2022

# Overview

Create a site that will display a landing page, display live stock market performance (break-down of the S&P 500 into its sectors), and allow the user to create a stock favorite/monitoring list that shows live valuations.

Created a site for users to log into to

1. Use lands on the Home Page
   1. See a random thought of the day
   2. User can login to get the Navigation Links enabled
2. Created a route/component where all 11 S&P Sectors are displayed
   1. Color coded, green: positive & unchanged; red: negative
   2. Stock Market data is refreshed every 5 seconds
3. View a footer on each page

# Technologies

1. Login Component is a Reactive Form
2. Stock Market Data is Service that calls an API to retrieve current prices of Sectors
   1. Transforming the returned API data to an Object/Class Model
      1. Not parsing the data
   2. Using the Subject approach, not the EventEmitter approach
3. Using a custom Timer Subscription to call API on a scheduled frequency
4. Created a fake (hard-coded) authentication and authorization service to control access to site features
   1. Login and Logout capabilities
   2. Uses Validator with error text displaying help info (red in color)
   3. Access to certain navigation links is restricted based on authorization.
   4. Handles invalid credentials.
5. Created a component to serve up a random market-related thought/saying/meme.

### Project Makeup

9 Components

1. App Component
2. Home page
3. Login component
4. Footer component
5. Navigation bar component
6. Random quote component
7. Stock listing component
8. Stock card component
9. My stock list component (just a stub)

2 Services (Using Subject/Next to expose data, not EventEmitters)

1. Authentication Service (handles authorization)
2. Sector Data Service (retrieves stock market data

2 Class Object Models

1. API Object Model – stored all data retrieved from an API Call
2. Internal Stock Data Object Model – stored the data that the site needs

1 Constants file

1. Stores global constants for the file, mainly the API data

# Things I’m most Proud about

**Learning**

1. Figured out how to call the API
2. Transforming the returned API data to a Class
3. Using Cards and Card Deck to display information

**Visuals**

1. The cards change order based on the % gains
   1. always ordered best to worst
   2. Color Coding - green is positive/unchanged; red is negative
2. Locked down navigation based on user's authorization
   1. disabled navbar links, didn't hide the links
   2. admin: disabled My-Stocks
   3. guest: disabled Sectors
   4. user: nothing disabled

# ToDo’s and Maybe’s

1. Complete the My-Stocks component
2. Lock down navigation to not allow manually typing in the URL
3. Play around with the CSS and colors and improve the look-and-feel
4. Show the S&P 500 performance
5. Maybe break the API call from the Service
6. Custom icon to replace the Angular icon in the browser tab
7. Store user data in a repository
8. **Gamification**