This is a report outlining the process I followed while building the web app for the Qwilr challenge. I **divided the problem statement** into the following technical tasks:

1. Check out **AlphaVantage** for real-time data on stock prices and see how to use its APIs

2. Build a **web page** to display (a) the current assets in cash and stock holdings, (b) **Buy** and Sell options followed by options to what to buy or sell, (c) updated **amount** of assets after any buy/sell operation

3. Setup application server

4. Optional: Host the app on **Heroku**

The first target for me was to build the back-end as that is my strong suite and then proceed to put a front-end in front of it. However, knowing that a spring-boot-starter would help me in both front-end and back-end development was crucial, as other frameworks might later have had to be changed if I built the app keeping only back-end in mind.

Although I was able to use the AlphaVantage API to return the price of a stock from my native Postman app once I went through all of its APIs, the same was not so simple from the Java code I wrote. This was due to a certificate that AlphaVantage requires which I didn’t have included in my jdk by default. This slowed me down during back-end development, as I had to try out both SSL-enabled and disabled end-points of AlphaVantage. When neither worked, it was up to me to go ahead with resolving this issue or switching the real-time data provider I was using to a simpler one. I decided to go ahead with the same one I had chosen earlier, as in a corporate scenario, I wouldn’t have had that choice and would have had to learn how to use what was given to me. I proceeded to download their certificate manually and learned how to include it in my JRE so that my code could return me the desired results on my local system.

Getting back a valid response from the required end-point led to my next step, putting a web-page in front of my REST API. I did not follow any good front-end coding practices and included everything from HTML/CSS to JavaScript in one index.html itself. If I had more time at hand, I would have separated out functionality from view on the front-end.

Although my app does not meet all its requirement (providing a buy/sell option and allowing the user to view their current cash and stock holdings), I have decided to stick by the timeline and stop development here for now. Since hosting the app on Heroku seemed like an interesting task, I went ahead with that.