Financial Sentiment Based on News: Test Results

Test Results

Test ID	Pass/Fail	Notes + Steps Taken After
U_FRONT1	Pass	All icons displayed correctly in the correct location
U_BACK1	Fail	Autocorrect functionality didn't work as our search wasn't specific enough. We decided to search for "{company_name} stock" instead of "{company_name}" to resolve this
U_BACK2	Pass	Used yahoo finance for correct results
U_BACK3	Pass	Similar to U_BACK1, by searching for "{company_name} stock articles," we found accurate stock articles
U_BACK4	Pass	Yahoo finance had good documentation for accurate results
U_BACK5	Pass	Implementing a Python dictionary worked well
U_BACK6	Fail	Asynchronous functionality would fail on occasion and raise errors By using try/catch statements we could ensure parsing was done correctly for valid documents
U_BACK7	Pass	The sentiment analysis algorithm we used was able to correctly parse text
I_FRONT1	Fail	Text overflowed for long article titles on mobile This was because the container had a set height, so we set a minimum height for the container of each article
I_BACK1	Fail	The format of the data was not consistent when null values were returned For invalid inputs, we returned the JSON with the structure of a valid input, but with invalid values to handle on the frontend
I_SYSTEM1	Pass	The queries were implemented correctly after many iterations of the back and frontend
V_SYSTEM1	Pass	The implementation of the cache helped boost efficiency and UX by a lot

Test Conclusions

All in all, by making modifications to our code, we were able to successfully pass all the tests we created. A lot of our challenges involved optimizations on the backend, whether it be getting more relevant articles or improving the efficiency of loading article data. On the frontend, there were some visual errors but we were able to overcome these and make a sleek, responsive design for our product.