Project Scope Update:
The scope of my project has changed minimally. Instead of loading all of the sponsor Google Trends data into one dataset, I have decided to retrieve data for each sponsor separately. Initially, I was interested in comparing sponsor trends against one another. However, I found that the general discussion of some sponsors is significantly higher than that of others, minimizing the data from lower-trending sponsors. Additionally, sponsor data should be compared to their own historical data, not others, because the priority is comparing their trends to the Warriors' performance.

Data Sources:
In Python, I retrieved data from the Golden State Warriors' news website (API), ESPN's season statistics (web scrape), and pytrends (API). All sourced data was converted into dataframes and saved to CSV files.

From the Warriors' news website, I extracted article details such as title, date, and description to examine historical sponsorship coverage. The NBA's official API requires a paid subscription, so I used an unofficial API URL that did not require a key.

To find historical game performance, I used Beautiful Soup to webscrape from ESPN's game schedule table. The table included date, score, season record, and player stats.

Lastly, I used pytrends, an unofficial API for Google Trends, to load data on interest over time for the Warriors and their major sponsors. The dailydata.get_daily_data() function retrieves the daily trends for a keyword over a designated period of time. The function pulls data by month and scales the data across the full time period.

Issues/Difficulties:
Originally, I tried webscraping the news website, but ran into trouble when the site only loaded the 10 most recent articles. After doing some research, I used Google Chrome's *Inspect Element* to find an alternative JSON file or API URL I could use to parse additional articles behind the 'load more' button. I found an API URL and implemented this formatting into my code.

While retrieving Google Trends data, the pytrends function initially returned a full dataset for the 5-year range I requested. However, when I attempted to retrieve more data for other keywords, the function returned a 429 (Too Many Requests) error. I assume this is because the .get_daily_data() is requesting data by month, and extracting 5 years of data is causing this error. Moving forward, I will need to wait more than 24 hours before requesting additional data.