Participants: Sravishtta Kommineni

For this project, I initially wanted to explore the Google AdWords API to programmatically make an Ad Campaign with various parameters and query the metrics.

The metrics could be the target gender, budget, target age, location, etc. And the metrics we can query from the API include the following:

Attribute	Segment	Metric
AccentColor	AdNetworkType1	AbsoluteTopImpressionPercentage
AccountCurrencyCode	AdNetworkType2	ActiveViewCpm
AccountDescriptiveName	ClickType	ActiveViewCtr
AccountTimeZone	ConversionAdjustmentLagBucket	ActiveViewImpressions
AdGroupId	ConversionCategoryName	ActiveViewMeasurability
AdGroupName	ConversionLagBucket	ActiveViewMeasurableCost
AdGroupStatus	ConversionTrackerId	ActiveViewMeasurableImpressions
AdStrengthInfo	ConversionTypeName	ActiveViewViewability
AdType	CriterionId	AllConversionRate
AllowFlexibleColor	CriterionType	AllConversions
Automated	Date	AllConversionValue
BaseAdGroupId	DayOfWeek	AverageCost
BaseCampaignId	Device	AverageCpc
BusinessName	ExternalConversionSource	AverageCpe
CallOnlyPhoneNumber	Month	AverageCpm
CallToActionText	MonthOfYear	AverageCpv
CampaignId	Quarter	AveragePageviews
CampaignName	Slot	AveragePosition
CampaignStatus	Week	AverageTimeOnSite
CombinedApprovalStatus	Year	BounceRate

We can use Google's AdWords Query Language (AWQL) for querying these metrics.

I was not able to set up the account properly in order to programmatically launch an Ad Campaign, so I decided to do a project with an emphasis on exercise on tools.

I decided to make a web-app called Stocker, which lets people track their stock portfolio. User make a new account, log-in, search companies and add them to their portfolio, and they will be able to see the real-time price of the company's stock.

In class I demonstrated the creation of new account, logging-in, search of the various companies, and adding them to the portfolio.

The tools I used to make this project are: Firebase Authentication, Firebase's Firestore NoSQL Database, ReactJS for front-end and connecting to the database and Real-Time Stock API. To implement the search feature, I had to find a list of all the possible ticker symbols, and then input them into Firestore using a python script, and then I implemented the search using queries in React.

This is the Stock API I used: https://financialmodelingprep.com/developer/docs/companies-key-stats-free-api

I request to using the following URL:
https://financialmodelingprep.com/api/v3/company/profile/AAPL

And I get the following JSON object from the request:


```
{
  "symbol" : "AAPL",
  "profile" : {
    "price" : 270.0,
    "beta" : "1.139593",
    "volAvg" : "36724977",
    "mktCap" : "1246815180000.00",
    "lastDiv" : "2.92",
    "range" : "142-233.47",
    "changes" : -0.12,
    "changesPercentage" : "(-0.04%)",
    "companyName" : "Apple Inc.",
    "exchange" : "Nasdaq Global Select",
    "industry" : "Computer Hardware",
    "website" : "http://www.apple.com",
    "description" : "Apple Inc is designs, manufactures and markets mobile
communication and media devices and personal computers, and sells a variety
of related software, services, accessories, networking solutions and third-
party digital content and applications.",
    "ceo" : "Timothy D. Cook",
    "sector" : "Technology",
    "image" : "https://financialmodelingprep.com/images-New-jpg/AAPL.jpg"
  }
}
```


To run the project, go inside the code folder, and type the command *npm start*, and then onces that is done, type *npm start*. This should open the browser at the address http://localhost:3000/. You will need NodeJS installed in order to do this.

Through this project, I wanted to familiarize myself with working knowledge of ReactJS and how a React App will connect to Firebase Authentication and Firestore Database.

A React App is split into components, for example, the Login Page is a component, the Dashboard is a component, each Stock info is a component, so the code I have written is located in src/components folder.