Index Data of Stock Market

A PROJECT REPORT

Submitted by

AMAN KUMAR

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

Vellore Institute of Technology

Vandalur - Kelambakkam Road, Chennai - 600 127

April - 2022

School of Computer Science and Engineering

DECLARATION

I hereby declare that the project entitled “Index data of stock market” submitted by me to the School of Computer Science and Engineering, Vellore Institute of Technology, Chennai, 600 127, in partial fulfillment of the requirements of the award of the degree of Master of Technology in Business Analytics (5 year Integrated Program) and as part of CSE3121 – Information Visualization Project is a bona-fide record of the work carried out by me under the supervision of Prof. Dr. Arun Kumar Sivaraman. I further declare that the work reported in this project, has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or University.

Aman Kumar
Signature of Candidate

Place: Chennai
Date: 27-04-2022

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

CERTIFICATE

This is to certify that the report entitled “Index Data of Stock Market” is prepared and submitted by Aman Kumar (Reg. No. 20MIA1144) to Vellore Institute of Technology, Chennai, in partial fulfillment of the requirement for the award of the degree of Master of Technology in Business Analytics (5 year Integrated Program) and as part of CSE3121 – Information Visualization Project is a bona-fide record carried out under my guidance. The project fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission.

Guide/Supervisor

Name: Dr. Arun Kumar Sivaraman Bhargavi
Date:

HoD

Name: Dr. Renta Chintala
Date:

Examiner

Name:
Date:

Examiner

Name:
Date:

(Seal of SCOPE)

Acknowledgement

I obliged to give my appreciation to a number of people without whom I could not have completed this thesis successfully.

I would like to place on record my deep sense of gratitude and thanks to my internal guide Prof. Dr. Arun Kumar Sivaraman, School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, whose esteemed support and immense guidance encouraged me to complete the project successfully.

I would like to thank our HoD Dr. Renta Chintala Bhargavi, School of Computer Science and Engineering (SCOPE) and Project Coordinator Dr. Rabindra Kumar Singh, Vellore Institute of Technology, Chennai, for their valuable support and encouragement to take up and complete this thesis.

Special mention to our Dean Dr. Ganesan R, Associate Dean Dr. S. Geetha, School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, for motivating us in every aspect of software engineering.

I thank our management of Vellore Institute of Technology, Chennai, for permitting me to use the library and laboratory resources. I also thank all the faculty members for giving me the courage and the strength that I needed to complete my goal. This acknowledgment would be incomplete without expressing the whole hearted thanks to my family and friends who motivated me during the course of my work.

Aman Kumar
20MIA1144

Abstract

Without a doubt, stock markets are an important and necessary part of any country's economy. However, the impact of stock markets on a country's economy may differ from the impact of stock markets on other countries' economies. This is due to the impact of stock markets on the economy being determined by several factors, including the organization of stock exchanges, its relationship with other financial system components, and the country's governance system, for example. All these variables are unique to each country. As a result, the impact of stock markets on the economy of a country is distinct. The Indian capital market system has undergone major fundamental institutional changes over the years, resulting in lower transaction costs, increased efficiency, transparency, and safety. All these changes have resulted in the economy's development through stock markets. Therefore, high demand for stock market development is expected. The current paper and visual technology will help to predict the future stock prices based on previous stock data and analyse the stock data. And we are going to help others to understand and easily invest in the stock market with the help of our visualization and it can also help them to predict the data with the data visualization done by Tableau and python.

Contents

Title:
Index Data of the Stock Market

Authors & Affiliations & Address:
Garrepally Sai Charan,20MIA1056, Vellore Institute of Technology, Chennai, India.
Viswa Kirthik,20MIA1165, Vellore Institute of Technology, Chennai, India.
Aman Kumar 20MIA1144, Vellore Institute of Technology, Chennai, India.

1. Introduction

We will be looking at data from the stock market, particularly some technology stocks. We will learn how to use pandas to get stock information, visualize different aspects of it, and finally we will look at a few ways of analyzing the risk of a stock, based on its previous performance history.

What is Index?
• Index is a tool that measures change
• Example: Human Development Index measures the development of a country.

What is the Stock Index?
• There are more than 7400 companies listed in both NSE (National Stock Exchange) and BSE (Bombay Stock Exchange). So, to track every company is a challenging task.
• To address this issue, the Stock Index was created.
• Stock Index, as the name implies, refers to changes in stock prices.

How is the Stock Index measured?
• Instead of calculating stock prices of every company they use the top 30 or top 50 companies' stock prices and see how the prices are changing whether they are increasing or decreasing.
• In India, Sensex (a stock market index) is made up of the top 30 stocks on BSE called Sensex30. Nifty (a stock market index) is made up of the top 50 stocks on NSE called Nifty50.
• The selection of these top 30 or top 50 companies will be based on many ways like market capitalization, sector etc... Both the BSE and the NSE utilize market capitalization as a benchmark.
• Market Capitalization = (Stock price) * (Total number of stocks).

Index value:
• Index Value = [(Free float market cap of grouped stocks/Base market cap)* Base index value]

• Free float market capitalization takes only the number of stocks which are publicly available for trade. • Base market capitalization is the total market value of stocks that were formed when Sensex was initially formed
• Base index value for Sensex is 100.
• Base index value for Nifty is 1000.
• So as the Sensex rises then the value of most of the stocks will increase.

U.S. Stock Market Indexes:

• In the United States the S&P 500, Dow Jones Industrial Average, and Nasdaq Composite are the three most broadly followed indexes by both the media and investors. There are more than 5000 other Stock Indexes used in USA.

S&P 500 (Standard & Poor's 500):

• The Standard & Poor's 500 Index (known commonly as the S&P 500) is an index with 500 of the top companies in the U.S. Stocks are chosen for the index primarily by capitalization
• constituent committee also considers other factors including liquidity, public float, sector classification, financial viability, and trading history.
• It represents approximately 80% of the total value of the U.S. stock market.
• Indexes are usually market-weighted or price-weighted. The S&P 500 Index is a market-weighted index (also referred to as capitalization-weighted).
• So, if the total market value of all 500 companies in the S&P 500 drops by 10%, the value of the index also drops by 10%.

How S&P 500 works?

The S&P 500 tracks the market capitalization of the companies in its index.
A company that has a market cap of $100 billion receives 10 times the representation as a company whose market cap is $10 billion
A committee selects each of the index's 500 corporations based on their liquidity, size, and industry. It rebalances the index quarterly, in March, June, September, and December.

Rules and Regulations:

To get listed by the S&P 500's committee the companies need to follow some rules and regulations.
1. A company must be in the United States.
2. The company should have an unadjusted market cap of at least $13.1 billion to qualify for the index.
3. At least 50% of the corporation's stock must be available to the public.
4. Its stock price must be at least $1 per share.
5.It must file a 10-K annual report.
6.At least 50% of its fixed assets and revenues must be in the United States.
7.It must have at least four consecutive quarters of positive earnings.

2. Planning and requirement

Stock Market data set and knowledge in python programming and Tableau is required in completing this project. The planning involved acquiring the data set and performing data analysis on the data set to find some information and presenting the information in the form of visualizations. And This dataset is collected data from the stock market daily transaction done by millions of people and it links is: Dataset . So, we are planning to visualise this data by using Tableau and Python code and help others to understand the data and help then to do transaction in Stock market.

3. System Design

The dataset was first linked in python IDE and Tableau to perform visualization and analysis, and we used Jupyter Notebook for this purpose as the code. After linking the dataset, I started cleaning the data and removed all the null values from the dataset. After that I performed analysis using python and Tableau.

4. Implementation of the system

After importing the dataset then we can start to do our analysis by plotting graph and summarizing the data. We use Tableau for graph and analysis.

Materials & Methods:

Dataset Link:

Dataset

Pandas:

It is a fast, powerful, flexible, and easy to use open-source data analysis and manipulation tool

Matplotlib:

It is a comprehensive library for creating static, animated, and interactive visualizations in Python. It makes easy things easy and hard things possible.

Seaborn:

Seaborn library aims to make a more attractive visualization of the central part of understanding and exploring data. It is built on the core of the matplotlib library and provides dataset-oriented APIs.

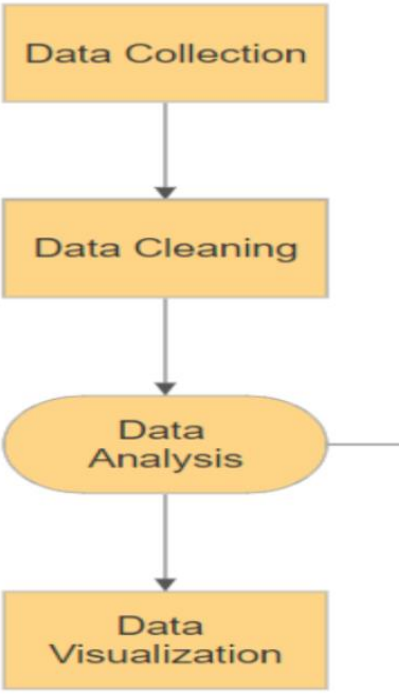
NumPy:

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices.

Tableau:

Tableau is a leading data visualization tool used for data analysis and business intelligence. Gartner's Magic Quadrant classified Tableau as a leader for analytics and business intelligence.

Methodology:



Data Collection:

Data Collection is a process of collecting information from relevant and reliable sources to find answers to research questions and evaluate outcomes.

Data Cleaning:

Data Cleaning is the process of fixing or removing incorrect, corrupt, poorly formatted, duplicated, or incomplete data within a dataset. The datasets used for this project are cleaned before any kind of data analysis is performed to ensure results obtained are accurate.

Data Analysis:

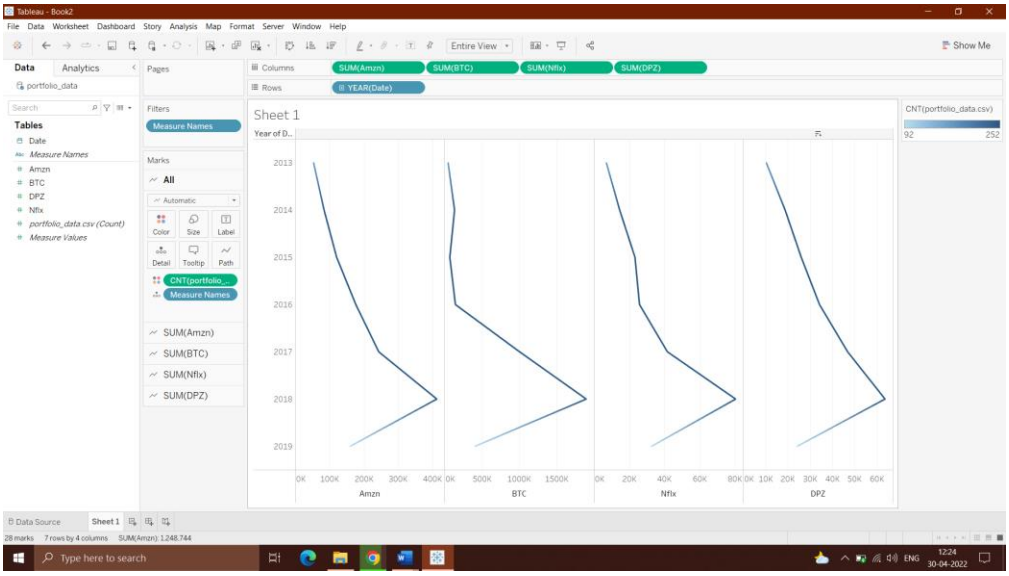
Data Analysis is a process of inspecting, modifying and modeling data with the goal of obtaining useful information, conclusions and support in decision making. Here, the data analysis is done by applying various Exploratory Data Analysis techniques in python to analyze the data and their relationship with each attribute.

Data Visualization:

Data Visualization is the process that deals with the graphical representation of data which simplifies the complex numerical data to a simple pictorial representation that is easy to understand. Here, the data visualization is also done through Python and the charts obtained are posted in this paper.

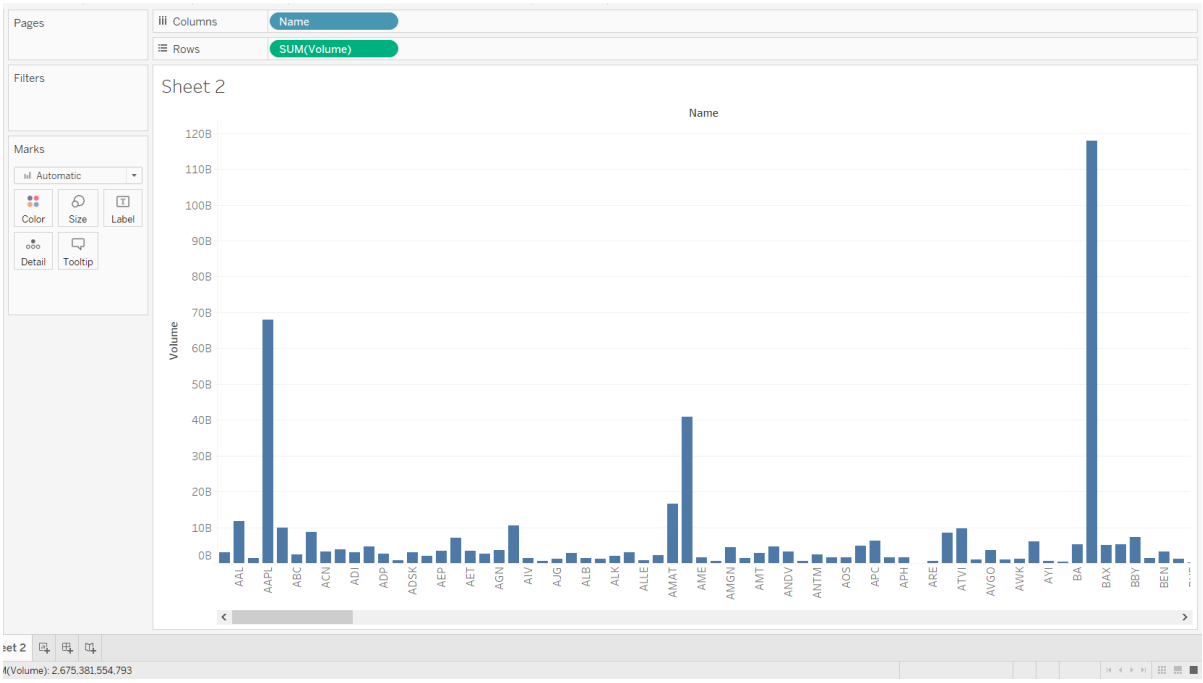
List of Chart:

This line chart shows increase of stock price over year(2013-2019)

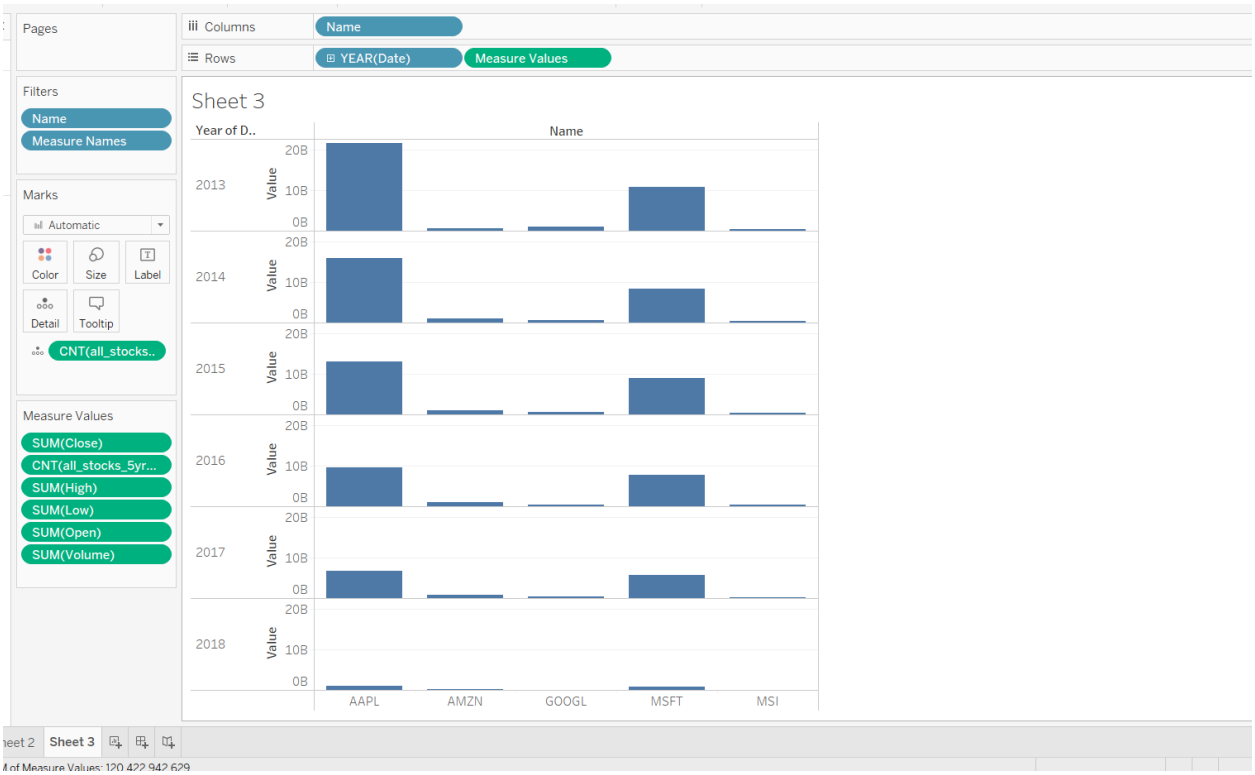


This scatter plot show dividend vs earning of company which show that if company earning in increasing the they are giving more dividend.

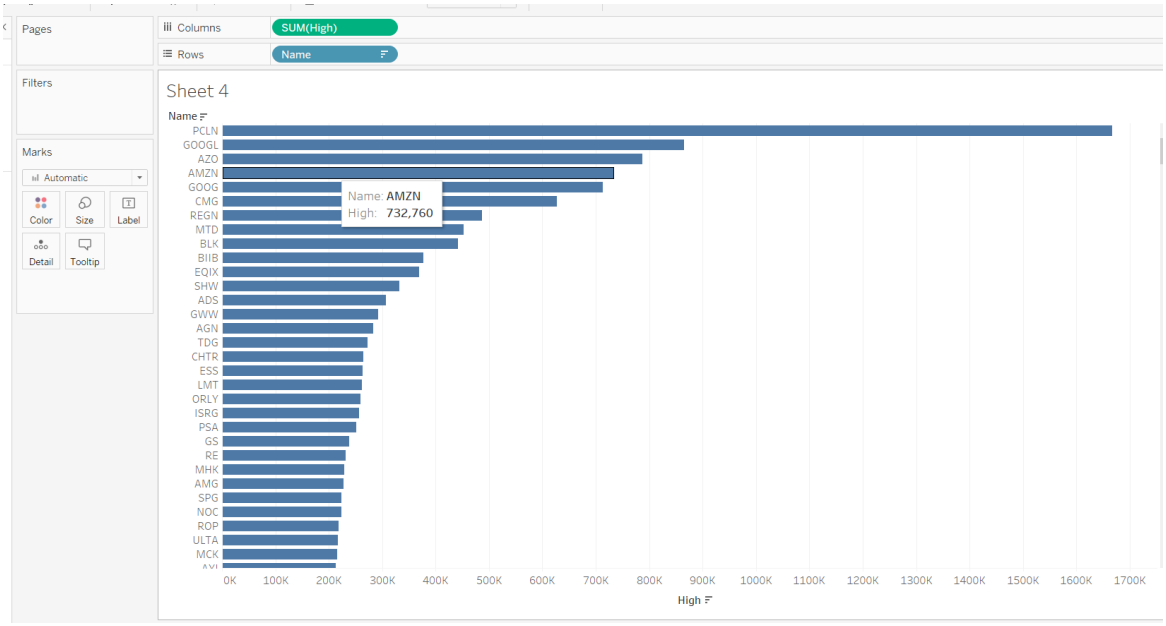
This bar graph shows that company volume



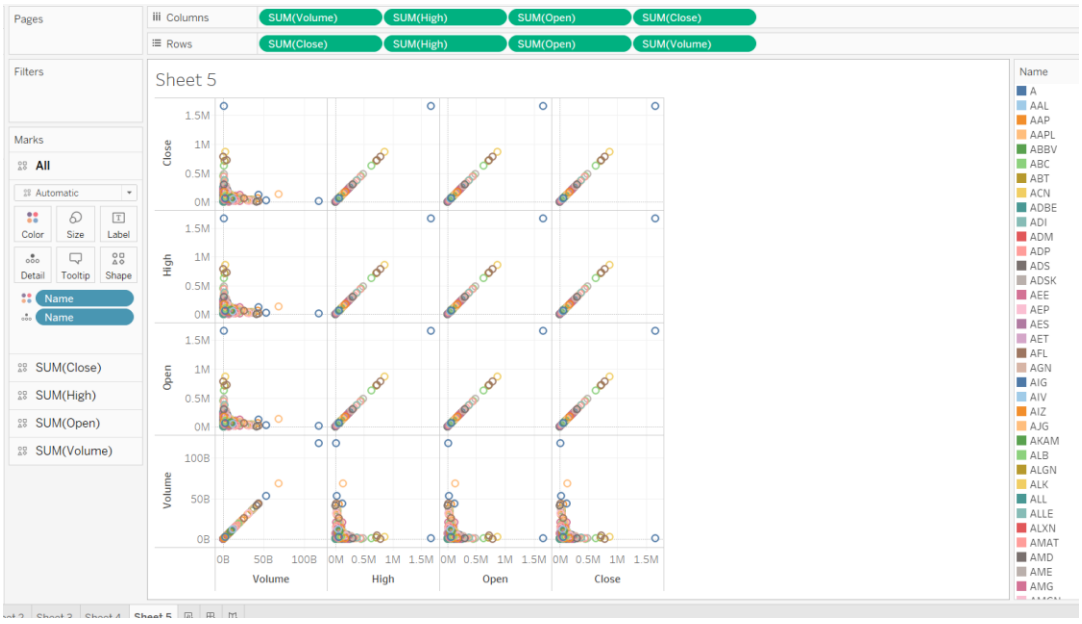
This table shows Apple, Amazon, Google, Microsoft and MSI volume over year



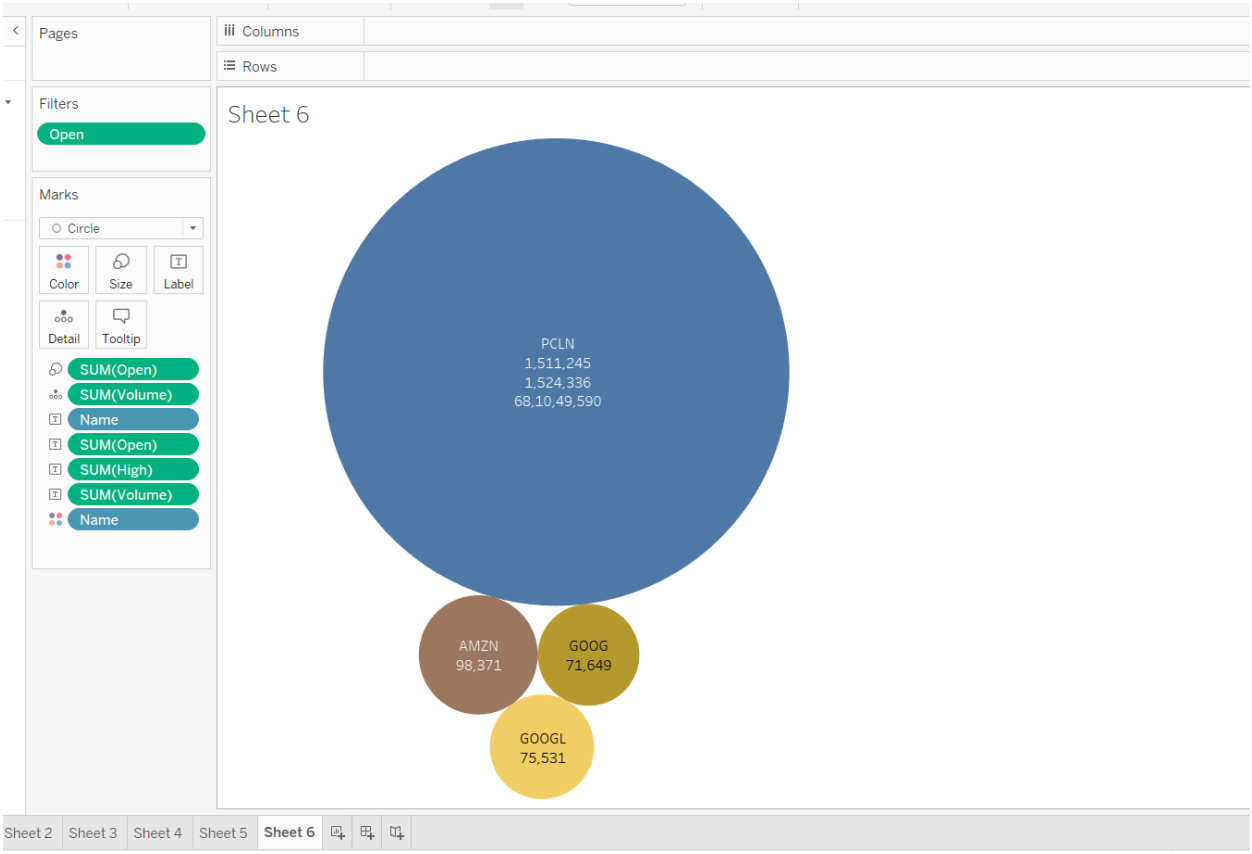
This graph shows which company is doing great and which company is in Top and making high return in investment to investers and help them to filter best company

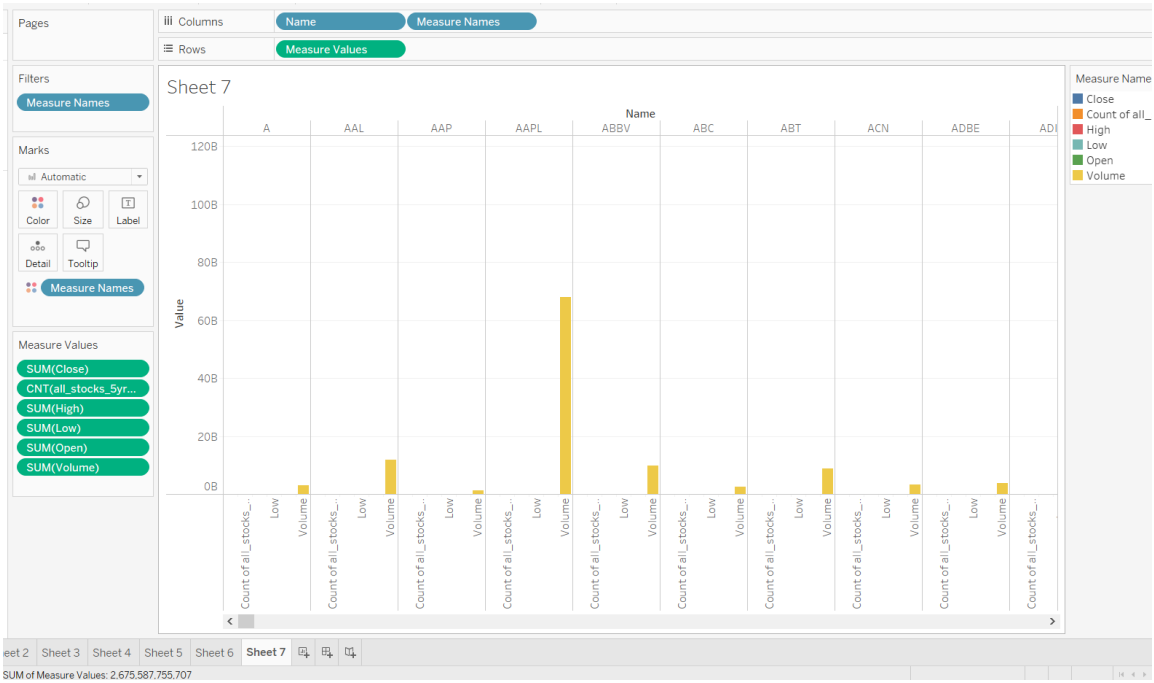


This show regression of stock data to predict relationship between number and predict which company is better.

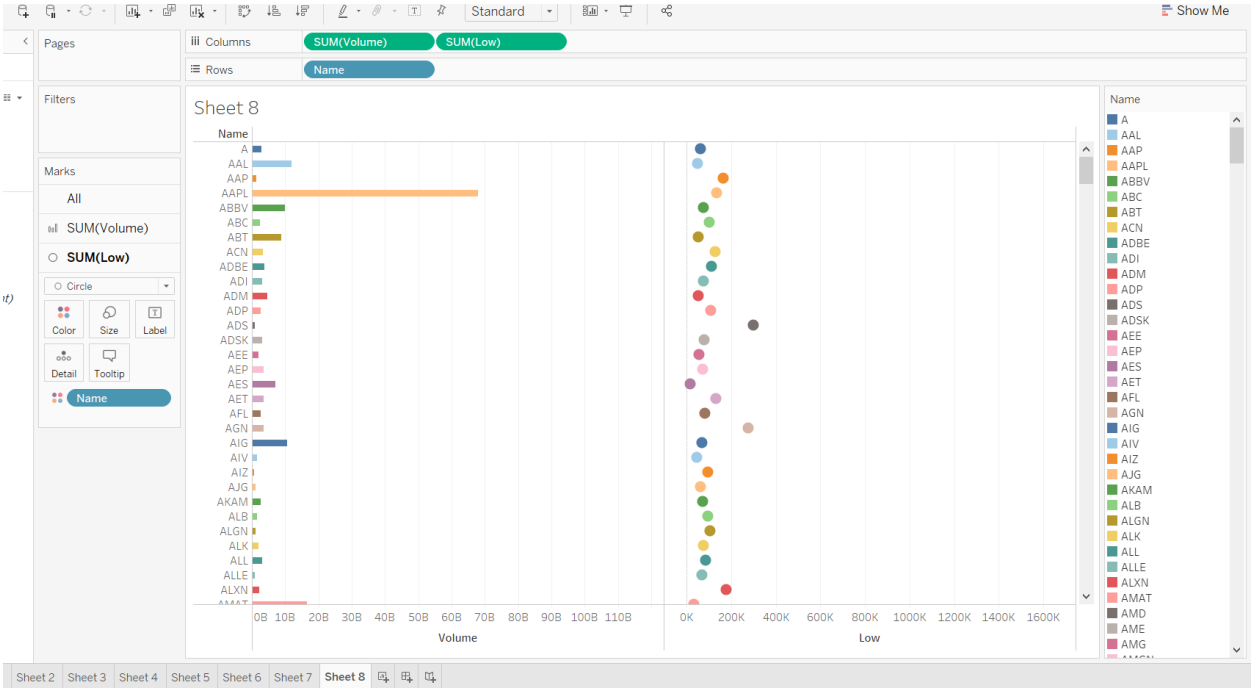


This buble chart shows highest opening compnay with their Details

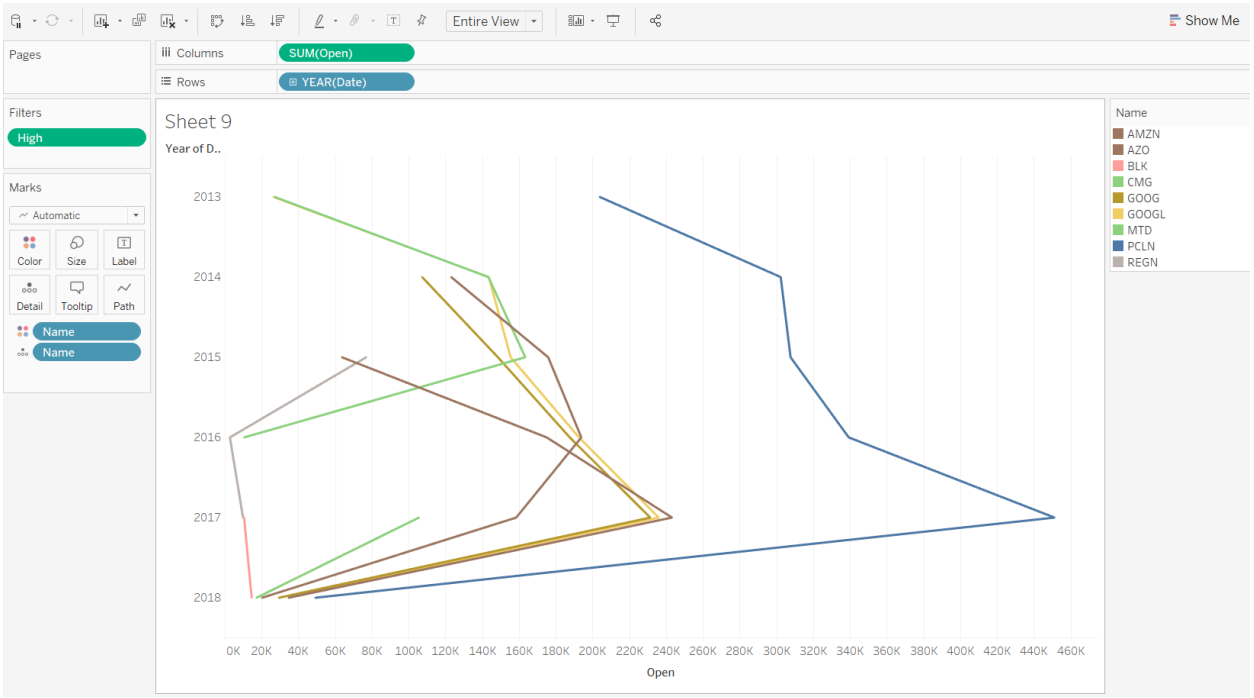




This show higher volume company sometime has low stock price



Top few company perfomace over few years



5. Results and Discussion:

We have discussed and explain many graph here and analyzed. we analyzed using jupyter Notepad and pasted the code and graph at end of our report.

What was the change in price of the stock overtime?
this section we'll go over how to handle requesting stock information with pandas, and how to analyze basic attributes of a stock.

Insert first graph here(closing price graph)

Analysis of apple:
- The first 2 months the adj close was low between 120-130 and then there was a rise in in price to 150 from July to November.
- Reached peak of 180 on January and was consolidated at March
- There was a fake breakdown at February .
- Closing price increased over the year

Analysis of google :
- There was a increase from May till November and also reached its peak at a stock rise of 300
- consolidation till April but dropped down to 2400 at May
- Closing price increased over the years

Analysis of microsoft:
- First 2 months adj close was fluctuating between 240-260
- A increase till 300 on November reached its peak at 340 and was consolidate
- Closing price increased over the years

Analysis of amazon:
- Closing stock became less in 1 year
- Reached its peak at July and August
- Was consolidate from May to December
- Stocks dropped down on January

What was the moving average of the various stocks?
Moving Average:
A succession of averages derived from successive segments (typically of constant size and overlapping) of a series of values.

What is the correlation between different stocks?
Correlation is a statistical measure that expresses the extent to which two variables are linearly related.

We will compare the daily percentage returns of two stocks to see how closely they are related.
Below are the closing values of the companies using these values we will plot Scatter plot.
A scatter plot is a diagram where each value in the data set is represented by a dot.

6. Conclusion

We can easily understand and predict prices by the help of analysis by Tableau and python and we can understand daily returns of stock market by visualization and help us to know when it's going up and down and it also help in investment in stock and Mutual Funds. The visualization make easy to invest and understand the company investment and work.

7. References

[1] NIFTY-50 Stock Market Data (2000 - 2021) . (2021, May 1). kaggle. Retrieved April 29, 2022, from https://www.kaggle.com/datasets/rohanrao/nifty50-stock-market-data

[2] Stock Market Index Data | Macro Trends . (2012). macro trend. Retrieved April 29, 2022

[3] Project Jupyter . (2022). home. Retrieved April 29, 2022, from https://jupyter.org

[4] tablueas. (2012). A Guide to Tableau Uses for Big Data Analysis and Viz

[5] R. Saravanan and P. Sujatha, "A state of art techniques on machine learning algorithms: A perspective of supervised learning approaches in data classification," in 2018 Second International Conference on Intelligent Computing and Control Systems.

[6] https://thinkingneuron.com/predicting-stock-prices-using-deep-learning-lstm-model-in-python/#:~:text=%20Predicting%20stock

[7] https://towardsdatascience.com/correlation-coefficient-clearly-explained-f034d00b66ac#:~:text=%20What%20is%20the%20range%20of%20the%20correlation,data%20point%20deviates%20from%20the%20mean.%20More%20

[8] https://realpython.com/linear-regression-in-python/

[9] https://www.w3schools.com/python/python_ml_scatterplot.asp

[10] https://www.ck12.org/c/statistics/scatter-plots-and-linear-correlation/lesson/Use-a-Scatterplot-to-Interpret-Data-MSM8/#:~:text=A%20trend%20line%20is%20a,trend%20or%20a%20negative%20tren.

[11] https://chartio.com/learn/charts/what-is-a-scatter-plot/

Code and Output Snapshots:

In [22]:
```
import pandas_datareader.data as web
```

In [24]:
```
import pip
pip.main(['install', 'pandas_datareader'])
```

In [25]:
```
!pip install -q yfinance
```

In [26]:
```
import pandas as pd
import numpy as np
```

In [27]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [28]:
```
sns.set_style('whitegrid')
plt.style.use("fivethirtyeight")
%matplotlib inline
import yfinance as yf
```

In [29]:
```
from datetime import datetime
```

In [30]:
```
tech_list = ['AAPL', 'GOOG', 'MSFT', 'AMZN']
tech_list = ['AAPL', 'GOOG', 'MSFT', 'AMZN']
end = datetime.now()
start = datetime(end.year - 1, end.month, end.day)

for stock in tech_list:
    globals()[stock] = yf.download(stock, start, end)
```

```
[**********************100%**********************]  1 of 1 completed
[**********************100%**********************]  1 of 1 completed
[**********************100%**********************]  1 of 1 completed
[**********************100%**********************]  1 of 1 completed
```

In [31]:
```
company_list = [AAPL, GOOG, MSFT, AMZN]
company_name = ["APPLE", "GOOGLE", "MICROSOFT", "AMAZON"]

for company, com_name in zip(company_list, company_name):
    company["company_name"] = com_name

df = pd.concat(company_list, axis=0)
df.tail(10000)
```

Out[31]:

Date	Open	High	Low	Close	Adj Close	Volume	company_nan
2021-04-22	133.039993	134.149994	131.410004	131.940002	131.160461	84566500	APP
2021-04-23	132.160004	135.119995	132.160004	134.320007	133.526382	78657500	APP
2021-04-26	134.830002	135.059998	133.559998	134.720001	133.924026	66905100	APP
2021-04-27	135.009995	135.410004	134.110001	134.389999	133.595963	66015800	APP

In [32]:
```
AAPL.describe()
```

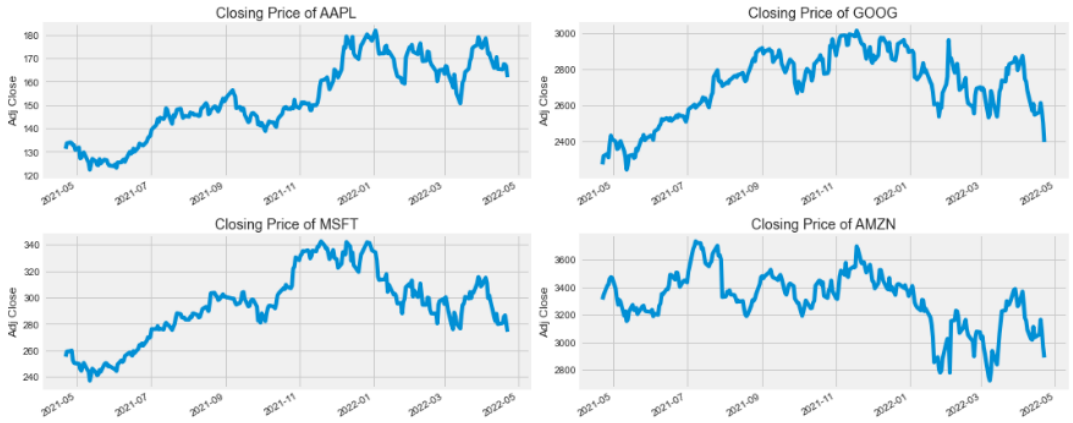
Out[32]:

	Open	High	Low	Close	Adj Close	Volume
count	254.000000	254.000000	254.000000	254.000000	254.000000	2.540000e+02
mean	152.721181	154.408111	151.213818	152.875276	152.546665	8.631972e+07
std	16.040420	16.381336	15.630090	16.028951	16.209690	2.547439e+07
min	123.160004	124.639999	122.250000	122.769997	122.251930	4.100000e+07
25%	143.234997	144.397503	141.720001	142.909996	142.520153	6.878902e+07
50%	149.880005	151.424995	148.724998	149.995003	149.695534	8.049350e+07
75%	167.457497	169.654995	165.384998	166.524998	166.524998	9.858352e+07
max	182.630005	182.940002	179.119995	182.009995	181.778397	1.954327e+08

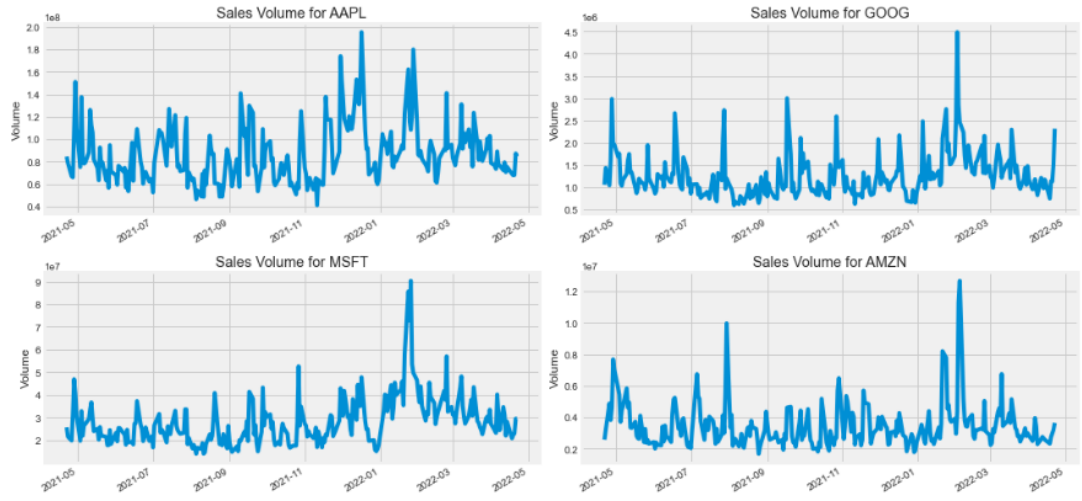
In [33]:
```
AAPL.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 254 entries, 2021-04-22 to 2022-04-22
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Open          254 non-null    float64
 1   High          254 non-null    float64
 2   Low           254 non-null    float64
 3   Close         254 non-null    float64
 4   Adj Close     254 non-null    float64
 5   Volume        254 non-null    int64
 6   company_name  254 non-null    object
dtypes: float64(5), int64(1), object(1)
memory usage: 15.9+ KB
```

In [34]:
```
plt.figure(figsize=(15, 6))
plt.subplots_adjust(top=1.25, bottom=1.2)

for i, company in enumerate(company_list, 1):
    plt.subplot(2, 2, i)
    company['Adj Close'].plot()
    plt.ylabel('Adj Close')
    plt.xlabel(None)
    plt.title(f"Closing Price of {tech_list[i - 1]}")

plt.tight_layout()
```



In [15]:
```
plt.figure(figsize=(15, 7))
plt.subplots_adjust(top=1.25, bottom=1.2)

for i, company in enumerate(company_list, 1):
    plt.subplot(2, 2, i)
    company['Volume'].plot()
    plt.ylabel('Volume')
    plt.xlabel(None)
    plt.title(f"Sales Volume for {tech_list[i - 1]}")

plt.tight_layout()
```

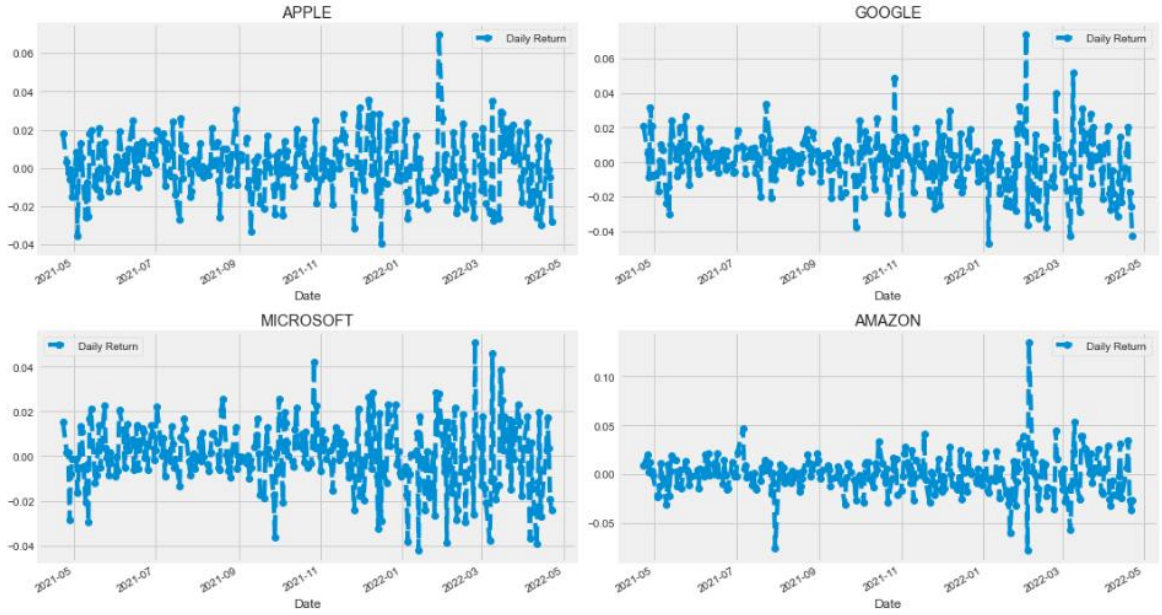


In [35]:
```
ma_day = [10, 20, 50]

for ma in ma_day:
    for company in company_list:
        column_name = f"MA for {ma} days"
        company[column_name] = company['Adj Close'].rolling(ma).mean()
```

In [26]:
```
fig, axes = plt.subplots(nrows=2, ncols=2)
fig.set_figheight(8)
fig.set_figwidth(15)

AAPL[['Adj Close', 'MA for 10 days', 'MA for 20 days', 'MA for 50 days']].plot(ax=axes[0,0])
axes[0,0].set_title('APPLE')

GOOG[['Adj Close', 'MA for 10 days', 'MA for 20 days', 'MA for 50 days']].plot(ax=axes[0,1])
axes[0,1].set_title('GOOGLE')

MSFT[['Adj Close', 'MA for 10 days', 'MA for 20 days', 'MA for 50 days']].plot(ax=axes[1,0])
axes[1,0].set_title('MICROSOFT')

AMZN[['Adj Close', 'MA for 10 days', 'MA for 20 days', 'MA for 50 days']].plot(ax=axes[1,1])
axes[1,1].set_title('AMAZON')

fig.tight_layout()
```

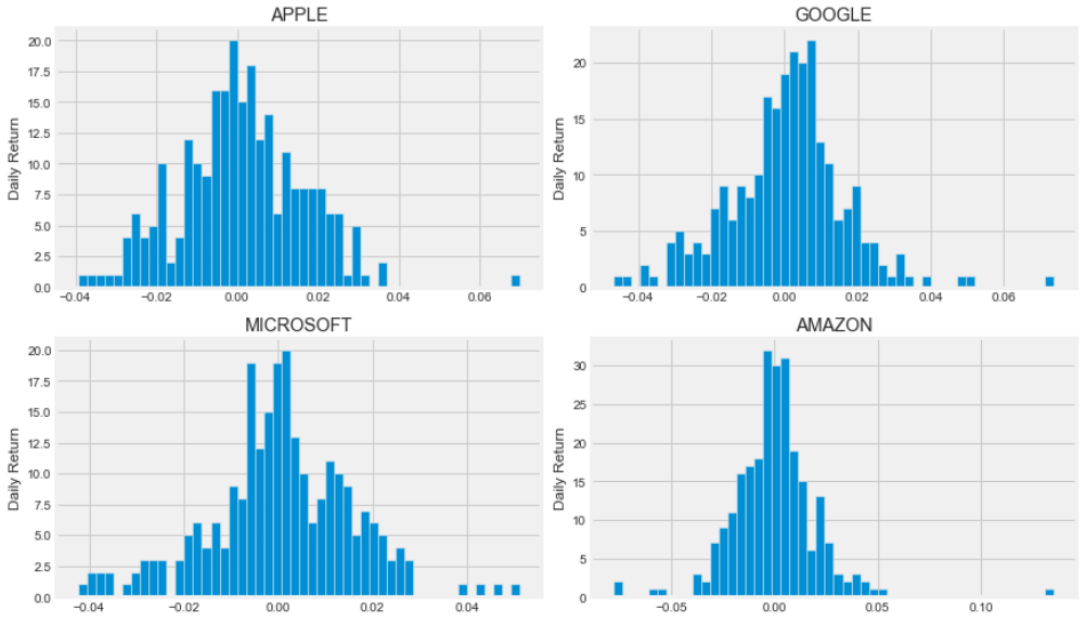
In [36]:

```
for company in company_list:
    company['Daily Return'] = company['Adj Close'].pct_change()

# Then we'll plot the daily return percentage
fig, axes = plt.subplots(nrows=2, ncols=2)
fig.set_figheight(8)
fig.set_figwidth(15)

AAPL['Daily Return'].plot(ax=axes[0,0], legend=True, linestyle='--', marker='o')
axes[0,0].set_title('APPLE')

GOOG['Daily Return'].plot(ax=axes[0,1], legend=True, linestyle='--', marker='o')
axes[0,1].set_title('GOOGLE')

MSFT['Daily Return'].plot(ax=axes[1,0], legend=True, linestyle='--', marker='o')
axes[1,0].set_title('MICROSOFT')

AMZN['Daily Return'].plot(ax=axes[1,1], legend=True, linestyle='--', marker='o')
axes[1,1].set_title('AMAZON')

fig.tight_layout()
```



In [28]:

```
plt.figure(figsize=(12, 7))

for i, company in enumerate(company_list, 1):
    plt.subplot(2, 2, i)
    company['Daily Return'].hist(bins=50)
    plt.ylabel('Daily Return')
    plt.title(f'{company_name[i - 1]}')

plt.tight_layout()
```



In [46]:
```
start_date = "2021-04-22"
end_date = "2022-04-22"
```

In [49]:
```
#closing_df = Datareader(tech_list,start, end)['Adj Close']
#closing_df.head()
closing_df = web.DataReader(tech_list, data_source='yahoo', start=start_date, end=end_date)['Adj Close']
print(closing_df)
```

```
Symbols            AAPL         GOOG        MSFT         AMZN
Date
2021-04-21  132.711227  2293.290039  258.474609  3362.020020
2021-04-22  131.160461  2267.919922  255.092163  3309.040039
2021-04-23  133.526398  2315.300049  259.039978  3340.879883
2021-04-26  133.924042  2326.739990  259.436737  3409.000000
2021-04-27  133.595963  2307.120117  259.853394  3417.429932
...                ...          ...         ...          ...
2022-04-18  165.070007  2559.219971  280.519989  3055.699951
2022-04-19  167.399994  2610.620117  285.299988  3162.310059
2022-04-20  167.229996  2564.909912  286.359985  3079.959961
2022-04-21  166.419998  2498.750000  280.809998  2965.919922
2022-04-22  161.789993  2392.280029  274.029999  2887.000000

[255 rows x 4 columns]
```

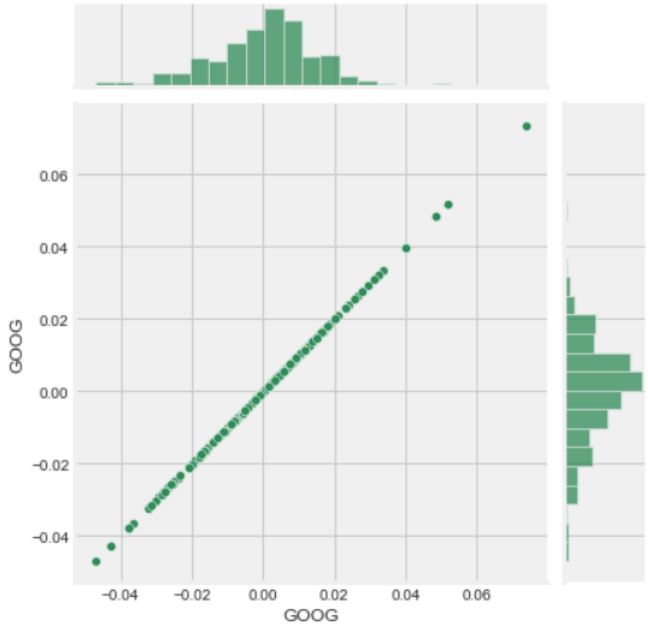
In [50]:
```
tech_rets = closing_df.pct_change()
tech_rets.head()
```

Out[50]:

Symbols	AAPL	GOOG	MSFT	AMZN
Date				
2021-04-21	NaN	NaN	NaN	NaN
2021-04-22	-0.011685	-0.011063	-0.013086	-0.015758
2021-04-23	0.018038	0.020891	0.015476	0.009622
2021-04-26	0.002978	0.004941	0.001532	0.020390

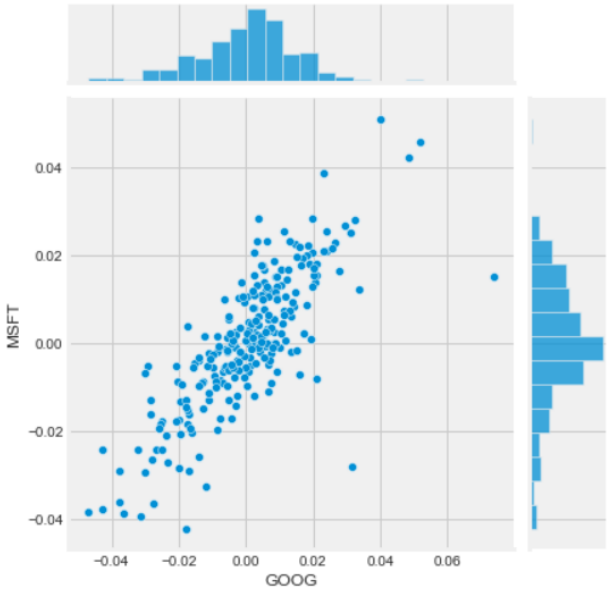
In [51]:
```
sns.jointplot(x='GOOG', y='GOOG', data=tech_rets, kind='scatter', color='seagreen')
```

Out[51]:
```
<seaborn.axisgrid.JointGrid at 0x1d5fb4a5e80>
```



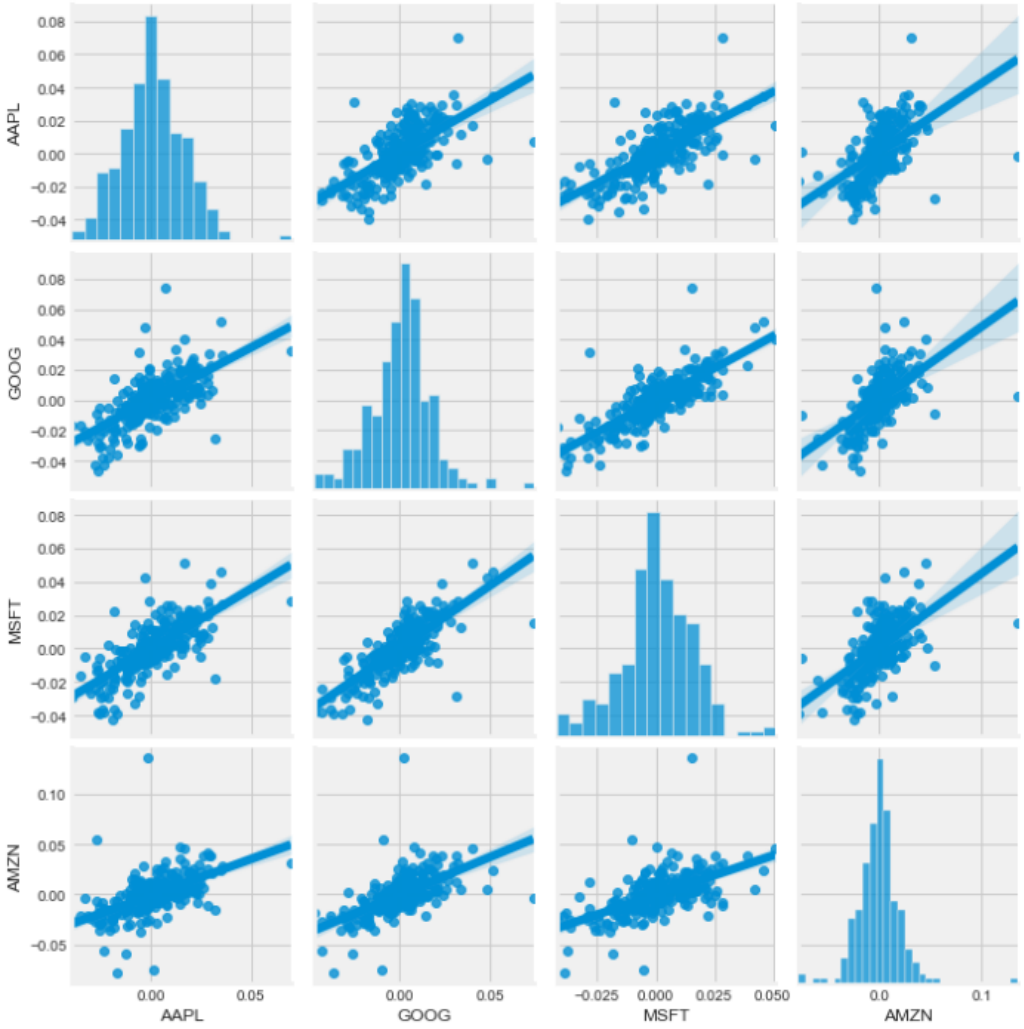
In [52]:
```
sns.jointplot(x='GOOG', y='MSFT', data=tech_rets, kind='scatter')
```

Out[52]:
```
<seaborn.axisgrid.JointGrid at 0x1d5fb609520>
```



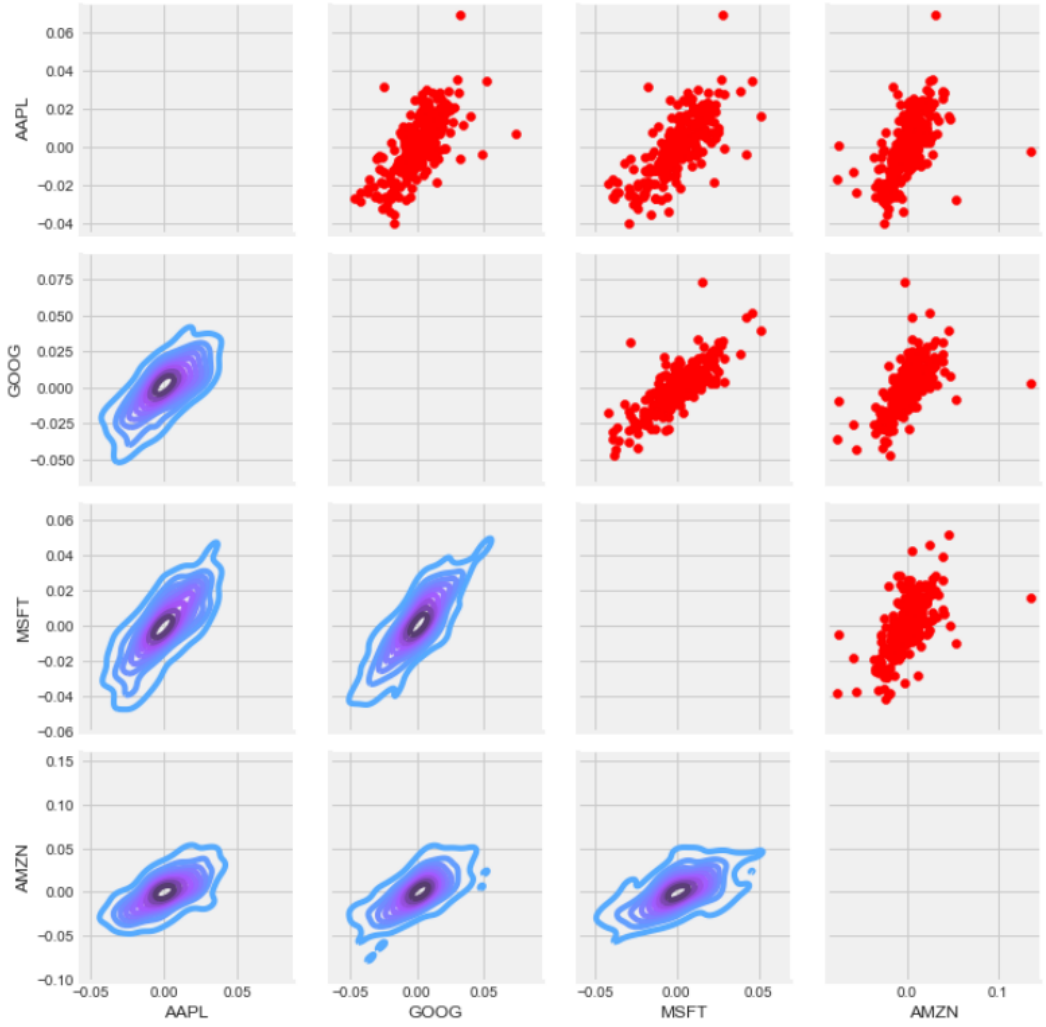
In [53]:
```
sns.pairplot(tech_rets, kind='reg')
```

Out[53]:
```
<seaborn.axisgrid.PairGrid at 0x1d5fb8e6670>
```



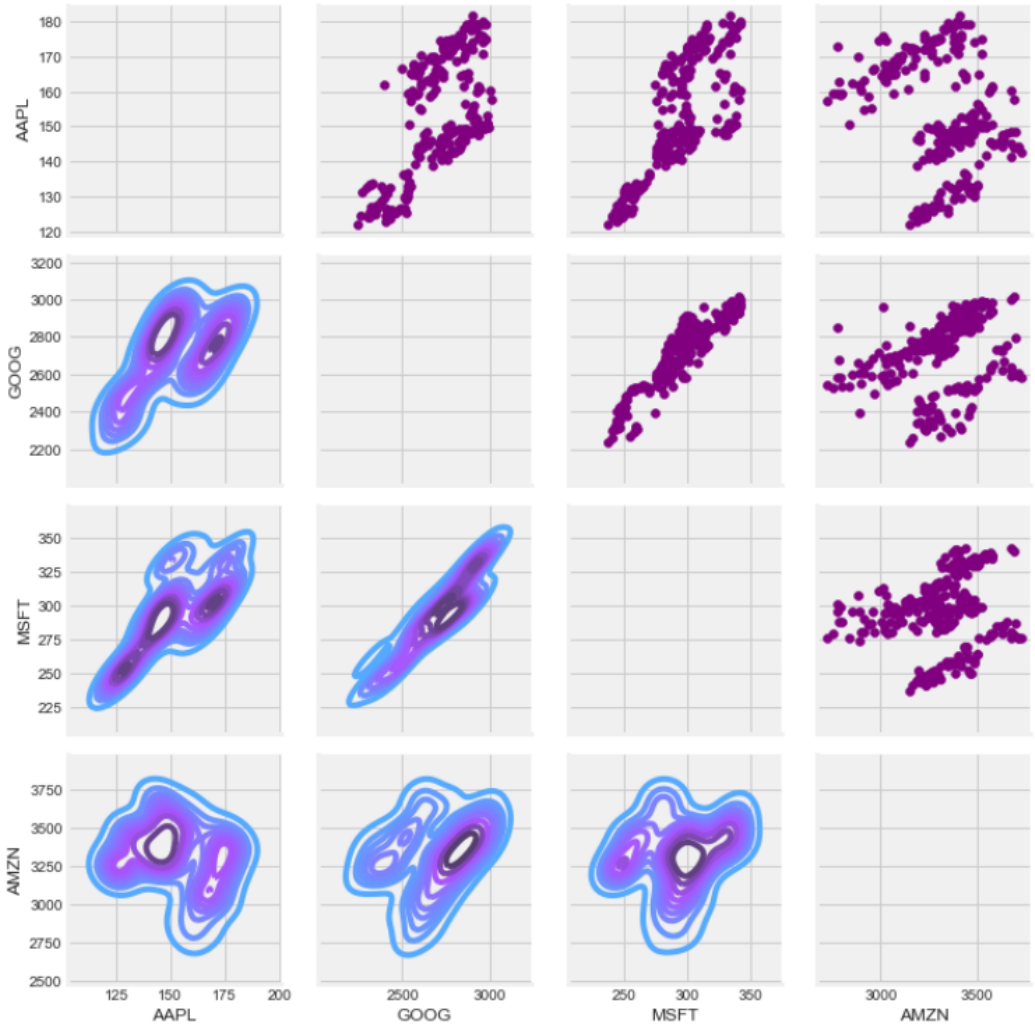
In [56]:
```
# Set up our figure by naming it returns_fig, call PairPlot on the DataFrame
return_fig = sns.PairGrid(tech_rets.dropna())

# Using map_upper we can specify what the upper triangle will look like.
return_fig.map_upper(plt.scatter, color='red')

# We can also define the lower triangle in the figure, inclufing the plot ty
# or the color map (BluePurple)
return_fig.map_lower(sns.kdeplot, cmap='cool_d')

# Finally we'll define the diagonal as a series of histogram plots of the da
return_fig.map_diag(plt.hist, bins=30)
```

Out[56]: <seaborn.axisgrid.PairGrid at 0x1d5fe2ab280>



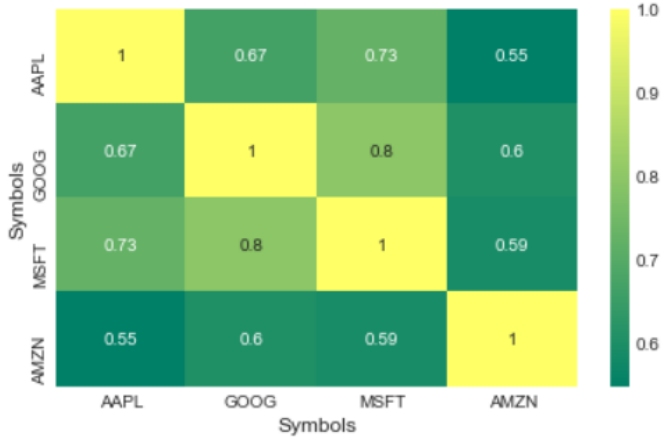
In [57]:
```
# Set up our figure by naming it returns_fig, call PairPLot on the DataFrame
returns_fig = sns.PairGrid(closing_df)

# Using map_upper we can specify what the upper triangle will look like.
returns_fig.map_upper(plt.scatter,color='purple')

# We can also define the lower triangle in the figure, inclufing the plot type (kde) c
returns_fig.map_lower(sns.kdeplot,cmap='cool_d')

# Finally we'll define the diagonal as a series of histogram plots of the daily return
returns_fig.map_diag(plt.hist,bins=30)
```

Out[57]: <seaborn.axisgrid.PairGrid at 0x1d580126880>



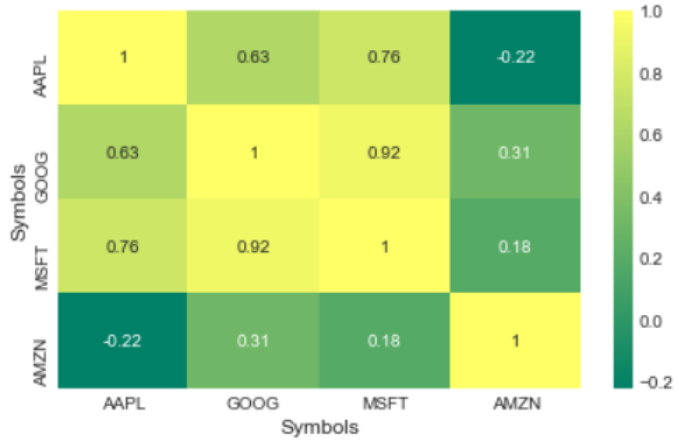
In [58]:
```
# Let's go ahead and use sebron for a quick correlation plot for the daily returns
sns.heatmap(tech_rets.corr(), annot=True, cmap='summer')
```

Out[58]: <AxesSubplot:xlabel='Symbols', ylabel='Symbols'>

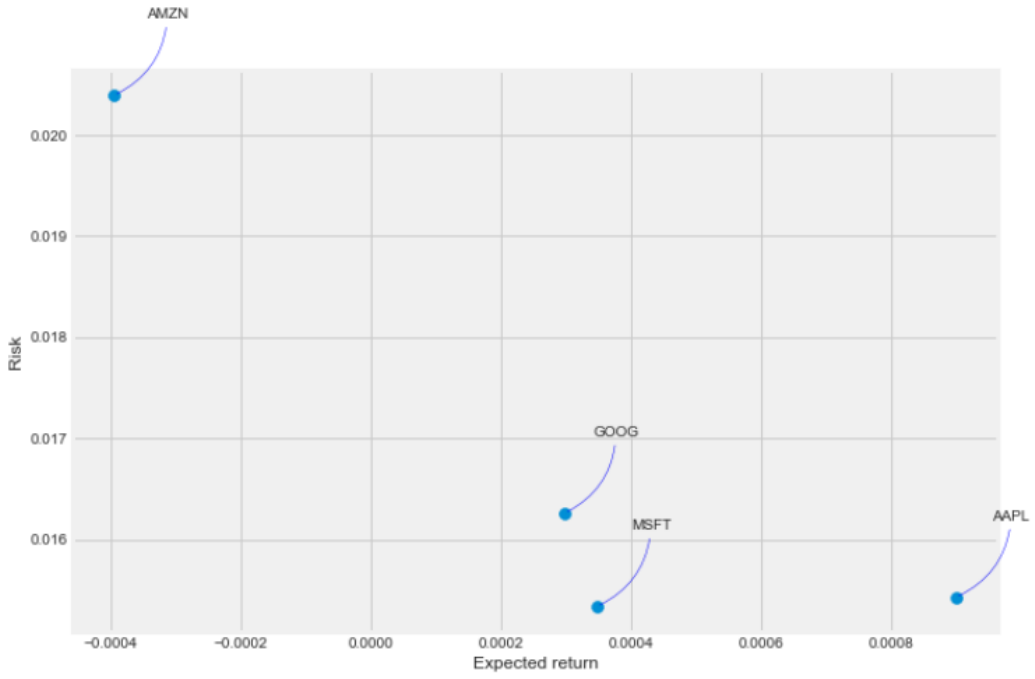


In [59]:
```
sns.heatmap(closing_df.corr(), annot=True, cmap='summer')
```

Out[59]: <AxesSubplot:xlabel='Symbols', ylabel='Symbols'>



In [60]:
```
# Let's start by defining a new DataFrame as a clenaed version of the oriignal tec
rets = tech_rets.dropna()

area = np.pi * 20

plt.figure(figsize=(10, 7))
plt.scatter(rets.mean(), rets.std(), s=area)
plt.xlabel('Expected return')
plt.ylabel('Risk')

for label, x, y in zip(rets.columns, rets.mean(), rets.std()):
    plt.annotate(label, xy=(x, y), xytext=(50, 50), textcoords='offset points', ha
                 arrowprops=dict(arrowstyle='-', color='blue', connectionstyle='ar
```



In [62]:
```
# Get the stock quote
df = web.DataReader('AAPL', data_source='yahoo', start='2012-01-01', end=datetime.
# Show teh data
df
```

Out[62]:

Date	High	Low	Open	Close	Volume	Adj Close
2012-01-03	14.732143	14.607143	14.621429	14.686786	302220800.0	12.575918
2012-01-04	14.810000	14.617143	14.642857	14.765714	260022000.0	12.643498
2012-01-05	14.948214	14.738214	14.819643	14.929643	271269600.0	12.783869
2012-01-06	15.098214	14.972143	14.991786	15.085714	318292800.0	12.917507
2012-01-09	15.276786	15.048214	15.196429	15.061786	394024400.0	12.897017
...	...	...	...	...	...	...
2022-04-18	166.600006	163.570007	163.919998	165.070007	69023900.0	165.070007
2022-04-19	167.820007	163.910004	165.020004	167.399994	67723800.0	167.399994
2022-04-20	168.880005	166.100006	168.759995	167.229996	67929800.0	167.229996
2022-04-21	171.529999	165.910004	168.910004	166.419998	87227800.0	166.419998
2022-04-22	167.869995	161.500000	166.460007	161.789993	84775200.0	161.789993

2594 rows × 6 columns

In [63]:
```
plt.figure(figsize=(16,6))
plt.title('Close Price History')
plt.plot(df['Close'])
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.show()
```

In [64]:
```
# Create a new dataframe with only the 'Close column
data = df.filter(['Close'])
# Convert the dataframe to a numpy array
dataset = data.values
# Get the number of rows to train the model on
training_data_len = int(np.ceil( len(dataset) * .95 ))

training_data_len
```

Out[64]:
```
2465
```

In [65]:
```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(dataset)

scaled_data
```

Out[65]:
```
array([[0.00439887],
       [0.00486851],
       [0.00584391],
       ...,
       [0.91205653],
       [0.90723691],
       [0.8796876 ]])
```

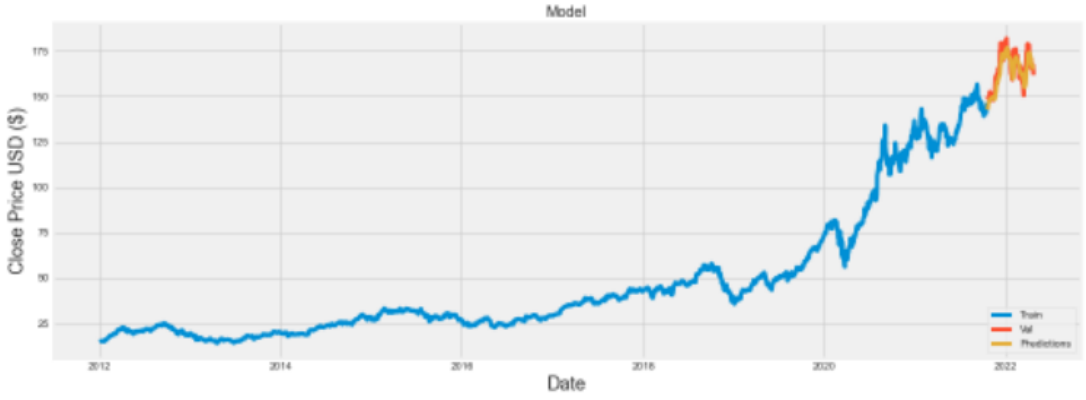
In [66]:
```
train_data = scaled_data[0:int(training_data_len), :]
# Split the data into x_train and y_train data sets
x_train = []
y_train = []

for i in range(60, len(train_data)):
    x_train.append(train_data[i-60:i, 0])
    y_train.append(train_data[i, 0])
    if i<= 61:
        print(x_train)
        print(y_train)
        print()

# Convert the x_train and y_train to numpy arrays
x_train, y_train = np.array(x_train), np.array(y_train)

# Reshape the data
x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))
```

```
[array([0.00439887, 0.00486851, 0.00584391, 0.00677256, 0.00663019,
       0.00695107, 0.00680444, 0.00655793, 0.00622217, 0.00726133,
       0.00819848, 0.00790947, 0.0063263 , 0.00783722, 0.00634968,
       0.01192796, 0.01149658, 0.01205972, 0.01327737, 0.01401476,
       0.01395314, 0.01372576, 0.01469479, 0.01560643, 0.01663922,
       0.01830739, 0.02181161, 0.02186474, 0.02381555, 0.02527333,
       0.0227679 , 0.02373267, 0.02371354, 0.02641875, 0.02603411,
       0.026746  , 0.02802528, 0.02873719, 0.03078787, 0.03228178,
       0.03271317, 0.03286405, 0.03030973, 0.02969346, 0.02978484,
       0.03218616, 0.03286193, 0.03431335, 0.03773469, 0.04229932,
       0.04144504, 0.04144716, 0.04474738, 0.04578017, 0.04504489,
       0.04437338, 0.04367423, 0.04599691, 0.04759072, 0.04825798])]
[0.04660893460974819]
```

In [80]:
```
from keras.models import Sequential
from keras.layers import Dense, LSTM

# Build the LSTM model
model = Sequential()
model.add(LSTM(128, return_sequences=True, input_shape= (x_train.shape[1], 1)))
model.add(LSTM(64, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')

# Train the model
model.fit(x_train, y_train, batch_size=1, epochs=1)
```

```
2405/2405 [==============================] - 69s 27ms/step - loss: 0.0011
```

In [82]:
```
train = data[:training_data_len]
valid = data[training_data_len:]
valid['Predictions'] = predictions
# Visualize the data
plt.figure(figsize=(16,6))
plt.title('Model')
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.plot(train['Close'])
plt.plot(valid[['Close', 'Predictions']])
plt.legend(['Train', 'Val', 'Predictions'], loc='lower right')
plt.show()
```



In [83]:
```
valid
```

Out[83]:

Date	Close	Predictions
2021-10-19	148.759995	142.702927
2021-10-20	149.259995	143.846283
2021-10-21	149.479996	145.065018
2021-10-22	148.690002	146.175201
2021-10-25	148.639999	146.950226
...	...	...
2022-04-18	165.070007	166.712967
2022-04-19	167.399994	165.749588
2022-04-20	167.229996	165.277542
2022-04-21	166.419998	165.071182
2022-04-22	161.789993	164.915405

129 rows × 2 columns