Documentation

Group : MIT-TEAM1
Project Title: OHLC ENGINE
Group Members:

Sr No	Name of Student	E-mail Address
1	Shivani Santosh Sorte	sssorte1428@gmail.com
2	Neha Dutta	duttaneha05@gmail.com
3	Aaradhya Raj	aaradhyadiscovery@gmail.com
4	Mayank Bhardwaj	mayankbhardwaj2703@gmail.com
5	Shivani Thombre	tshivani270@gmail.com

Problem Statement:

Create a Dashboard based on which User would be able to analyze the sentiment of the specific stock. Creation of an Analytical Server "OHLC " (Open/High/Low/Close) time series based on the ‘Stock List’ dataset present in JSON format and its output shown through timely reports printed in Charts.

Input

The input used is the ‘Stock List’ dataset present in JSON format which comprises certain attributes. Following are the associated meanings of some of the significant attributes.

```
struct OHLC{
        open : opening price Double
        high : highest price Double
        low : lowest price Double
        close : closed price Double
        date : date of transaction Date
        }
```

The above important attributes proved to have provided us with satisfactory results.

Language used

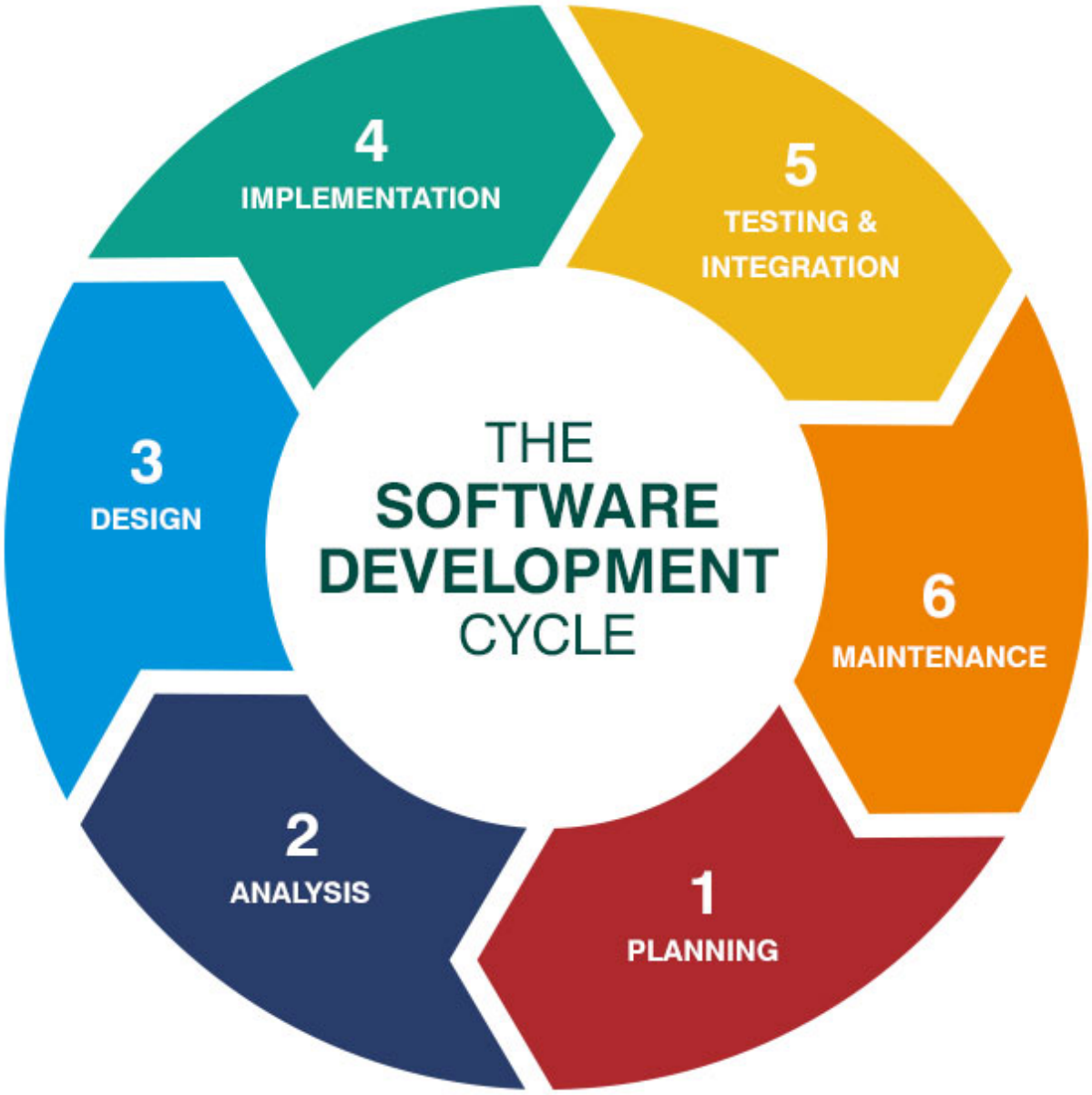
Python programming language has been used by the team to implement the solution for the problem statement.

Why Python?

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse.

Software Development Life Cycle:

Since building software is inherently complex and we were running short of time due to the strict close deadline, we decided to create the project with the well known and trusted approach, using Software Development Life Cycle.



In essence, a software development life cycle is a roadmap for working on a digital solution. In other words, SDLC is a blueprint designed for a team to create, maintain, and fix software products.

Following is a detailed summary of how the team functioned at every stage of the SDLC to obtain the final output.

1)Planning:

Mission - Process the problem statement
Approach - We read the problem statement carefully and researched the terms related to it. After introducing ourselves to the assigned teammates and finding out about each other's programming skills, and researching potential programming languages and packages that they provide we decided to choose **Python**.

2)Analysis:

Mission -Analysing the problem and planning the solution
Approach - We first analyzed the problem and understood the requirements properly. Following that, we listed out possible solutions for every sub-problem. We concluded this stage by selecting the best possible solution for each sub-problem.

3)Design:

Designing the solution and creating a blueprint of the solution

Design for the Dashboard

-Users can search for different companies' stock by clicking at the bar on top.
-Visualizations dropdown will show different options of graph which are (OHLC, CANDLE STICK, COLOURED BAR, VERTEX LINE)
-Start and end button is for selecting the date limit for the graph.
-At the bottom additional information of the company will be displayed.
-The graph is shown at the right side of the dashboard.

4)Implementation:

Mission - Implementing the solution formed
Approach -

- Since we had decided to use Python, we decided to convert the .json file to .csv format, so that the processing of data goes smoothly. We did so, by converting json to txt and txt to csv. Then we saved a copy of the csv file and performed processing on it.

  Code used

```
import json import csv import pandas as pd
f=open('Stock List.json') file=json.load(f)
file.close()
#convert file to text format…
read_file = pd.read_csv (r'jsonformatter.txt') read_file.to_csv (r'final.csv',
index=None)
pd.read_csv('final.csv')
```

- We did some data analyzation to understand the dataset properly

```
[2] import pandas as pd
```

```
[3] data = pd.read_csv('final.csv')
    data.head()
```

	close	high	low	open	symbol	volume	id	key	subkey	date	updated	changeOverTime	marketChangeOverTime	uOpen	uClose	uHigh	uLow	
0	129.41	133.6116	126.760	133.52	AAPL	143301887	HISTORICAL_PRICES	AAPL	NaN	2021-01-04	1628263122000	0.000000	0.000000	133.52	129.41	133.6116	126.760	14
1	131.01	131.7400	128.430	128.89	AAPL	97664898	HISTORICAL_PRICES	AAPL	NaN	2021-01-05	1628262969000	0.012364	0.012364	128.89	131.01	131.7400	128.430	9
2	126.60	131.0499	126.382	127.72	AAPL	155087970	HISTORICAL_PRICES	AAPL	NaN	2021-01-06	1628263183000	-0.021714	-0.021714	127.72	126.60	131.0499	126.382	15
3	130.92	131.6300	127.860	128.36	AAPL	109578157	HISTORICAL_PRICES	AAPL	NaN	2021-01-07	1628262969000	0.011668	0.011668	128.36	130.92	131.6300	127.860	10
4	132.05	132.6300	130.230	132.43	AAPL	105158245	HISTORICAL_PRICES	AAPL	NaN	2021-01-08	1628262978000	0.020400	0.020400	132.43	132.05	132.6300	130.230	10

```
[5]  data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16624 entries, 0 to 16623
Data columns (total 26 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   close                 16624 non-null  float64
 1   high                  16624 non-null  float64
 2   low                   16624 non-null  float64
 3   open                  16624 non-null  float64
 4   symbol                16624 non-null  object
 5   volume                16624 non-null  int64
 6   id                    16624 non-null  object
 7   key                   16624 non-null  object
 8   subkey                0 non-null      float64
 9   date                  16624 non-null  object
 10  updated               16624 non-null  int64
 11  changeOverTime        16624 non-null  float64
 12  marketChangeOverTime  16624 non-null  float64
 13  uOpen                 16624 non-null  float64
 14  uClose                16624 non-null  float64
 15  uHigh                 16624 non-null  float64
 16  uLow                  16624 non-null  float64
 17  uVolume               16624 non-null  int64
 18  fOpen                 16624 non-null  float64
 19  fClose                16624 non-null  float64
 20  fHigh                 16624 non-null  float64
 21  fLow                  16624 non-null  float64
 22  fVolume               16624 non-null  int64
 23  label                 16624 non-null  object
 24  change                16624 non-null  float64
 25  changePercent         16624 non-null  float64
dtypes: float64(17), int64(4), object(5)
memory usage: 3.3+ MB
```

```
[6]  data.describe()
```

	close	high	low	open	volume	subkey	updated	changeOverTime	marketChangeOverTime	uOpen	uClose	uHigh	uL
count	16624.000000	16624.000000	16624.000000	16624.000000	1.662400e+04	0.0	1.662400e+04	16624.000000	16624.000000	16624.000000	16624.000000	16624.000000	16624.0000
mean	284.686008	287.721300	281.412055	284.551123	9.560931e+06	NaN	1.624440e+12	0.111195	0.111195	288.279972	288.417803	291.508913	285.0800
std	464.555281	469.044273	459.668257	464.340235	1.493224e+07	NaN	4.834122e+09	0.181527	0.181527	465.160374	465.378195	469.891361	460.4603
min	27.430000	27.520000	15.320000	27.450000	9.310700e+04	NaN	1.609812e+12	-0.552165	-0.552165	27.450000	27.430000	27.520000	15.3200
25%	79.637500	80.443750	78.766250	79.642500	1.855725e+06	NaN	1.622247e+12	0.000947	0.000947	79.642500	79.637500	80.443750	78.7662
50%	147.646250	148.692500	146.110000	147.550000	4.356159e+06	NaN	1.625833e+12	0.080610	0.080610	149.050000	149.085000	150.475000	147.6800
75%	270.705000	273.812500	267.017500	269.957500	1.073591e+07	NaN	1.628475e+12	0.183150	0.183150	277.860000	277.690000	280.794975	274.5762
max	3731.410000	3773.078200	3696.792900	3744.000000	2.470524e+08	NaN	1.629959e+12	3.336078	3.336078	3744.000000	3731.410000	3773.078200	3696.7929

- Data Visualization

OHLC Chart

```
[39] df = pd.read_csv('final.csv')

     fig = go.Figure(data=go.Ohlc(x=df['date'],

                         open=df['open'],
                         high=df['high'],
                         low=df['low'],
                         close=df['close'],
                         increasing_line_color = 'red',
                         decreasing_line_color = 'black'))

     fig.update_layout(xaxis_rangeslider_visible = True)
     fig.show()
```

Candlestick Chart

```
[32] from datetime import datetime

     df = pd.read_csv('final.csv')

     fig = go.Figure(data=[go.Candlestick(x=df['date'],
                     open=df['open'],
                     high=df['high'],
                     low=df['low'],
                     close=df['close'],
                     increasing_line_color = 'red',
                     decreasing_line_color = 'black')])

     fig.update_layout(xaxis_rangeslider_visible = True)
     fig.show()
```

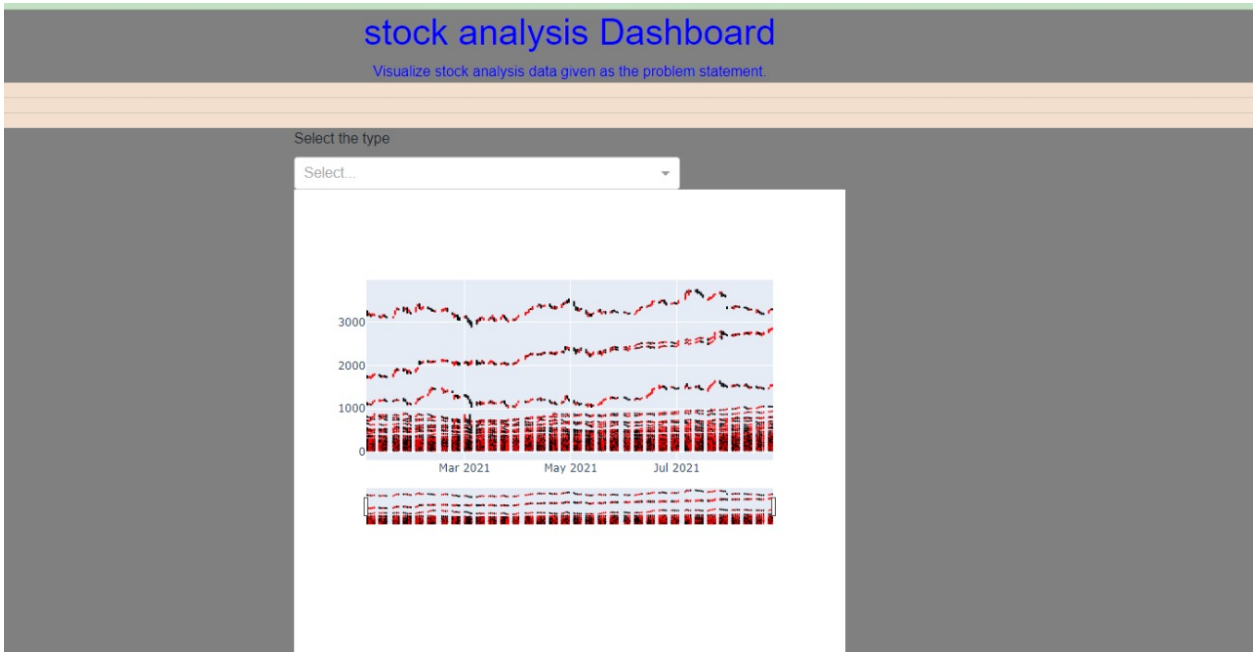
Vertex Line Chart & Colored Bar Chart

```
[60] mpf.plot(data,type='line', style='charles',volume=True)
```

Hollow Candle

To plot Hollow Candlestick we need to check whether it was up day or down day. For Up-Day open should be less than close. For Down-Day open should be more than close

- **Dashboard (Frontend)**
  Frontend created in the Flask environment using Python programming language.

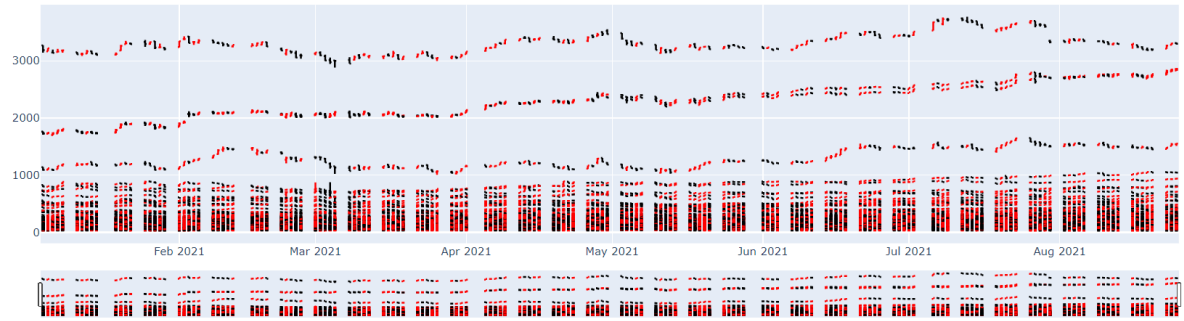


  Why Flask?
  Flask is an API of Python that allows us to build up web-applications. Flask's framework is more explicit than Django's framework and is also easier to learn because it has less base code to implement a simple web-Application.

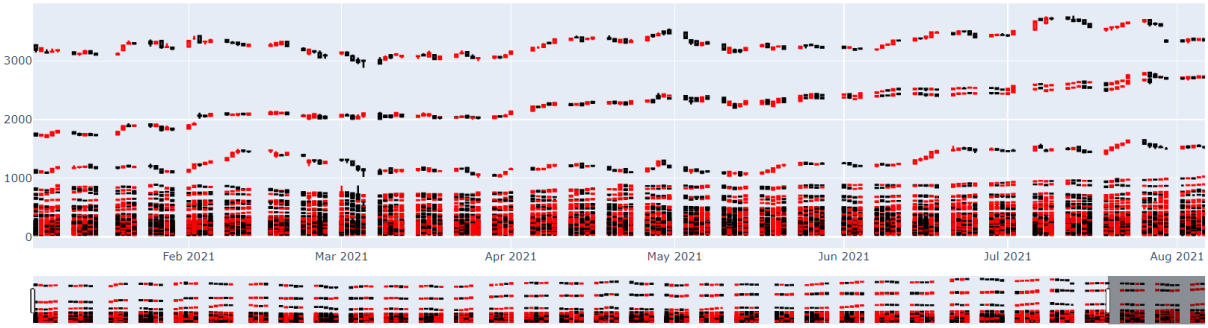
5)Testing and integration:

Testing the solution through visualization and exploring the dataset etc

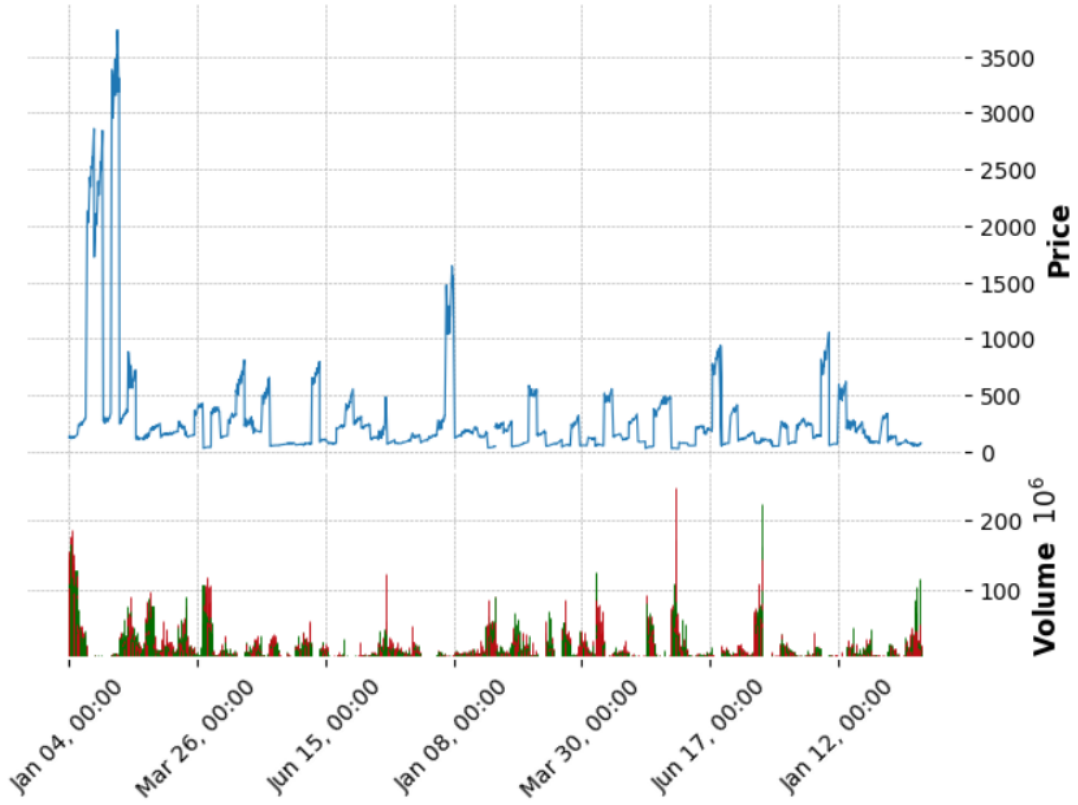
OHLC Chart



Candlestick Chart



Vertex Line Chart & Colored Bar Chart



Integration performed using Flask.

6)Maintenance:

Mission - Maintaining the project i.e. solving the errors and updating the code according to requirements

Approach -

We went through the project once again and noted all errors and places that require development. Our next step will be to solve them and do proper further development.

Following are some future possible development aspects -

- Adding options for the user to enter start and end date to view analytics during that time frame
- Adding functionality for the user to view search history and to let them click on the previously searched item to view appropriate results.
- Adding functionality for users to view company details based on the

‘symbol’ value .

Conclusion:

After working for 14 continuous hours and gaining abundant knowledge through extensive learning, our team achieved commendable results and experienced an immense sense of joy and satisfaction. We are grateful to the hackathon organizers for giving us this platform to showcase our skills and, an opportunity to connect, learn and grow.