Stock Price Data Analysis Of Banks

07/11/2024

BAC - BANK OF AMERICA STOCK PRICE HISTORY CHARTS

In [3]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [7]:
```python
bac=pd.read_csv('BAC.csv')
bac.head(10)
```

Out[7]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	03-01-2006	46.919998	47.180000	46.150002	47.080002	34.811729	16296700
1	04-01-2006	47.000000	47.240002	46.450001	46.580002	34.442013	17757900
2	05-01-2006	46.580002	46.830002	46.320000	46.639999	34.486385	14970700
3	06-01-2006	46.799999	46.910000	46.349998	46.570000	34.434616	12599800
4	09-01-2006	46.720001	46.970001	46.360001	46.599998	34.456806	15619400
5	10-01-2006	46.400002	46.509998	45.880001	46.209999	34.168419	15634600
6	11-01-2006	46.060001	46.250000	45.750000	46.099998	34.087093	14742100
7	12-01-2006	46.220001	46.230000	45.709999	45.799999	33.865276	10546600
8	13-01-2006	45.830002	46.000000	45.680000	45.799999	33.865276	10791000
9	17-01-2006	45.400002	45.580002	45.000000	45.310001	33.502972	14605900

In [13]:
```python
bac.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3523 entries, 0 to 3522
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       3523 non-null   object
 1   Open       3523 non-null   float64
 2   High       3523 non-null   float64
 3   Low        3523 non-null   float64
 4   Close      3523 non-null   float64
 5   Adj Close  3523 non-null   float64
 6   Volume     3523 non-null   int64
dtypes: float64(5), int64(1), object(1)
memory usage: 192.8+ KB
```

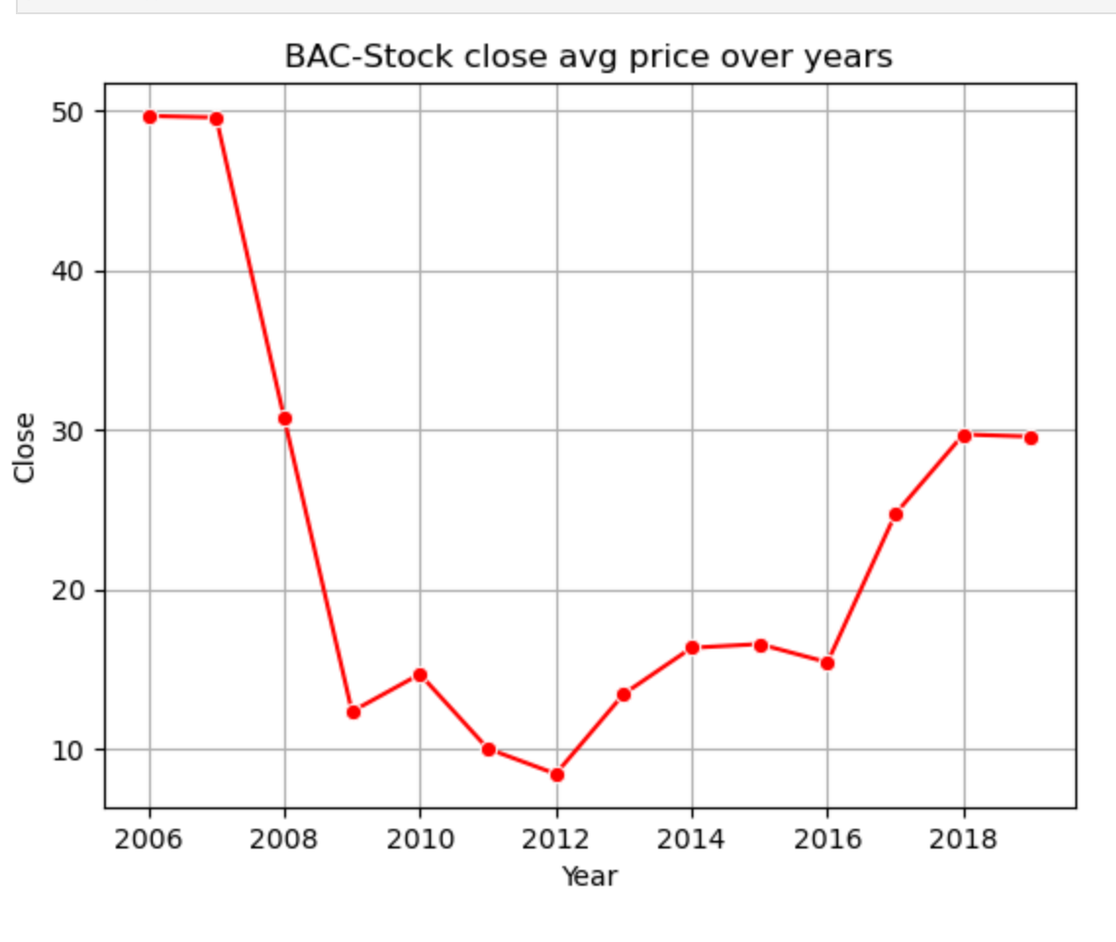
In [ ]:
```python
# Bac column is in object datatype that is string hence
# we are converting it into date
```

In [23]:
```python
bac['Date']=bac['Date'].astype('datetime64[ns]')
bac.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3523 entries, 0 to 3522
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       3523 non-null   datetime64[ns]
 1   Open       3523 non-null   float64
 2   High       3523 non-null   float64
 3   Low        3523 non-null   float64
 4   Close      3523 non-null   float64
 5   Adj Close  3523 non-null   float64
 6   Volume     3523 non-null   int64
dtypes: datetime64[ns](1), float64(5), int64(1)
memory usage: 192.8 KB
```

Line chart showing avg of closing price over the years

In [97]:
```python
bac['Year']=bac['Date'].dt.year

yearly_data=bac.groupby('Year')['Close'].mean().reset_index()

sns.lineplot(yearly_date,x='Year',y='Close',marker='o',color='red')
plt.title("BAC-Stock close avg price over years")
plt.grid(True)
plt.show()
```



Scatter plot showing relationship betwen open price and close price

In [123…
```python
sns.scatterplot(bac,x='Open',y='Close')
plt.title('Open price VS Close price')
plt.grid(True)
plt.show()
```

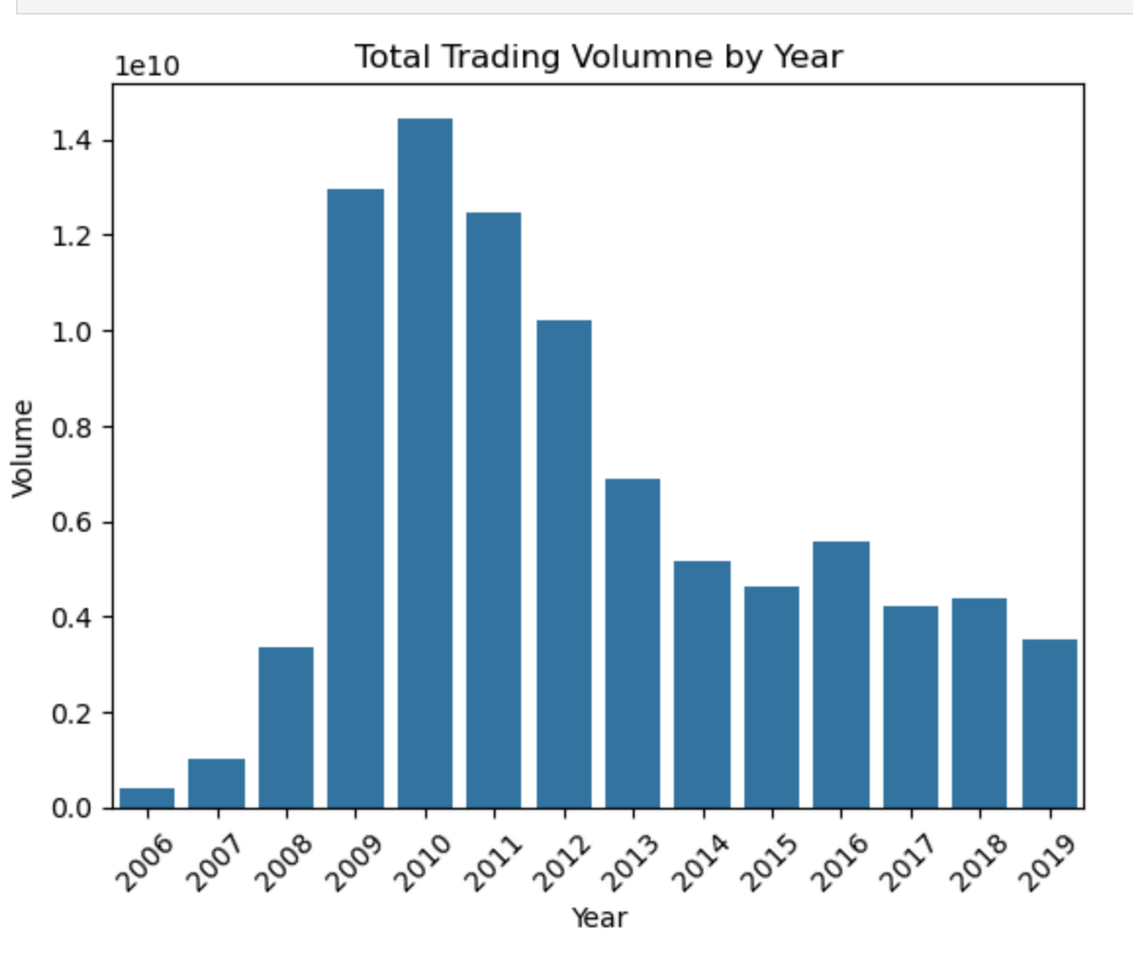
C-CITI-BANK STOCK PRICE HISTORY CHARTS

In [107…
```python
citi=pd.read_csv('c.csv')
citi['Date']=citi['Date'].astype('datetime64[ns]')
citi.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3523 entries, 0 to 3522
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       3523 non-null   datetime64[ns]
 1   Open       3523 non-null   float64
 2   High       3523 non-null   float64
 3   Low        3523 non-null   float64
 4   Close      3523 non-null   float64
 5   Adj Close  3523 non-null   float64
 6   Volume     3523 non-null   int64
dtypes: datetime64[ns](1), float64(5), int64(1)
memory usage: 192.8 KB
```

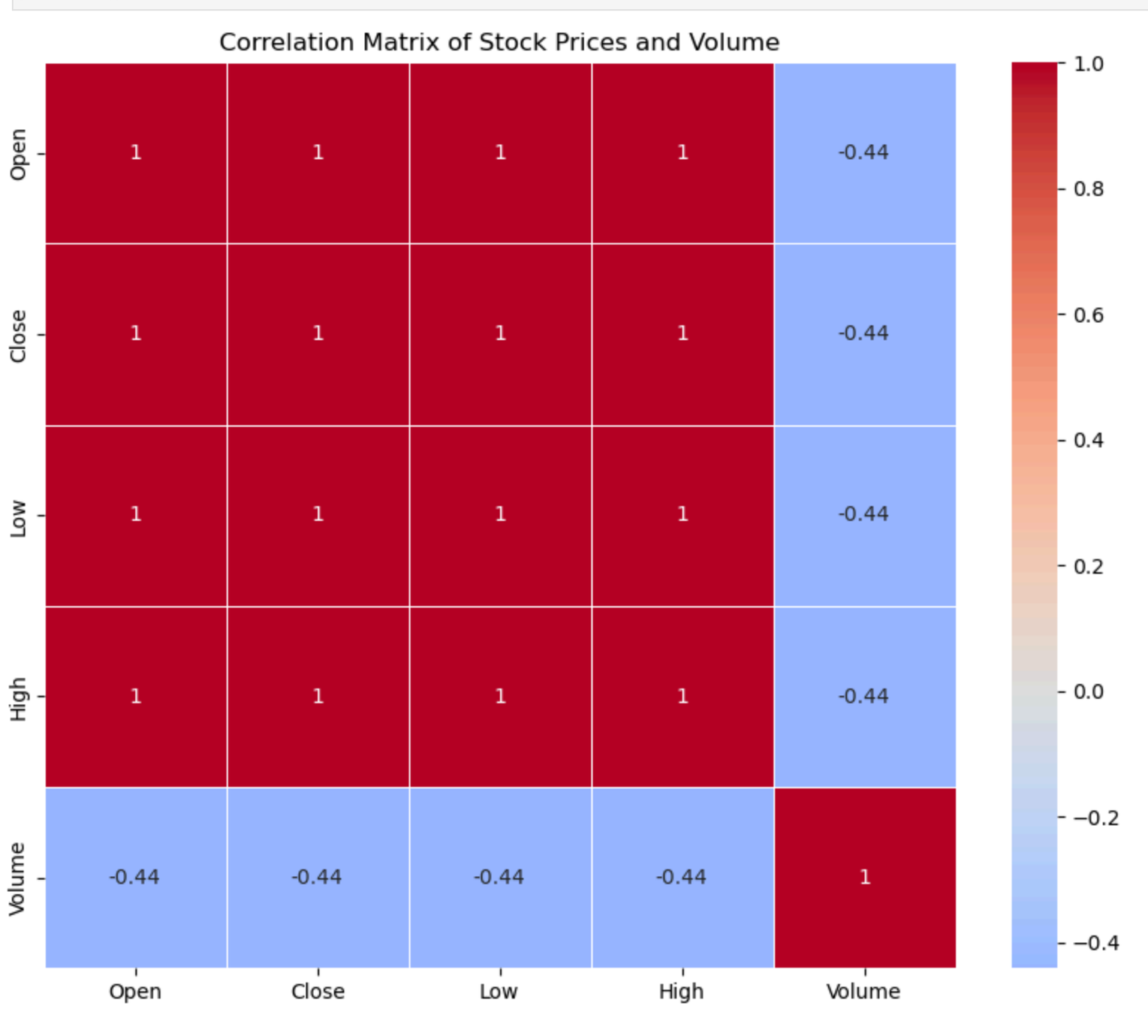
Bar plot showing total volume over the years

In [115…
```python
citi["Year_c"]=citi['Date'].dt.year
yearly_volume=citi.groupby("Year_c")['Volume'].sum().reset_index()

sns.barplot(data=yearly_volume,x="Year_c",y="Volume")
plt.title('Total Trading Volumne by Year')
plt.xlabel('Year')
plt.xticks(rotation=45)
plt.show()
```



Heatmap showing relationship between various apspects

In [156…
```python
plt.figure(figsize=(10,8))

sns.heatmap(data=citi[['Open','Close','Low','High','Volume']].corr(),annot=True,cmap='coolwarm',center=0,linewidth=.5)
plt.title("Correlation Matrix of Stock Prices and Volume")
plt.show()
```



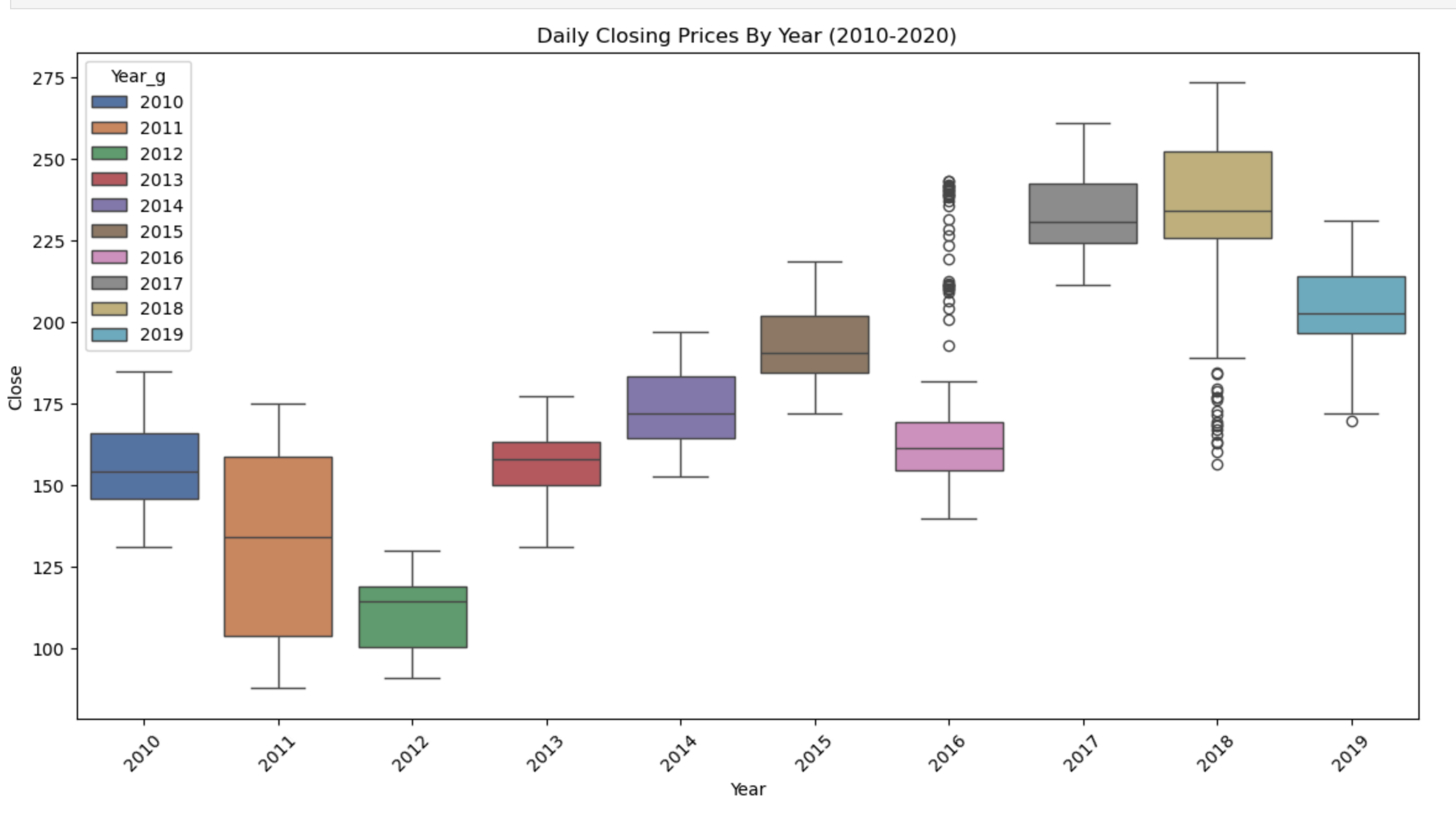
GS- GOLDMANSACH STOCK PRICE HISTORY CHARTS

In [170…
```python
gold=pd.read_csv('gs.csv')
gold['Date']=gold['Date'].astype('datetime64[ns]')
gold.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3523 entries, 0 to 3522
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       3523 non-null   datetime64[ns]
 1   Open       3523 non-null   float64
 2   High       3523 non-null   float64
 3   Low        3523 non-null   float64
 4   Close      3523 non-null   float64
 5   Adj Close  3523 non-null   float64
 6   Volume     3523 non-null   int64
dtypes: datetime64[ns](1), float64(5), int64(1)
memory usage: 192.8 KB
```

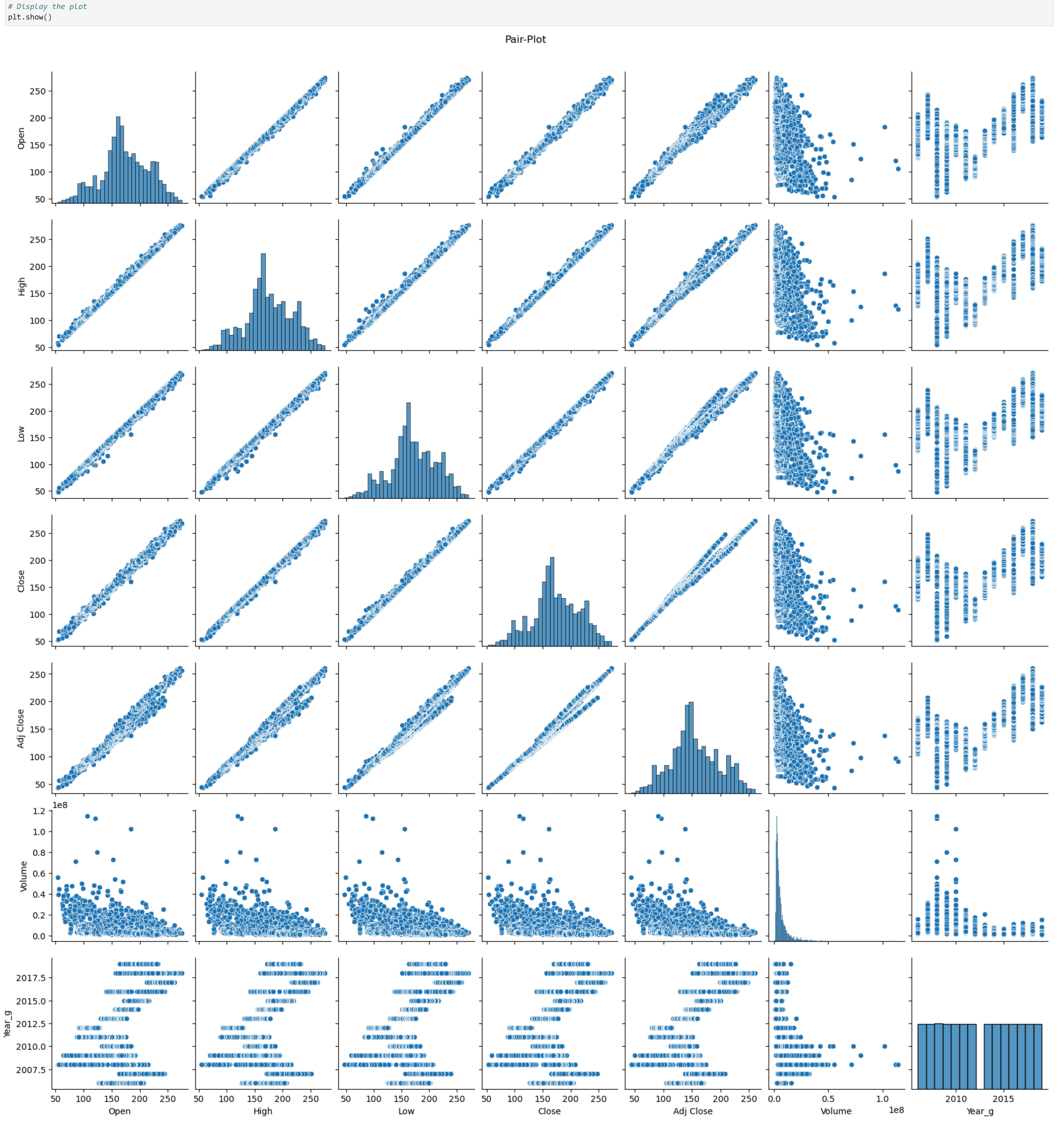
BOX Plot showing daily closing balance from 2010 - 2019

In [186…
```python
gold['Year_g']=gold['Date'].dt.year
filtered_gold=gold[(gold['Year_g']>=2010) & (gold['Year_g']<=2020)]


plt.figure(figsize=(14,7))
sns.boxplot(data=filtered_gold,x='Year_g',y='Close',palette='deep',hue='Year_g',dodge=False)
plt.title('Daily Closing Prices By Year (2010-2020)')
plt.xlabel("Year")
plt.xticks(rotation=45)
plt.show()
```



Pair plot

In [196…
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Create the pair plot
pair_plot = sns.pairplot(data=gold)
pair_plot.fig.suptitle("Pair-Plot", y=1.02)  # Adjust title position

# Display the plot
plt.show()
```



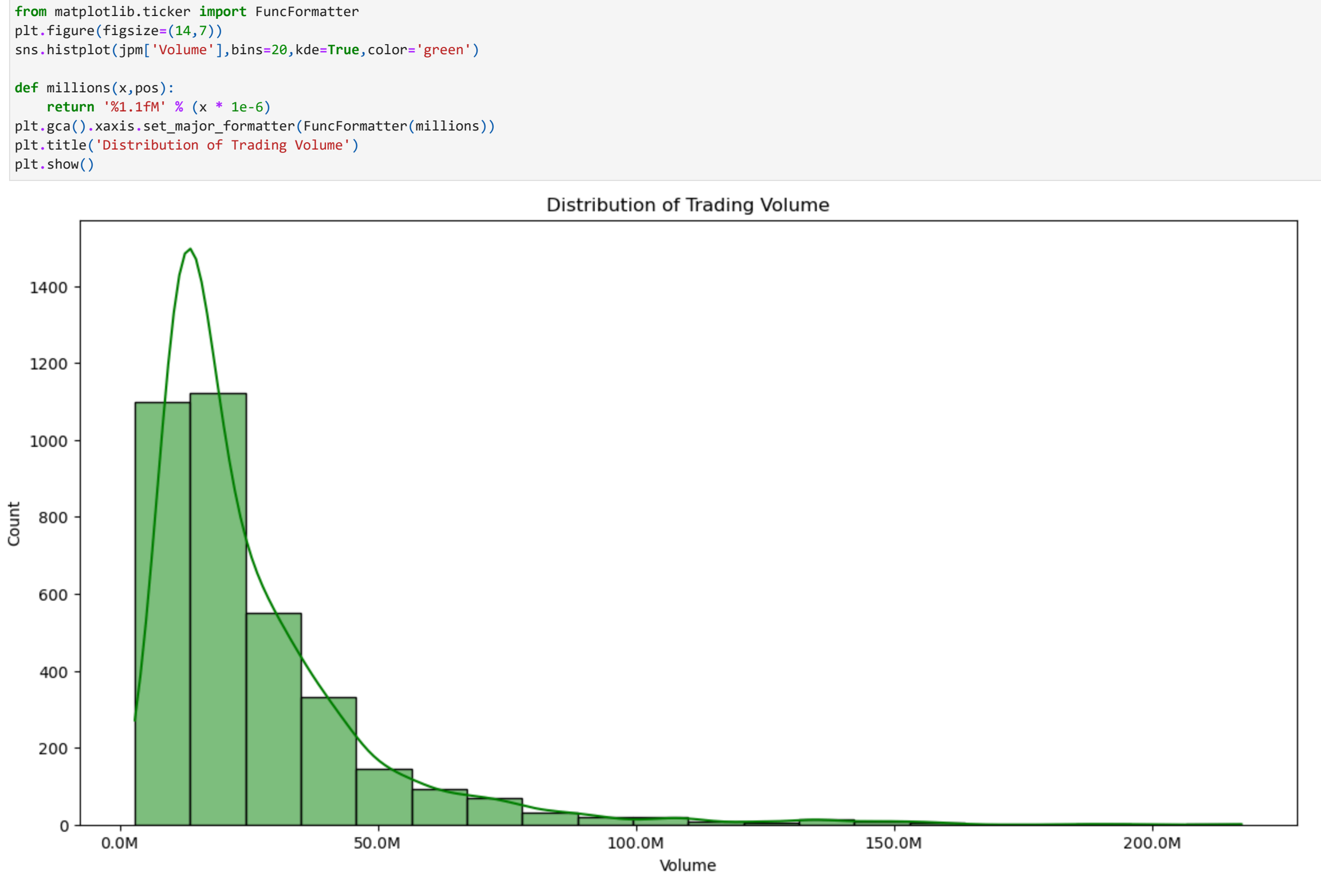
JPM- JP MORGAAN BANK STOCK PRICE HISTORY CHARTS

In [194…
```python
jpm=pd.read_csv('JPM.csv')
jpm['Date']=jpm['Date'].astype('datetime64[ns]')
jpm.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3523 entries, 0 to 3522
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       3523 non-null   datetime64[ns]
 1   Open       3523 non-null   float64
 2   High       3523 non-null   float64
 3   Low        3523 non-null   float64
 4   Close      3523 non-null   float64
 5   Adj Close  3523 non-null   float64
 6   Volume     3523 non-null   int64
dtypes: datetime64[ns](1), float64(5), int64(1)
memory usage: 192.8 KB
```

HISTOGRAM showing distribution of trading volume

In [204…
```python
from matplotlib.ticker import FuncFormatter
plt.figure(figsize=(14,7))
sns.histplot(jpm['Volume'],bins=20,kde=True,color='green')

def millions(x,pos):
    return '%1.1fM' % (x * 1e-6)
plt.gca().xaxis.set_major_formatter(FuncFormatter(millions))
plt.title('Distribution of Trading Volume')
plt.show()
```



MS-MORAGAN STANLEY STOCK PRICE HISTORY CHARTS

In [209…
```python
morgan=pd.read_csv('MS.csv')
morgan['Date']=morgan['Date'].astype('datetime64[ns]')
morgan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3523 entries, 0 to 3522
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       3523 non-null   datetime64[ns]
 1   Open       3523 non-null   float64
 2   High       3523 non-null   float64
 3   Low        3523 non-null   float64
 4   Close      3523 non-null   float64
 5   Adj Close  3523 non-null   float64
 6   Volume     3523 non-null   int64
dtypes: datetime64[ns](1), float64(5), int64(1)
memory usage: 192.8 KB
```

Count Plot showing

In [222…
```python
morgan['Price_Category'] = ['High' if x > morgan['Close'].mean() else 'Low' for x in morgan['Close']]

# Create the count plot
plt.figure(figsize=(14, 7))
sns.countplot(data=morgan, x='Price_Category', palette='coolwarm')
plt.title('Count of High and Low Prices')
plt.xlabel('Price Category')
plt.ylabel('Count')
plt.show()
```

```
C:\Users\PIYUSH KADAM\AppData\Local\Temp\ipykernel_60552\4039362715.py:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.countplot(data=morgan, x='Price_Category', palette='coolwarm')
```