In [1]:
```
import pandas as pd
```

In [2]:
```
df1 = pd.read_csv('goldstock.csv')
```

In [3]:
```
df1.head()
```

Out[3]:

	Unnamed: 0	Date	Close	Volume	Open	High	Low
0	0	2024-01-19	2029.3	166078.0	2027.4	2041.9	2022.2
1	1	2024-01-18	2021.6	167013.0	2009.1	2025.6	2007.7
2	2	2024-01-17	2006.5	245194.0	2031.7	2036.1	2004.6
3	3	2024-01-16	2030.2	277995.0	2053.4	2062.8	2027.6
4	4	2024-01-12	2051.6	250946.0	2033.2	2067.3	2033.1

In [4]:
```
df1.tail()
```

Out[4]:

	Unnamed: 0	Date	Close	Volume	Open	High	Low
2506	2528	2014-01-28	1250.5	81426.0	1254.9	1261.9	1248.0
2507	2529	2014-01-27	1263.5	63419.0	1269.9	1280.1	1252.0
2508	2530	2014-01-24	1264.5	34998.0	1264.3	1273.2	1256.9
2509	2531	2014-01-23	1262.5	41697.0	1235.1	1267.1	1230.8
2510	2532	2014-01-22	1238.6	80262.0	1240.5	1243.5	1235.5

In [6]:
```
df1.shape
```

Out[6]:
```
(2511, 7)
```

In [7]:
```
df1.columns
```

Out[7]:
```
Index(['Unnamed: 0', 'Date', 'Close', 'Volume', 'Open', 'High', 'Low'], dtype='object')
```

In [8]:
```
df1.duplicated().sum()
```

Out[8]:
```
0
```

In [9]:
```
df1.isnull().sum()
```

Out[9]:
```
Unnamed: 0    0
Date          0
Close         0
Volume        0
Open          0
High          0
Low           0
dtype: int64
```

In [10]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2511 entries, 0 to 2510
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Unnamed: 0  2511 non-null   int64  
 1   Date        2511 non-null   object 
 2   Close       2511 non-null   float64
 3   Volume      2511 non-null   float64
 4   Open        2511 non-null   float64
 5   High        2511 non-null   float64
 6   Low         2511 non-null   float64
dtypes: float64(5), int64(1), object(1)
memory usage: 137.4+ KB
```

In [12]:
```
df1.describe()
```

Out[12]:

	Unnamed: 0	Close	Volume	Open	High	Low
count	2511.000000	2511.000000	2511.000000	2511.000000	2511.000000	2511.000000
mean	1260.792911	1498.726085	185970.770609	1498.725528	1508.451454	1488.869932
std	729.262879	298.824811	97600.769382	299.118187	301.262244	296.417703
min	0.000000	1049.600000	1.000000	1051.500000	1062.700000	1045.400000
25%	630.500000	1249.850000	126693.500000	1249.500000	1257.300000	1242.350000
50%	1259.000000	1332.800000	175421.000000	1334.000000	1342.400000	1326.600000
75%	1888.500000	1805.850000	234832.000000	1805.600000	1815.450000	1793.050000
max	2532.000000	2093.100000	787217.000000	2094.400000	2098.200000	2074.600000

In [15]:

```python
df1 = df1.drop(columns=['Unnamed: 0'])
```

In [16]:

```python
df1
```

Out[16]:

	Date	Close	Volume	Open	High	Low
0	2024-01-19	2029.3	166078.0	2027.4	2041.9	2022.2
1	2024-01-18	2021.6	167013.0	2009.1	2025.6	2007.7
2	2024-01-17	2006.5	245194.0	2031.7	2036.1	2004.6
3	2024-01-16	2030.2	277995.0	2053.4	2062.8	2027.6
4	2024-01-12	2051.6	250946.0	2033.2	2067.3	2033.1
...	...	...	...	...	...	...
2506	2014-01-28	1250.5	81426.0	1254.9	1261.9	1248.0
2507	2014-01-27	1263.5	63419.0	1269.9	1280.1	1252.0
2508	2014-01-24	1264.5	34998.0	1264.3	1273.2	1256.9
2509	2014-01-23	1262.5	41697.0	1235.1	1267.1	1230.8
2510	2014-01-22	1238.6	80262.0	1240.5	1243.5	1235.5

2511 rows × 6 columns

In [17]:

```python
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

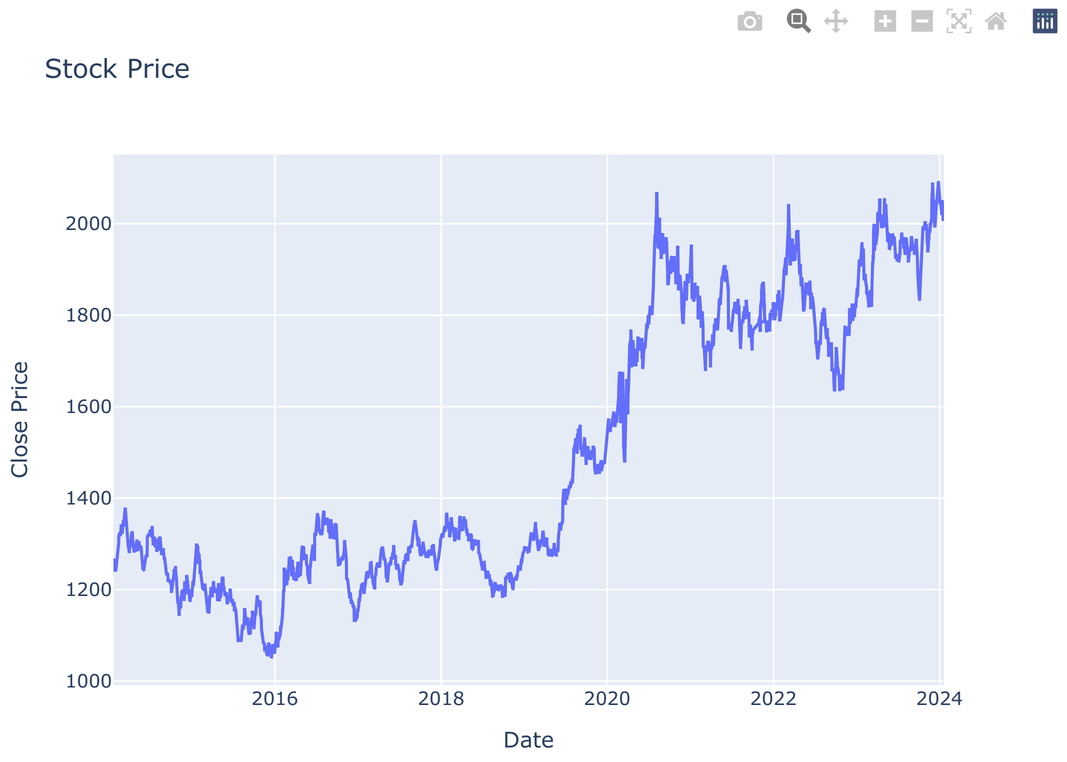
In [18]:

```python
import plotly.graph_objects as go

fig = go.Figure(data=go.Scatter(x=df1['Date'], y=df1['Close'], mode='lines'))

fig.update_layout(
    title='Stock Price',
    xaxis=dict(title='Date'),
    yaxis=dict(title='Close Price'),
)

fig.show()
```



In [19]:
```
fig = go.Figure()

# Add the scatter plot
fig.add_trace(go.Scatter(
    x=df1['Date'],
    y=df1['Close'],
    mode='markers',
    marker=dict(
        size=8,
        color='red'
    ),
    name='Close Prices'
))

fig.update_layout(
    title='Scatter Plot of Close Prices',
    xaxis_title='Date',
    yaxis_title='Close Price',
    showlegend=True,
    plot_bgcolor='black',  # Set the background color to black
    paper_bgcolor='white',  # Set the paper background color to white
    xaxis=dict(showgrid=True, gridcolor='gray'),  # Add gridlines for x-axis
    yaxis=dict(showgrid=True, gridcolor='gray')  # Add gridlines for y-axis
)

fig.show()
```

In [20]:
```python
# Find the minimum value and its corresponding date
min_close = df1['Close'].min()
min_date = df1.loc[df1['Close'] == min_close, 'Date'].values[0]

# Find the maximum value and its corresponding date
max_close = df1['Close'].max()
max_date = df1.loc[df1['Close'] == max_close, 'Date'].values[0]

print("Minimum Close Value: ", min_close, " Date: ", min_date)
print("Maximum Close Value: ", max_close, " Date: ", max_date)
```

```
Minimum Close Value:  1049.6  Date:  2015-12-17
Maximum Close Value:  2093.1  Date:  2023-12-27
```

In [21]:
```python
fig = go.Figure(data=go.Scatter(x=df1['Date'], y=df1['Volume'], mode='lines'))

fig.update_layout(
    title='Volume Trends',
    xaxis=dict(title='Date'),
    yaxis=dict(title='Volume'),
)

fig.show()
```

In [22]:
```python
fig = go.Figure()

# Add the scatter plot
fig.add_trace(go.Scatter(
    x=df1['Date'],
    y=df1['Volume'],
    mode='markers',
    marker=dict(
        size=8,
        color='blue'
    ),
    name='Volume'
))

fig.update_layout(
    title='Scatter Plot of Volume',
    xaxis_title='Date',
    yaxis_title='Volume',
    showlegend=True
)

fig.show()
```

```
In [23]: # Find the minimum value and its corresponding date
min_volume = df1['Volume'].min()
min_date = df1.loc[df1['Volume'] == min_volume, 'Date'].values[0]

# Find the maximum value and its corresponding date
max_volume = df1['Volume'].max()
max_date = df1.loc[df1['Volume'] == max_volume, 'Date'].values[0]

print("Minimum Volume: ", min_volume, " Date: ", min_date)
print("Maximum Volume: ", max_volume, " Date: ", max_date)
```

```
Minimum Volume:  1.0  Date:  2019-09-13
Maximum Volume:  787217.0  Date:  2020-01-08
```

```
In [24]: df1['Market Cap'] = df1['Open']*df1['Volume']
```

```
In [25]: fig = go.Figure(data=go.Scatter(x=df1['Date'], y=df1['Market Cap'], mode='lines'))

fig.update_layout(
    title='Market Cap',
    xaxis=dict(title='Date'),
    yaxis=dict(title='Market Cap'),
)

fig.show()
```

In [26]:

```python
fig = go.Figure()

# Add the scatter plot
fig.add_trace(go.Scatter(
    x=df1['Date'],
    y=df1['Market Cap'],
    mode='markers',
    marker=dict(
        size=8,
        color='blue'
    ),
    name='Market Cap'
))

fig.update_layout(
    title='Scatter Plot of Market Cap',
    xaxis_title='Date',
    yaxis_title='Market Cap',
    showlegend=True
)

fig.show()
```

In [28]:
```
# Assuming df1 is your DataFrame containing the data
minimum_value = df1['Market Cap'].min()
maximum_value = df1['Market Cap'].max()

# Find the corresponding dates for the minimum and maximum values
minimum_date = df1[df1['Market Cap'] == minimum_value]['Date'].iloc[0]
maximum_date = df1[df1['Market Cap'] == maximum_value]['Date'].iloc[0]

print("Minimum Market Cap:", minimum_value, "on", minimum_date)
print("Maximum Market Cap:", maximum_value, "on", maximum_date)
```

```
Minimum Market Cap: 1201.6 on 2018-11-12
Maximum Market Cap: 1225460703.9 on 2020-01-08
```

In [29]:
```
df1.iloc[df1['Market Cap'].argmax()]
```

Out[29]:
```
Date              2020-01-08
Close                 1560.2
Volume              787217.0
Open                  1556.7
High                  1563.8
Low                   1556.5
Market Cap      1225460703.9
Name: 1010, dtype: object
```

In [30]:
```
df1.iloc[df1['Market Cap'].argmin()]
```

Out[30]:
```
Date            2018-11-12
Close               1201.3
Volume                 1.0
Open                1201.6
High                1201.6
Low                 1201.6
Market Cap          1201.6
Name: 1299, dtype: object
```

In [31]:
```
ohlc = df1[pd.to_datetime(df1['Date']).dt.date > pd.to_datetime('2021-01-01').date()]
```

In [32]:
```
fig = go.Figure(data=go.Scatter(
    x=ohlc['Date'],
    y=ohlc['Market Cap'],
    line=dict(color='red')
))

fig.update_layout(
    title='Market Cap (After 1st Jan, 2021)',
    xaxis_title='Date',
    yaxis_title='Market Cap',
    showlegend=False,
    xaxis_tickangle=-45,
    yaxis_showgrid=True,
    width=900,
    height=500
)
```

```
fig.show()
```

In [33]:
```
fig = go.Figure()

fig.add_trace(go.Scatter(
    x=ohlc['Date'],
    y=ohlc['Market Cap'],
    mode='lines',
    line=dict(color='red'),
    name='Market Cap'
))

fig.add_trace(go.Scatter(
    x=ohlc['Date'],
    y=ohlc['Market Cap'],
    mode='markers',
    marker=dict(color='blue', size=5),
    name='Market Cap Scatter'
))

fig.update_layout(
    title='Market Cap (After 1st Jan, 2021)',
    xaxis_title='Date',
    yaxis_title='Market Cap',
    showlegend=False,
    xaxis_tickangle=-45,
    yaxis_showgrid=True,
    width=900,
    height=500
)

fig.show()
```

In [34]:
```
df1['vol'] = (df1['Close']/df1['Close'].shift(1)) - 1
```

In [35]:
```
fig = go.Figure(data=go.Scatter(x=df1['Date'], y=df1['vol'], mode='lines'))

fig.update_layout(
    title='Volatility Plot',
    xaxis=dict(title='Date'),
    yaxis=dict(title='Volatility'),
)

fig.show()
```

In [36]:
```
fig = go.Figure()

# Add the scatter plot
fig.add_trace(go.Scatter(
    x=df1['Date'],
    y=df1['vol'],
    mode='markers',
    marker=dict(
        size=8,
        color='blue'
    ),
    name='Volatility'
))

fig.update_layout(
    title='Scatter Plot of Volatility',
    xaxis_title='Date',
    yaxis_title='Volatility',
    showlegend=True
)

fig.show()
```

In [37]:

```
fig = go.Figure()

fig.add_trace(go.Histogram(
    x=df1['vol'],
    nbinsx=100,
    marker_color='red'
))

fig.update_layout(
    title='Histogram of Volatility',
    xaxis_title='Volume',
    yaxis_title='Count'
)

fig.show()
```

In [38]:
```
df1['Cumulative Return'] = (1 + df1['vol']).cumprod()
```

In [39]:
```
fig = go.Figure(data=go.Scatter(x=df1['Date'], y=df1['Cumulative Return'], mode='lines'))

fig.update_layout(
    title='Cumulative Return',
    xaxis=dict(title='Date'),
    yaxis=dict(title= 'Cumulative Return'),
)

fig.show()
```

In [40]:
```
fig = go.Figure()

# Add the scatter plot
fig.add_trace(go.Scatter(
    x=df1['Date'],
    y=df1['Cumulative Return'],
    mode='markers',
    marker=dict(
        size=8,
        color='blue'
    ),
    name='Cumulative Return'
))

fig.update_layout(
    title='Scatter Plot of Cumulative Return',
    xaxis_title='Date',
    yaxis_title= 'Cumulative Return',
    showlegend=True
)

fig.show()
```

In [41]:
```
ohlc = df1[pd.to_datetime(df1['Date']).dt.date > pd.to_datetime('2021-01-01').date()]
```

In [42]:
```
ohlc
```

Out[42]:

	Date	Close	Volume	Open	High	Low	Market Cap	vol	Cumulative Return
0	2024-01-19	2029.3	166078.0	2027.4	2041.9	2022.2	336706537.2	NaN	NaN
1	2024-01-18	2021.6	167013.0	2009.1	2025.6	2007.7	335545818.3	-0.003794	0.996206
2	2024-01-17	2006.5	245194.0	2031.7	2036.1	2004.6	498160649.8	-0.007469	0.988765
3	2024-01-16	2030.2	277995.0	2053.4	2062.8	2027.6	570834933.0	0.011812	1.000444
4	2024-01-12	2051.6	250946.0	2033.2	2067.3	2033.1	510223407.2	0.010541	1.010989
...	...	...	...	...	...	...	...	...	...
758	2021-01-08	1835.4	422485.0	1915.2	1918.4	1827.8	809143272.0	-0.008321	0.904450
759	2021-01-07	1913.6	192365.0	1921.5	1929.6	1907.5	369629347.5	0.042607	0.942985
760	2021-01-06	1908.6	356182.0	1952.8	1962.5	1902.6	695552209.6	-0.002613	0.940521
761	2021-01-05	1954.4	192111.0	1946.0	1957.0	1938.4	373848006.0	0.023997	0.963091
762	2021-01-04	1946.6	261675.0	1908.2	1948.7	1906.1	499328235.0	-0.003991	0.959247

763 rows × 9 columns

In [43]:
```
fig = go.Figure(data=go.Scatter(
    x=ohlc['Date'],
    y=ohlc['Cumulative Return'],
    line=dict(color='red')
))

fig.update_layout(
    title='Cumulative Return (After 1st Jan, 2021)',
    xaxis_title='Date',
    yaxis_title= 'Cumulative Return',
    showlegend=False,
    xaxis_tickangle=-45,
```

```python
    yaxis_showgrid=True,
    width=900,
    height=500
)

fig.show()
```

In [44]:

```python
fig = go.Figure()

fig.add_trace(go.Scatter(
    x=ohlc['Date'],
    y=ohlc['Cumulative Return'],
    mode='lines',
    line=dict(color='red'),
    name= 'Cumulative Return'
))

fig.add_trace(go.Scatter(
    x=ohlc['Date'],
    y=ohlc['Cumulative Return'],
    mode='markers',
    marker=dict(color='blue', size=5),
    name='Cumulative Return Scatter'
))

fig.update_layout(
    title='Cumulative Return (After 1st Jan, 2021)',
    xaxis_title='Date',
    yaxis_title= 'Cumulative Return',
    showlegend=False,
    xaxis_tickangle=-45,
    yaxis_showgrid=True,
    width=900,
    height=500
)

fig.show()
```

```
In [45]: df1.iloc[df1['Cumulative Return'].argmax()]
```

```
Out[45]: Date                     2023-12-27
         Close                        2093.1
         Volume                     124021.0
         Open                         2079.3
         High                         2095.8
         Low                          2072.8
         Market Cap              257876865.3
         vol                        0.004608
         Cumulative Return          1.031439
         Name: 15, dtype: object
```

```
In [46]: df1.iloc[df1['Cumulative Return'].argmin()]
```

```
Out[46]: Date                     2015-12-17
         Close                        1049.6
         Volume                     157113.0
         Open                         1072.2
         High                         1072.7
         Low                          1046.8
         Market Cap              168456558.6
         vol                        -0.01446
         Cumulative Return          0.517223
         Name: 2029, dtype: object
```

```
In [47]: from sklearn.preprocessing import MinMaxScaler
         from keras.models import Sequential
         from keras.layers import Dense, LSTM
         import math
```

```
WARNING:tensorflow:From C:\Users\Acer\anaconda3\Lib\site-packages\keras\src\losses.py:2976: The name tf.losses.
sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cross_entropy instead
.
```

```
In [48]: df1['Date'] = pd.to_datetime(df1['Date'])
         df1.set_index('Date',inplace=True)
```

```
In [49]: df1
```

Out[49]:

	Close	Volume	Open	High	Low	Market Cap	vol	Cumulative Return
Date								
2024-01-19	2029.3	166078.0	2027.4	2041.9	2022.2	336706537.2	NaN	NaN
2024-01-18	2021.6	167013.0	2009.1	2025.6	2007.7	335545818.3	-0.003794	0.996206
2024-01-17	2006.5	245194.0	2031.7	2036.1	2004.6	498160649.8	-0.007469	0.988765
2024-01-16	2030.2	277995.0	2053.4	2062.8	2027.6	570834933.0	0.011812	1.000444
2024-01-12	2051.6	250946.0	2033.2	2067.3	2033.1	510223407.2	0.010541	1.010989
...	...	...	...	...	...	...	...	...
2014-01-28	1250.5	81426.0	1254.9	1261.9	1248.0	102181487.4	-0.009270	0.616222
2014-01-27	1263.5	63419.0	1269.9	1280.1	1252.0	80535788.1	0.010396	0.622628
2014-01-24	1264.5	34998.0	1264.3	1273.2	1256.9	44247971.4	0.000791	0.623121
2014-01-23	1262.5	41697.0	1235.1	1267.1	1230.8	51499964.7	-0.001582	0.622136
2014-01-22	1238.6	80262.0	1240.5	1243.5	1235.5	99565011.0	-0.018931	0.610358

2511 rows × 8 columns

In [50]:

```
data = df1.filter(['Close'])
data
```

Out[50]:

	Close
Date	
2024-01-19	2029.3
2024-01-18	2021.6
2024-01-17	2006.5
2024-01-16	2030.2
2024-01-12	2051.6
...	...
2014-01-28	1250.5
2014-01-27	1263.5
2014-01-24	1264.5
2014-01-23	1262.5
2014-01-22	1238.6

2511 rows × 1 columns

In [ ]: