In [1]:
```
import pandas as pd
%pip install yfinance
import yfinance as yf
import datetime
from datetime import date, timedelta
```

```
Requirement already satisfied: yfinance in c:\users\marjan\anaconda3\lib\site-packages (0.1.74)
Requirement already satisfied: multitasking>=0.0.7 in c:\users\marjan\anaconda3\lib\site-packages (from yfinance) (0.0.
11)
Requirement already satisfied: numpy>=1.15 in c:\users\marjan\anaconda3\lib\site-packages (from yfinance) (1.16.5)
Requirement already satisfied: pandas>=0.24.0 in c:\users\marjan\anaconda3\lib\site-packages (from yfinance) (0.25.1)
Requirement already satisfied: lxml>=4.5.1 in c:\users\marjan\anaconda3\lib\site-packages (from yfinance) (4.9.1)
Requirement already satisfied: requests>=2.26 in c:\users\marjan\anaconda3\lib\site-packages (from yfinance) (2.28.1)
Requirement already satisfied: pytz>=2017.2 in c:\users\marjan\anaconda3\lib\site-packages (from pandas>=0.24.0->yfinan
ce) (2019.3)
Requirement already satisfied: python-dateutil>=2.6.1 in c:\users\marjan\anaconda3\lib\site-packages (from pandas>=0.2
4.0->yfinance) (2.8.0)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\marjan\anaconda3\lib\site-packages (from requests>=
2.26->yfinance) (2.1.1)
Requirement already satisfied: idna<4,>=2.5 in c:\users\marjan\anaconda3\lib\site-packages (from requests>=2.26->yfinan
ce) (2.8)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\marjan\anaconda3\lib\site-packages (from requests>=2.26->
yfinance) (2019.9.11)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\marjan\anaconda3\lib\site-packages (from requests>=2.2
6->yfinance) (1.24.2)
Requirement already satisfied: six>=1.5 in c:\users\marjan\anaconda3\lib\site-packages (from python-dateutil>=2.6.1->pa
ndas>=0.24.0->yfinance) (1.12.0)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
today = date.today()
today
```

Out[2]:
```
datetime.date(2022, 8, 28)
```

In [4]:
```
day1 = today.strftime("%Y-%m-%d")
end_date = day1
day2 = date.today() - timedelta(days=720)
day2 = day2.strftime("%Y-%m-%d")
start_date = day2
```

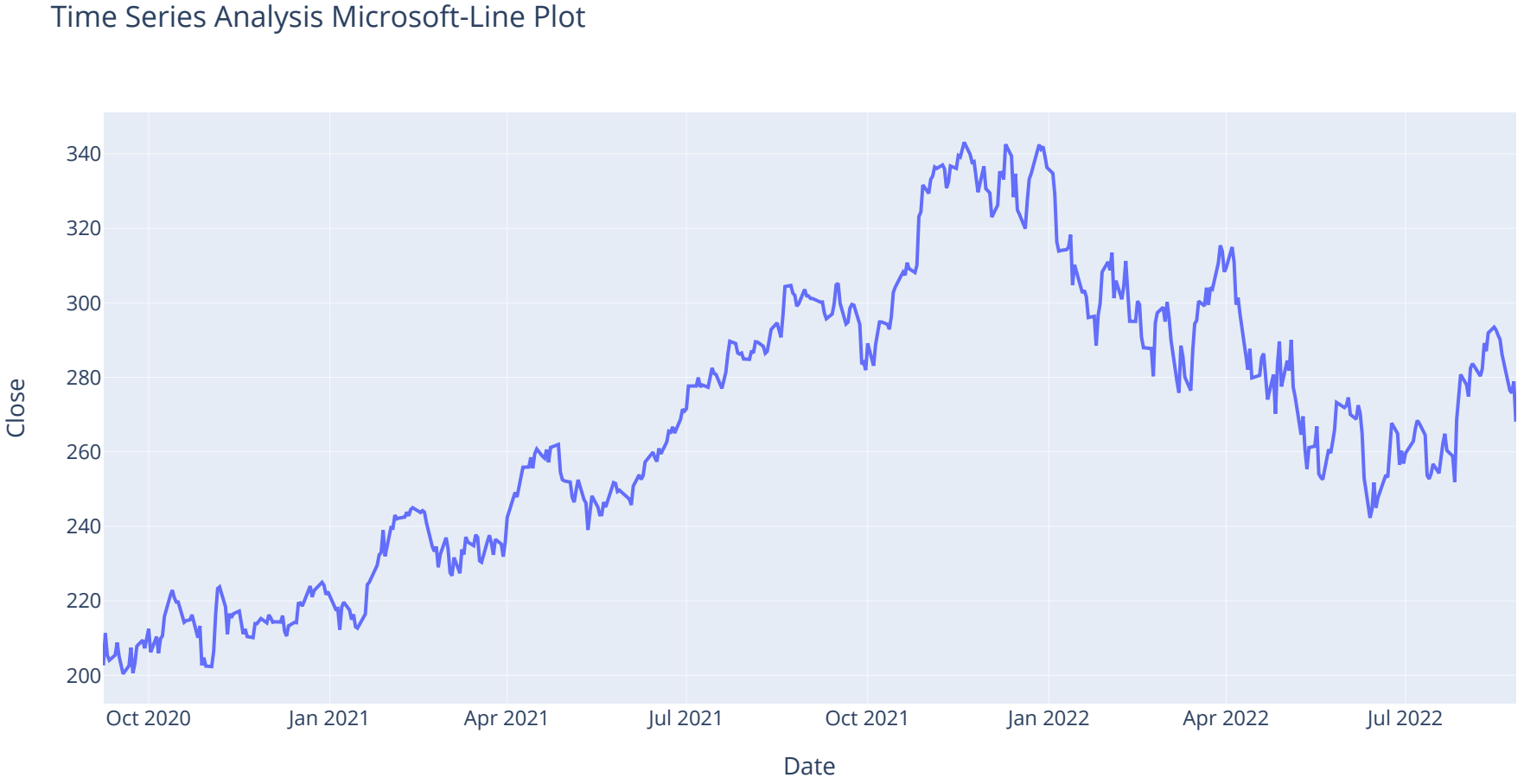
In [5]:
```
#Extracting the lates stock price of Apple by using the yfinance (Yahoo Finance) API
data_a = yf.download('AAPL',
                  start = start_date,
                  end = end_date,
                  progress=False)
print(data_a.head())


#Extracting the lates stock price of Microsoft by using the yfinance (Yahoo Finance) API
data_m = yf.download('MSFT',
                  start = start_date,
                  end = end_date,
                  progress=False)
print(data_m.head())
```

```
                 Open        High         Low       Close   Adj Close  \
Date
2020-09-08  113.949997  118.989998  112.680000  112.820000  111.474350
2020-09-09  117.260002  119.139999  115.260002  117.320000  115.920662
2020-09-10  120.360001  120.500000  112.500000  113.489998  112.136345
2020-09-11  114.570000  115.230003  110.000000  112.000000  110.664131
2020-09-14  114.720001  115.930000  112.800003  115.360001  113.984055

               Volume
Date
2020-09-08  231366600
2020-09-09  176940500
2020-09-10  182274400
2020-09-11  180860300
2020-09-14  140150100
                 Open        High         Low       Close   Adj Close  \
Date
2020-09-08  206.500000  210.029999  202.199997  202.660004  199.149429
2020-09-09  207.600006  214.839996  206.699997  211.289993  207.629898
2020-09-10  213.399994  214.740005  204.110001  205.369995  201.812469
2020-09-11  207.199997  208.630005  201.240005  204.029999  200.495682
2020-09-14  204.240005  209.199997  204.029999  205.410004  201.851761

              Volume
Date
2020-09-08  52924300
2020-09-09  45679000
2020-09-10  35461500
2020-09-11  33620100
2020-09-14  30375800
```

In [6]:

```
# visualizing the data

%pip install plotly
import plotly.express as px
figure_a = px.line(data_a, x= data_a.index,
                y = "Close",
                title = "Time Series Analysis Apple-Line Plot")
figure_a.data[0].line.color = "green"
figure_a.show()

#The trends is based on the Close prices of Samsung in each date.
#By placing the cursor on the line, we can see each price and date


figure_m = px.line(data_m, x= data_m.index,
                y = "Close",
                title = "Time Series Analysis Microsoft-Line Plot")
figure_m.show()
```

```
Requirement already satisfied: plotly in c:\users\marjan\anaconda3\lib\site-packages (5.10.0)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\marjan\anaconda3\lib\site-packages (from plotly) (8.0.1)
Note: you may need to restart the kernel to use updated packages.
```


Time Series Analysis Microsoft-Line Plot
Close
340
320
300
280
260
240
220
200
Oct 2020
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Date

In [17]:

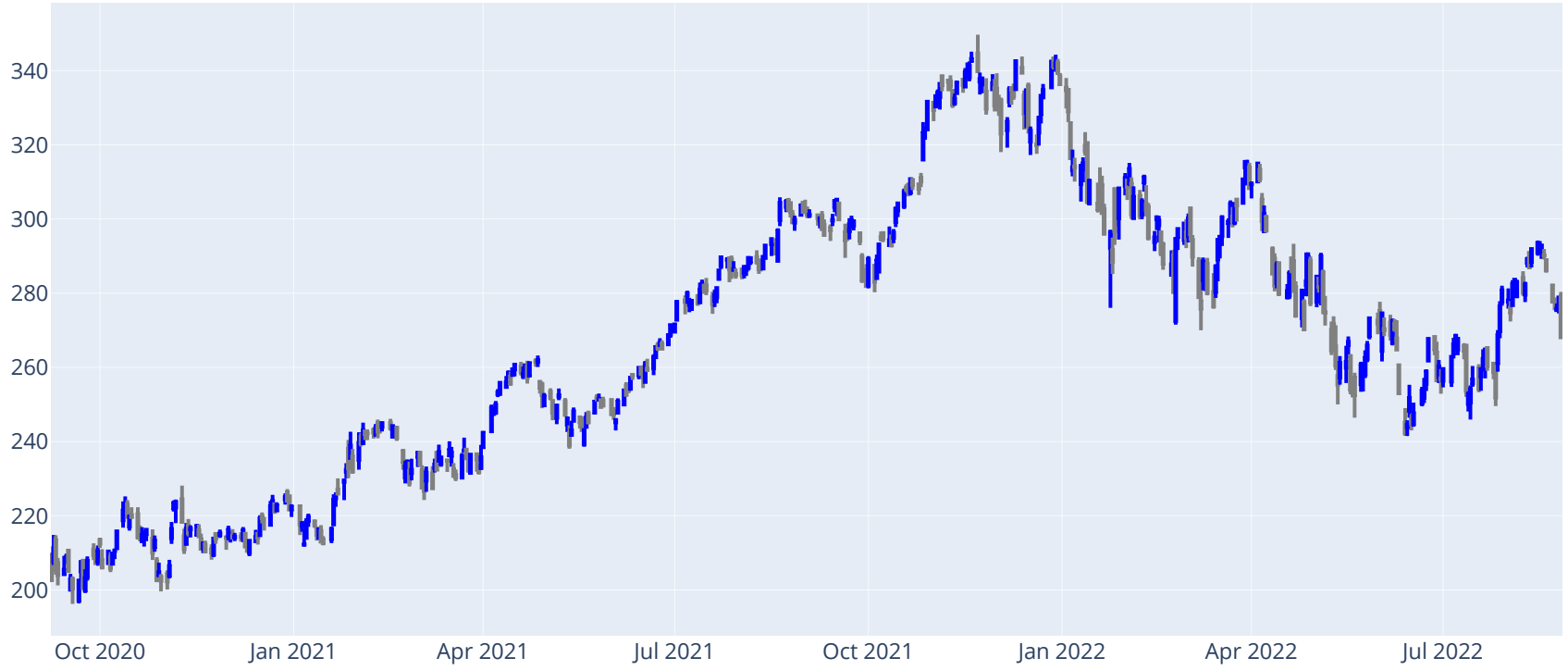
```
#more visualization this time on open, high, low, and close prices all together with a candle stick chart
#import plotly.graph_objects as go
from plotly import graph_objs as go

figure_a = go.Figure(data=[go.Candlestick(x = data_a.index,
                                          open = data_a['Open'],
                                          high = data_a['High'],
                                          low = data_a['Low'],
                                          close = data_a['Close'])])
figure_a.update_layout(title = "Time Series Analysis Apple-Candle Stick Chart",
                       xaxis_rangeslider_visible = False)
#figure_a.show()

#The gray lines show the fall in prices,
#and the green lines show the increase in prices

cs = figure_a.data[0]

# Set line and fill colors
cs.increasing.fillcolor = 'green'
cs.increasing.line.color = 'green'
cs.decreasing.fillcolor = 'gray'
cs.decreasing.line.color = 'gray'

figure_a.show()
```

Time Series Analysis Apple-Candle Stick Chart

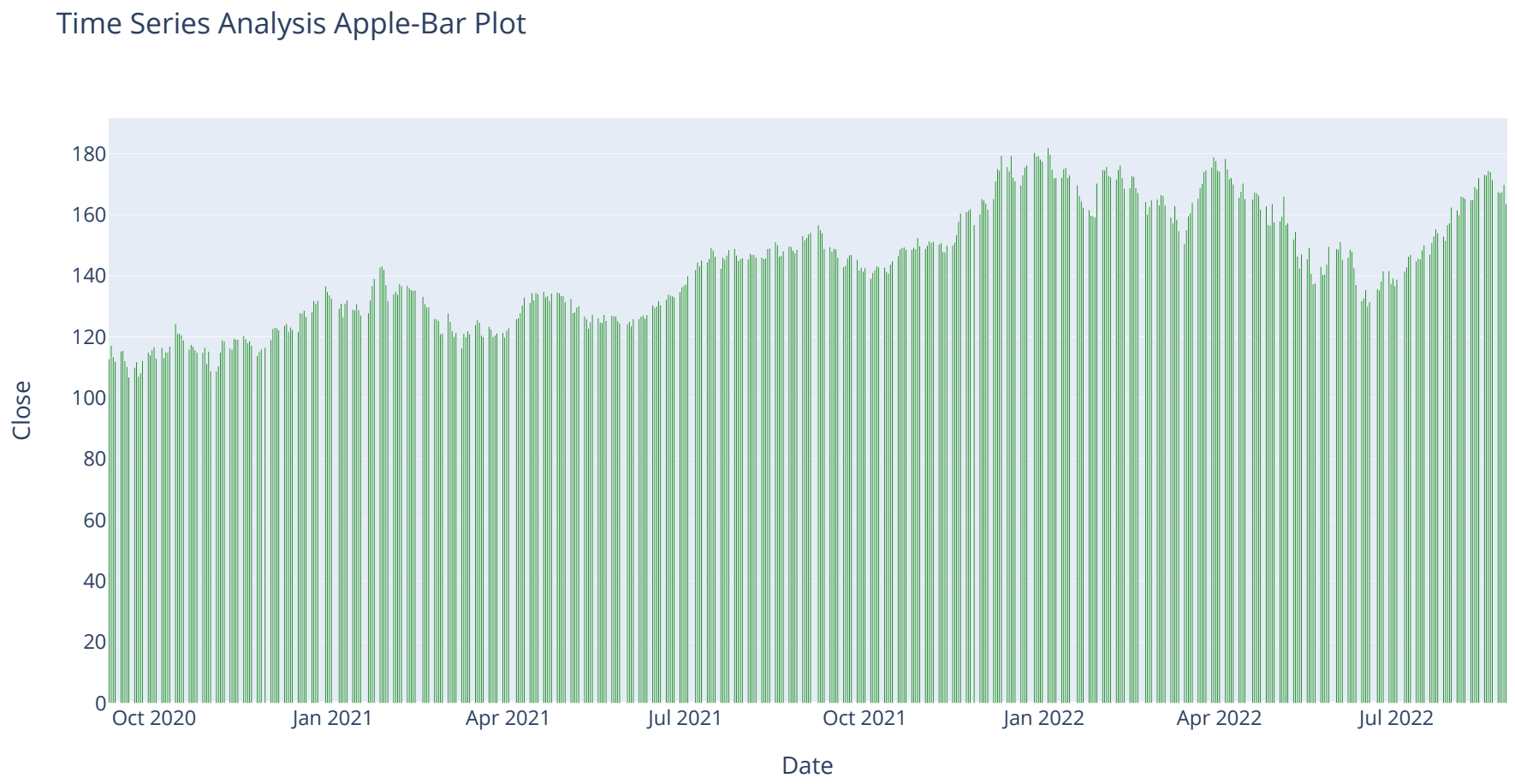
In [18]:

```
figure_m = go.Figure(data=[go.Candlestick(x = data_m.index,
                                          open = data_m['Open'],
                                          high = data_m['High'],
                                          low = data_m['Low'],
                                          close = data_m['Close'])])
figure_m.update_layout(title = "Time Series Analysis Microsoft-Candle Stick Chart",
                       xaxis_rangeslider_visible = False)
#figure_m.show()

cs = figure_m.data[0]

# Set line and fill colors
# The gray lines show the fall in prices,
# and the blue lines show the increase in prices

cs.increasing.fillcolor = 'blue'
cs.increasing.line.color = 'blue'
cs.decreasing.fillcolor = 'gray'
cs.decreasing.line.color = 'gray'

figure_m.show()
```

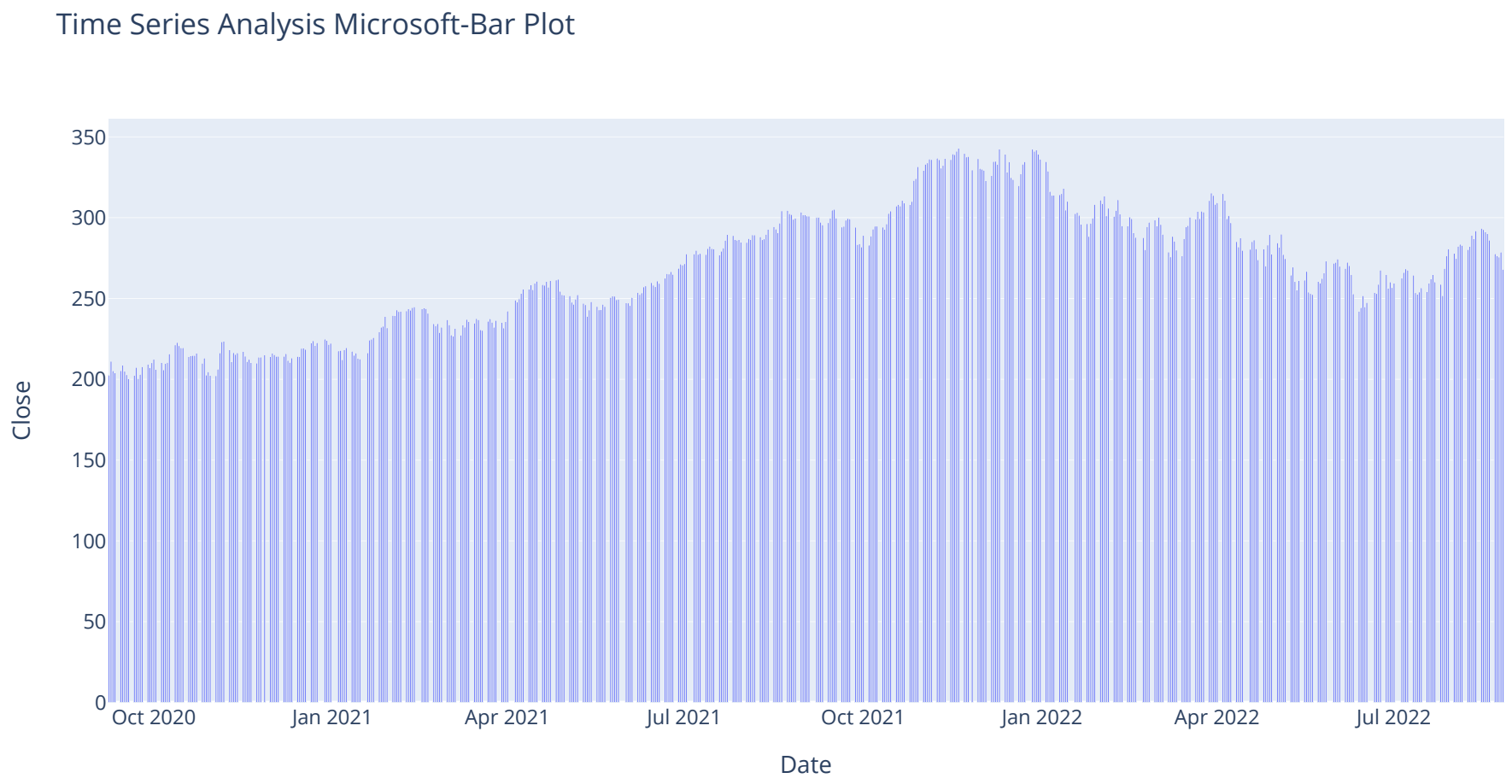
Time Series Analysis Microsoft-Candle Stick Chart



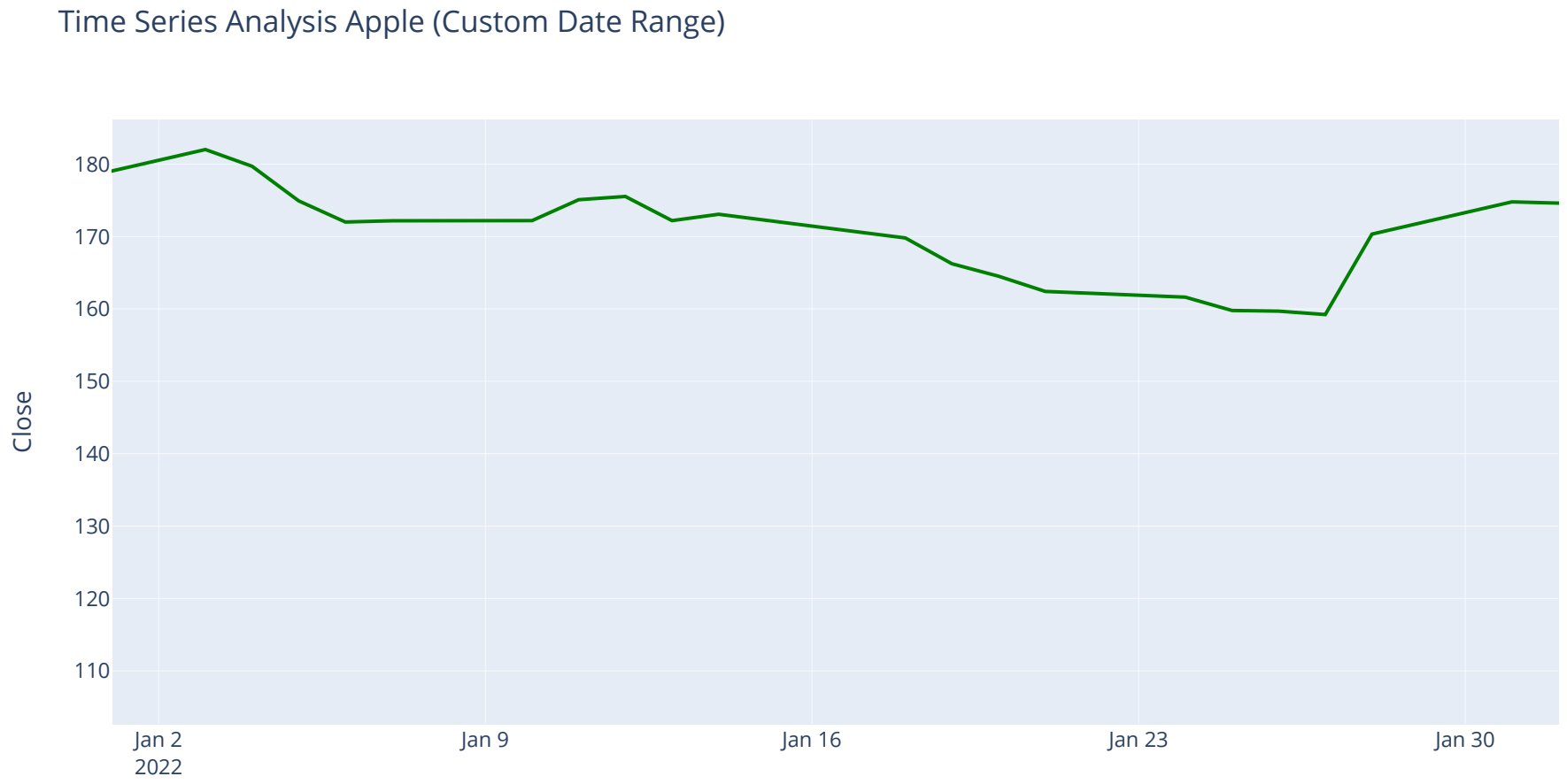
In [9]:
```
#Using a bar plot to visualize the trends of close prices over the period
figure_a = px.bar(data_a, x = data_a.index,
                  y = "Close",
                  title = "Time Series Analysis Apple-Bar Plot")
figure_a.update_traces(marker_color='green')
figure_a.show()
```



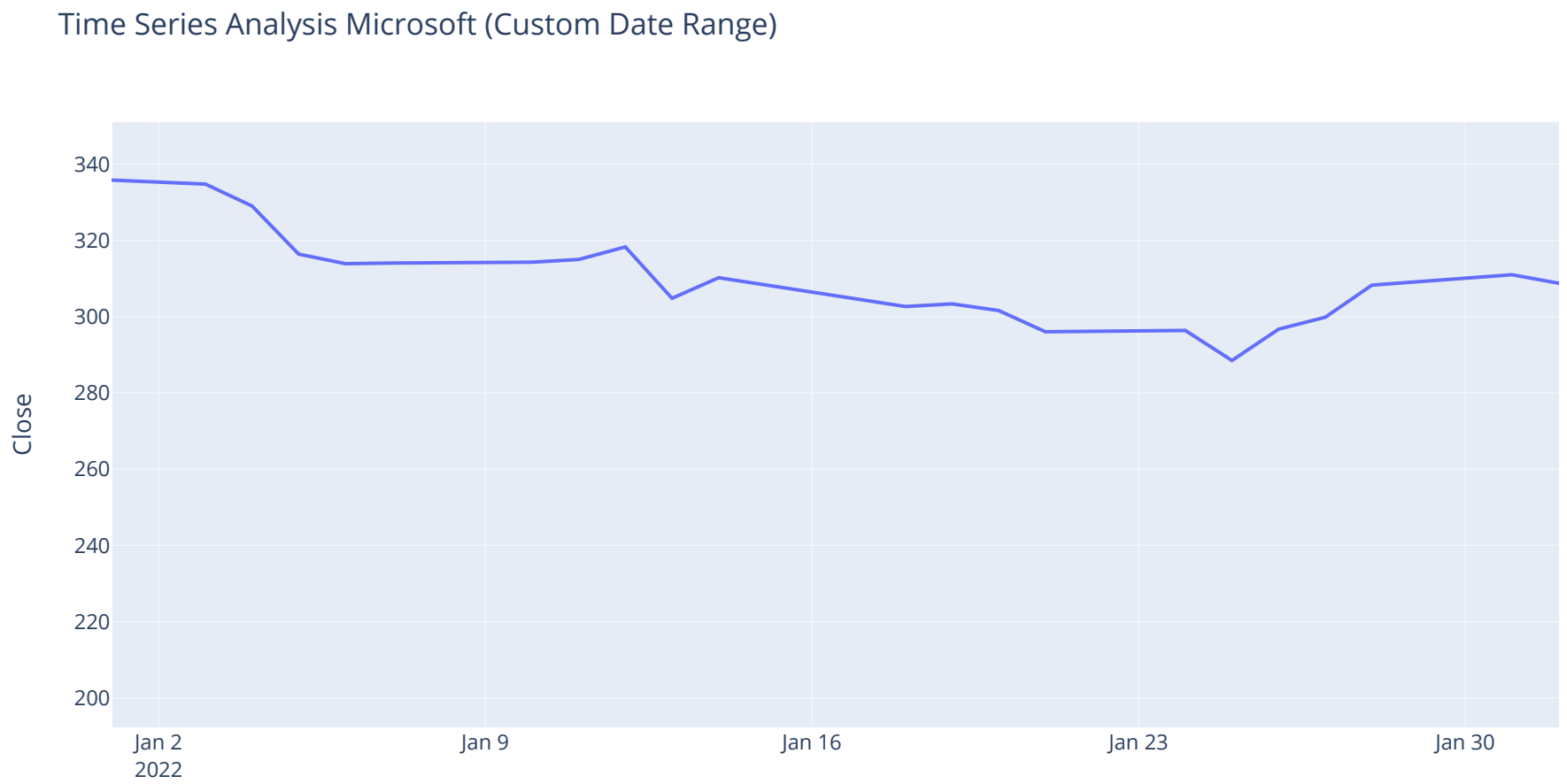
In [10]:
```
#Using a bar plot to visualize the trends of close prices over the period
figure_m = px.bar(data_m, x = data_m.index,
                  y = "Close",
                  title = "Time Series Analysis Microsoft-Bar Plot")
figure_m.show()
```



In [11]:
```
# analyzing and narrowing down the chart between two specific dates
figure_a = px.line(data_a, x = data_a.index,
                   y = 'Close',
                   range_x = ['2022-01-01','2022-02-01'],
                   title = "Time Series Analysis Apple (Custom Date Range)")
figure_a.data[0].line.color = "green"
figure_a.show()
```

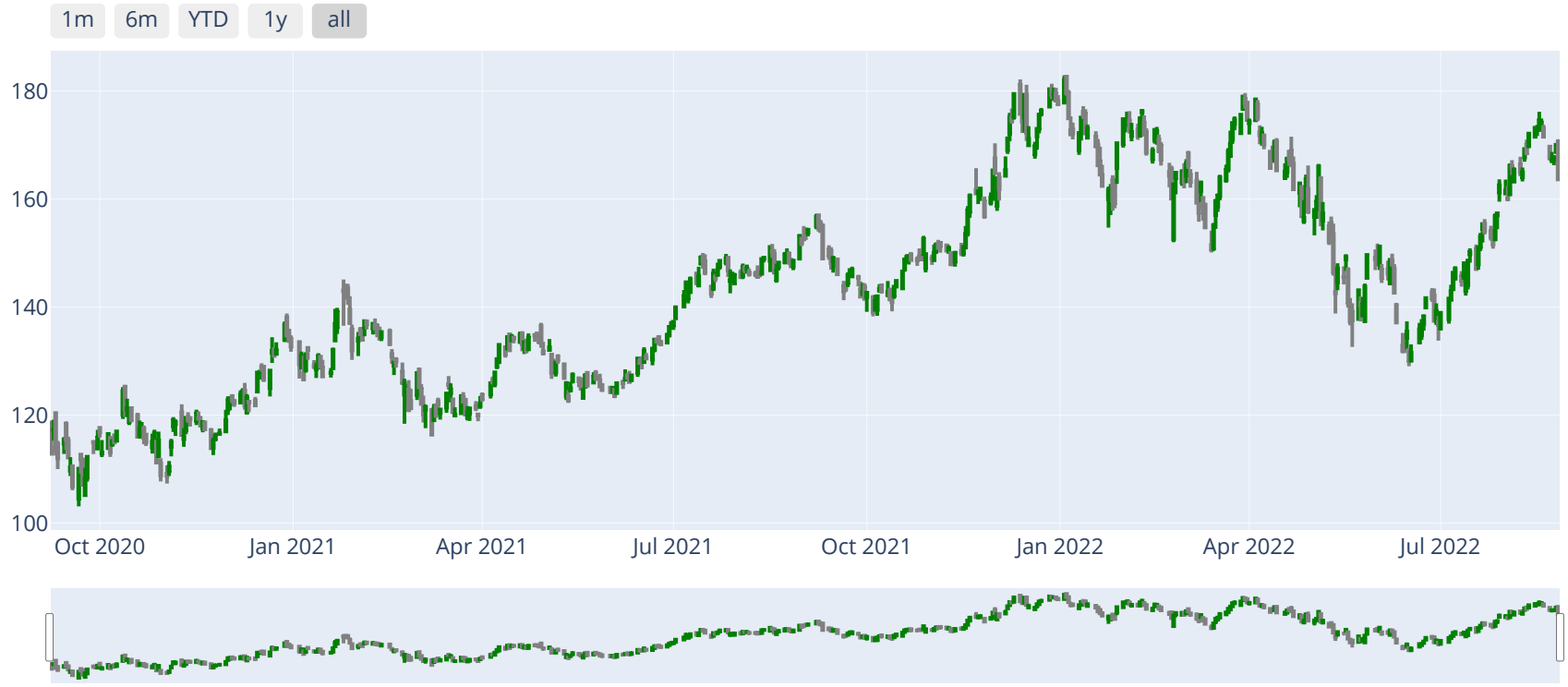


In [12]:
```
# analyzing and narrowing down the chart between two specific dates
figure_m = px.line(data_m, x = data_m.index,
                   y = 'Close',
                   range_x = ['2022-01-01','2022-02-01'],
                   title = "Time Series Analysis Microsoft (Custom Date Range)")
figure_m.show()
```



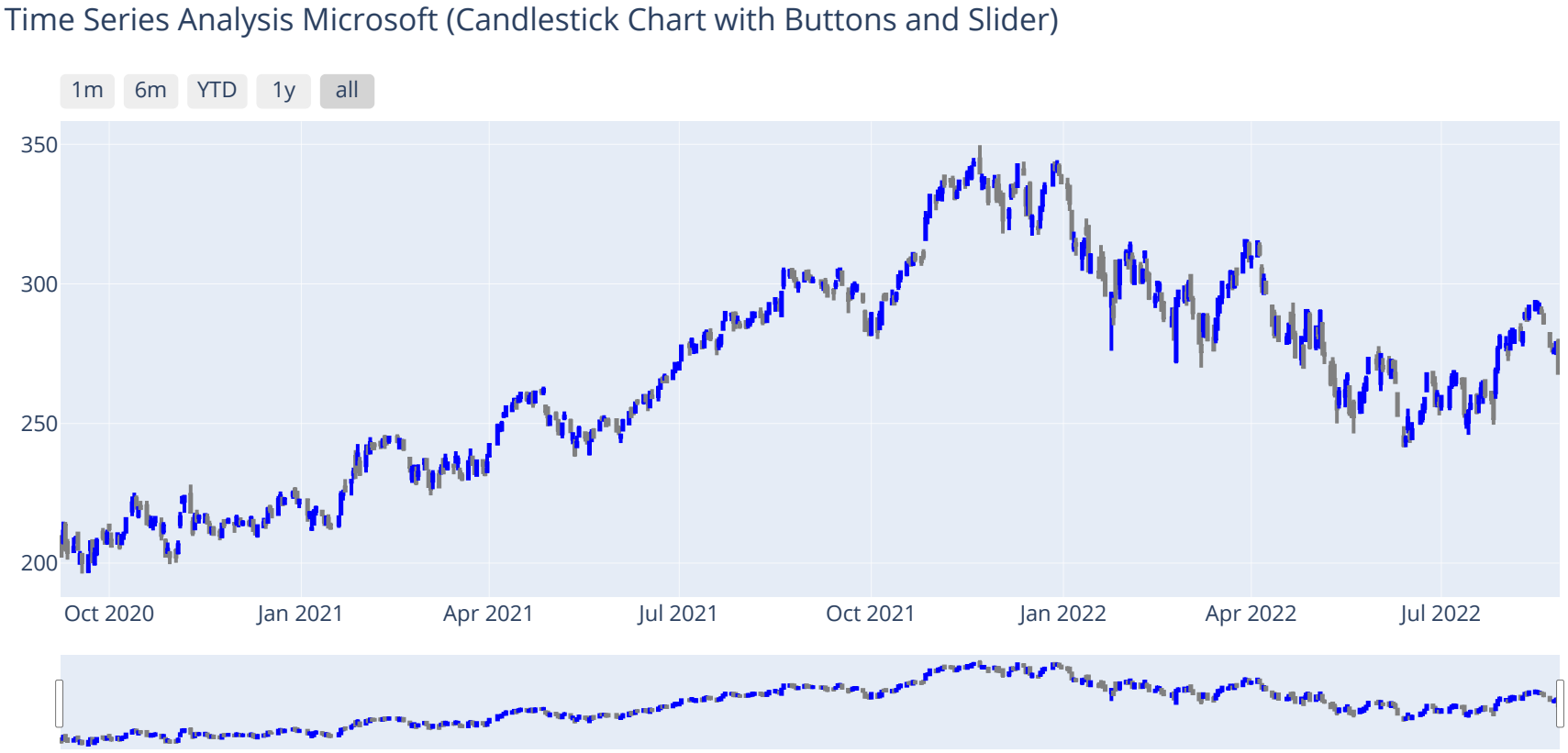
In [16]:
```
#adding slicers to one of the charts to make it even more interactive
#better suited for financial organizations


figure_a = go.Figure(data = [go.Candlestick(x = data_a.index,
                                            open = data_a["Open"],
                                            high = data_a["High"],
                                            low = data_a["Low"],
                                            close = data_a["Close"],
                                            increasing_line_color= 'green', decreasing_line_color= 'gray')])
figure_a.update_layout(title = "Time Series Analysis Apple (Candlestick Chart with Buttons and Slider)")

figure_a.update_xaxes(
    rangeslider_visible = True,
    rangeselector = dict(
        buttons = list([
            dict(count = 1, label = "1m", step = "month", stepmode = "backward"),
            dict(count = 6, label = "6m", step = "month", stepmode = "backward"),
            dict(count = 1, label = "YTD", step = "year", stepmode = "todate"),
            dict(count = 1, label = "1y", step = "year", stepmode = "backward"),
            dict(step = "all")
        ])
    )
)
figure_a.show()
```

Time Series Analysis Apple (Candlestick Chart with Buttons and Slider)



In [15]:

```
#adding slicers to one of the charts to make it even more interactive
#better suited for financial organizations


figure_m = go.Figure(data = [go.Candlestick(x = data_m.index,
                                            open = data_m["Open"],
                                            high = data_m["High"],
                                            low = data_m["Low"],
                                            close = data_m["Close"],
                                            increasing_line_color= 'blue', decreasing_line_color= 'gray')])
figure_m.update_layout(title = "Time Series Analysis Microsoft (Candlestick Chart with Buttons and Slider)")

figure_m.update_xaxes(
    rangeslider_visible = True,
    rangeselector = dict(
        buttons = list([
            dict(count = 1, label = "1m", step = "month", stepmode = "backward"),
            dict(count = 6, label = "6m", step = "month", stepmode = "backward"),
            dict(count = 1, label = "YTD", step = "year", stepmode = "todate"),
            dict(count = 1, label = "1y", step = "year", stepmode = "backward"),
            dict(step = "all")
        ])
    )
)
figure_m.show()
```


Time Series Analysis Microsoft (Candlestick Chart with Buttons and Slider)
1m
6m
YTD
1y
all
350
300
250
200
Oct 2020
Jan 2021
Apr 2021
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022

In [ ]: