In [27]:
```
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-p
ackages (0.1.74)
Requirement already satisfied: requests>=2.26 in c:\users\marjan\anaconda3\lib
\site-packages (from yfinance) (2.28.1)
Requirement already satisfied: pandas>=0.24.0 in c:\users\marjan\anaconda3\lib
\site-packages (from yfinance) (0.25.1)
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\sit
e-packages (from yfinance) (1.16.5)
Requirement already satisfied: lxml>=4.5.1 in c:\users\marjan\anaconda3\lib\sit
e-packages (from yfinance) (4.9.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\marjan\anacond
a3\lib\site-packages (from requests>=2.26->yfinance) (1.24.2)
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: idna<4,>=2.5 in c:\users\marjan\anaconda3\lib\si
te-packages (from requests>=2.26->yfinance) (2.8)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\marjan\anac
onda3\lib\site-packages (from requests>=2.26->yfinance) (2.1.1)
Requirement already satisfied: pytz>=2017.2 in c:\users\marjan\anaconda3\lib\si
te-packages (from pandas>=0.24.0->yfinance) (2019.3)
Requirement already satisfied: python-dateutil>=2.6.1 in c:\users\marjan\anacon
da3\lib\site-packages (from pandas>=0.24.0->yfinance) (2.8.0)
Requirement already satisfied: six>=1.5 in c:\users\marjan\anaconda3\lib\site-p
ackages (from python-dateutil>=2.6.1->pandas>=0.24.0->yfinance) (1.12.0)
Note: you may need to restart the kernel to use updated packages.
```

In [28]:
```
today = date.today()
today
```

Out[28]:
```
datetime.date(2022, 8, 27)
```

In [29]:
```
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 [30]:

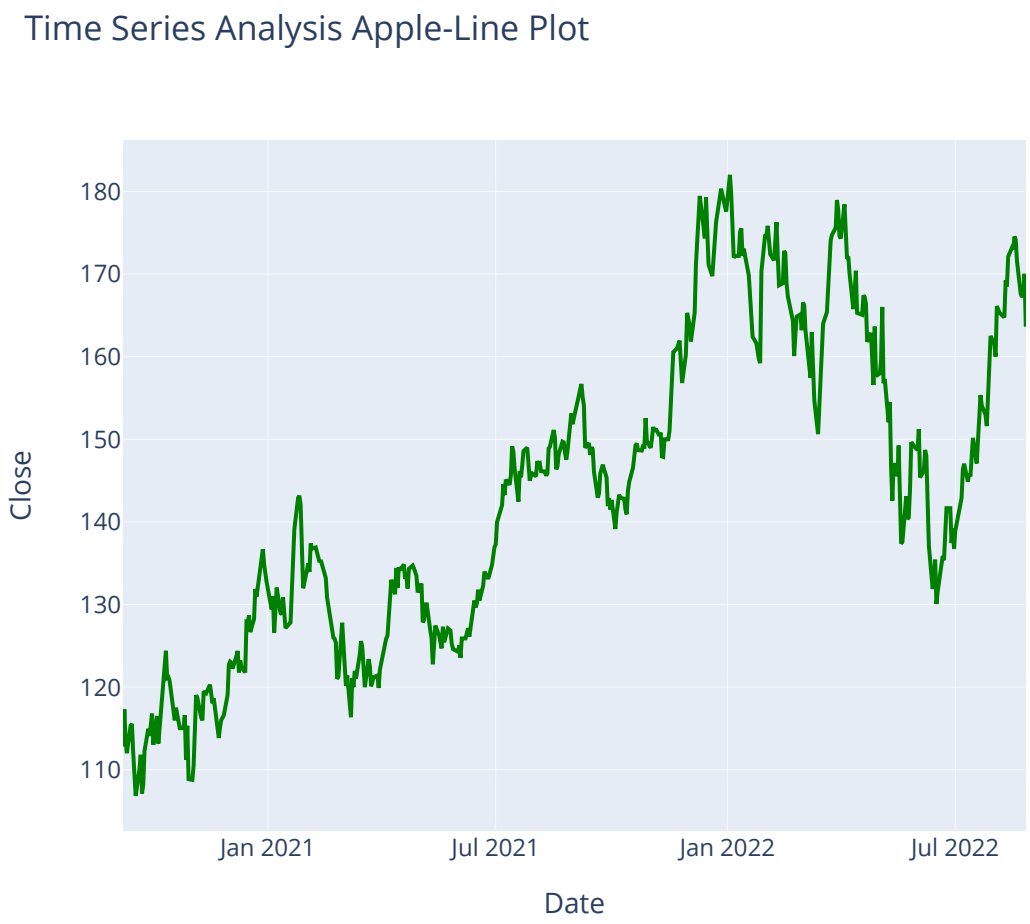
```
#Extracting the lates stock price of Apple by using the yfinance (Yahoo Fina
data_a = yf.download('AAPL',
                     start = start_date,
                     end = end_date,
                     progress=False)
print(data_a.head())


#Extracting the lates stock price of Microsoft by using the yfinance (Yahoo
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  115.920670
2020-09-10  120.360001  120.500000  112.500000  113.489998  112.136345
2020-09-11  114.570000  115.230003  110.000000  112.000000  110.664124
2020-09-14  114.720001  115.930000  112.800003  115.360001  113.984039

               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  199.149414
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  201.851776

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

In [50]:

```
# 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 one ach 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()
```

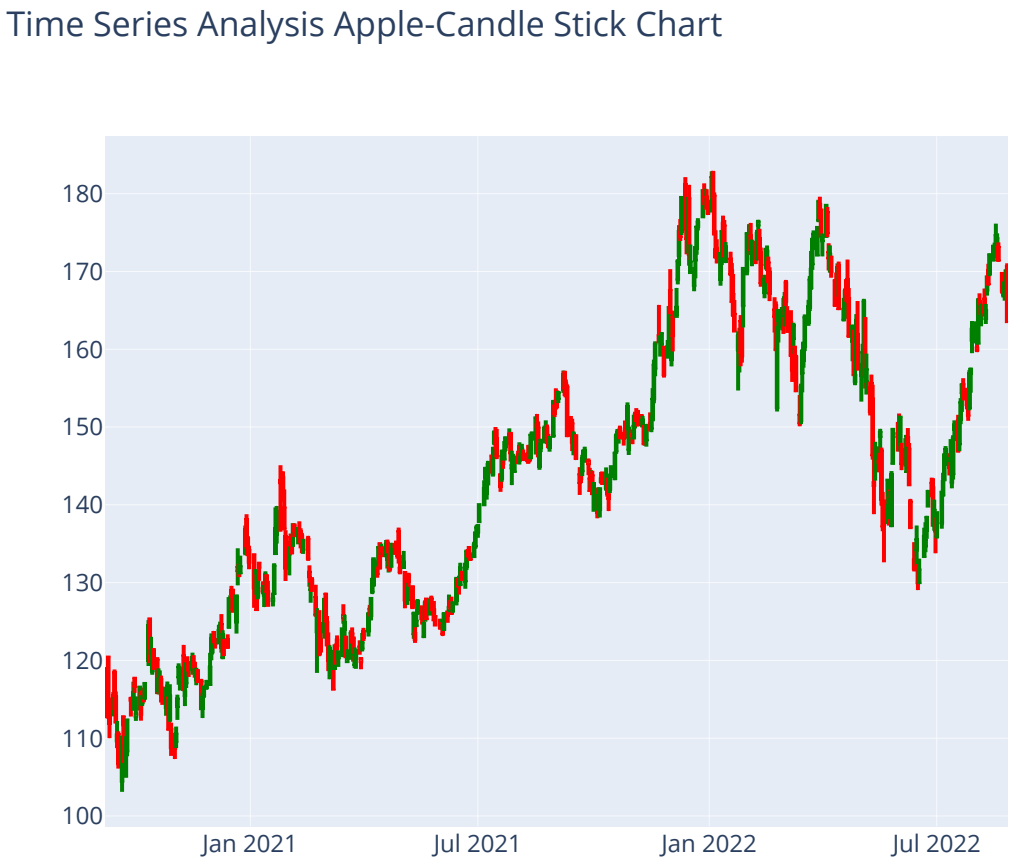
```
Requirement already satisfied: plotly in c:\users\marjan\anaconda3\lib\site-pac
kages (5.10.0)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\marjan\anaconda3\lib
\site-packages (from plotly) (8.0.1)
Note: you may need to restart the kernel to use updated packages.
```



Time Series Analysis Microsoft-Line Plot

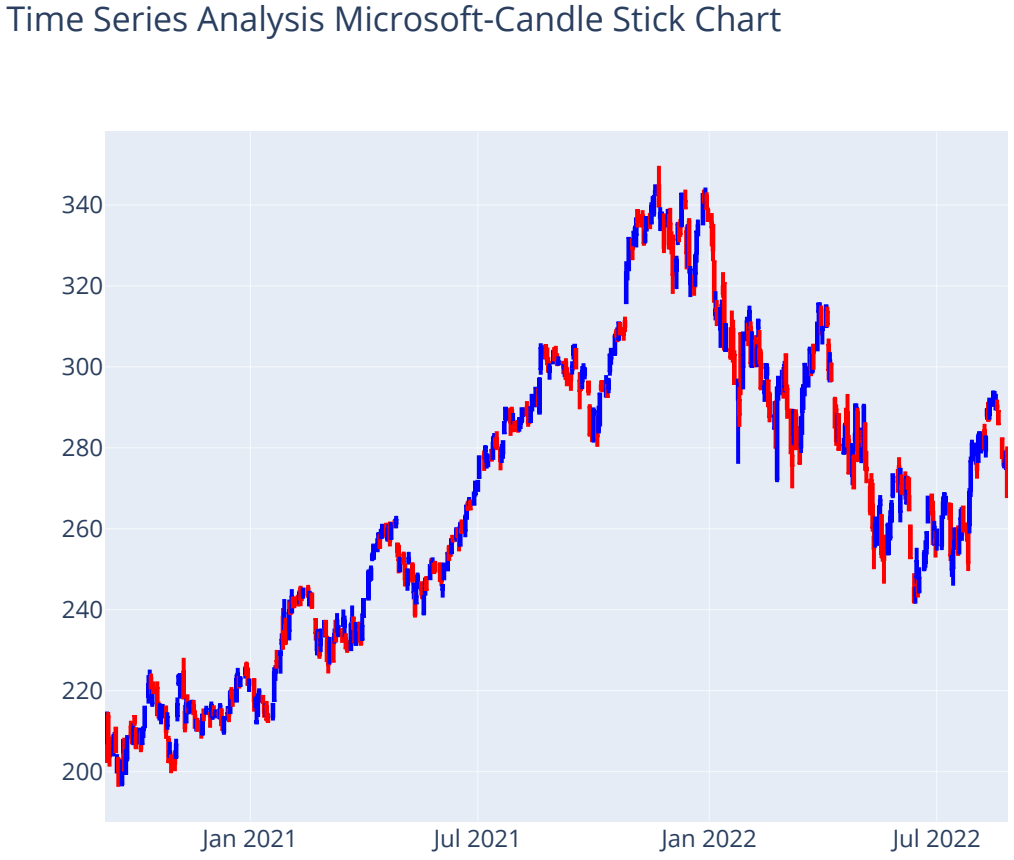
In [56]:

```
#more visualization this time on open, high, low, and close prices all toget
#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 Char
                       xaxis_rangeslider_visible = False)
#figure_a.show()

#The red 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 = 'red'
cs.decreasing.line.color = 'red'

figure_a.show()
```



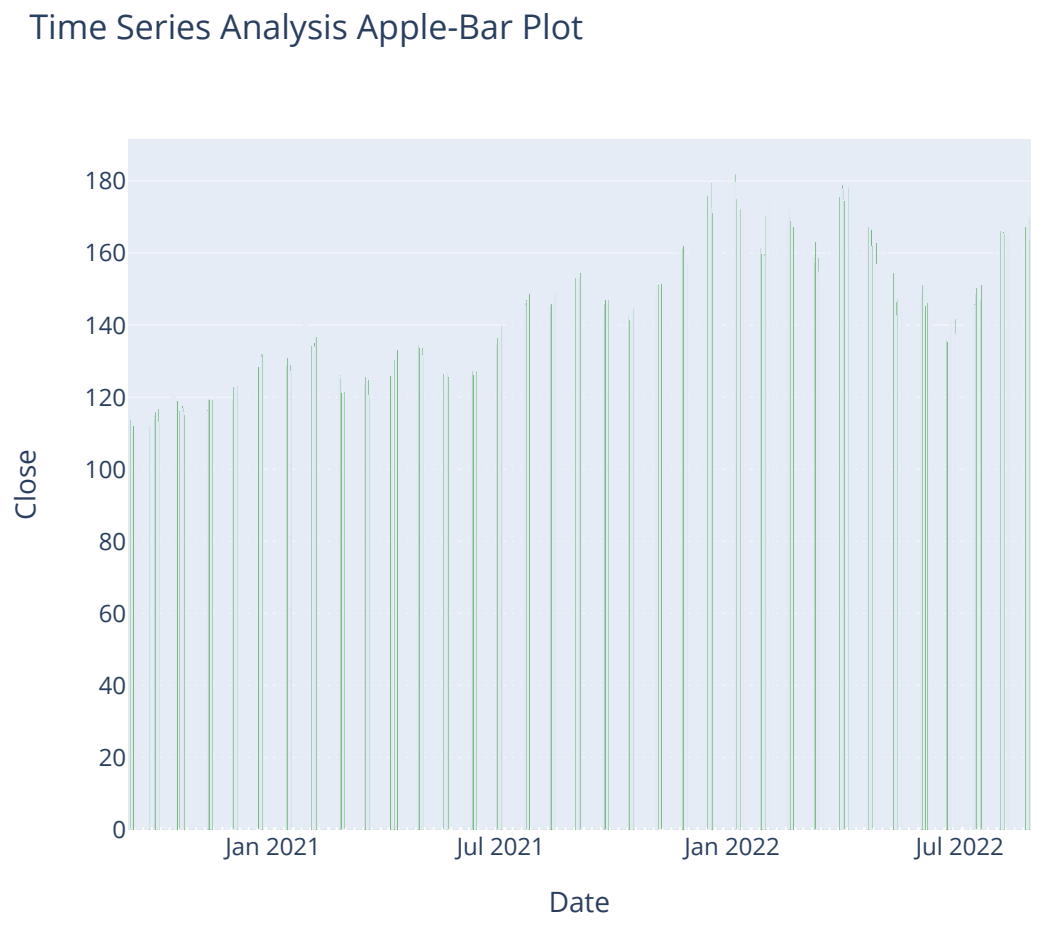
In [57]:

```
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
                       xaxis_rangeslider_visible = False)
#figure_m.show()

cs = figure_m.data[0]

# Set line and fill colors
# The red 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 = 'red'
cs.decreasing.line.color = 'red'

figure_m.show()
```



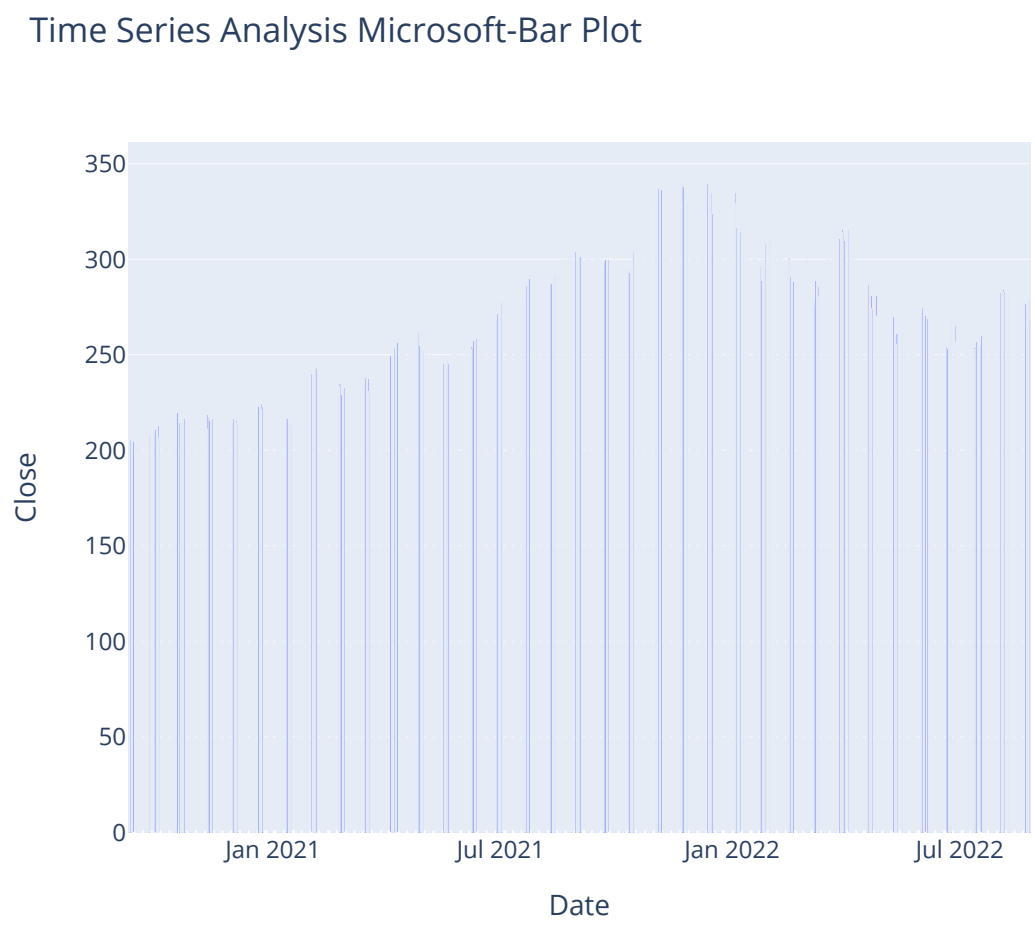
In [48]:

```
#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()

```



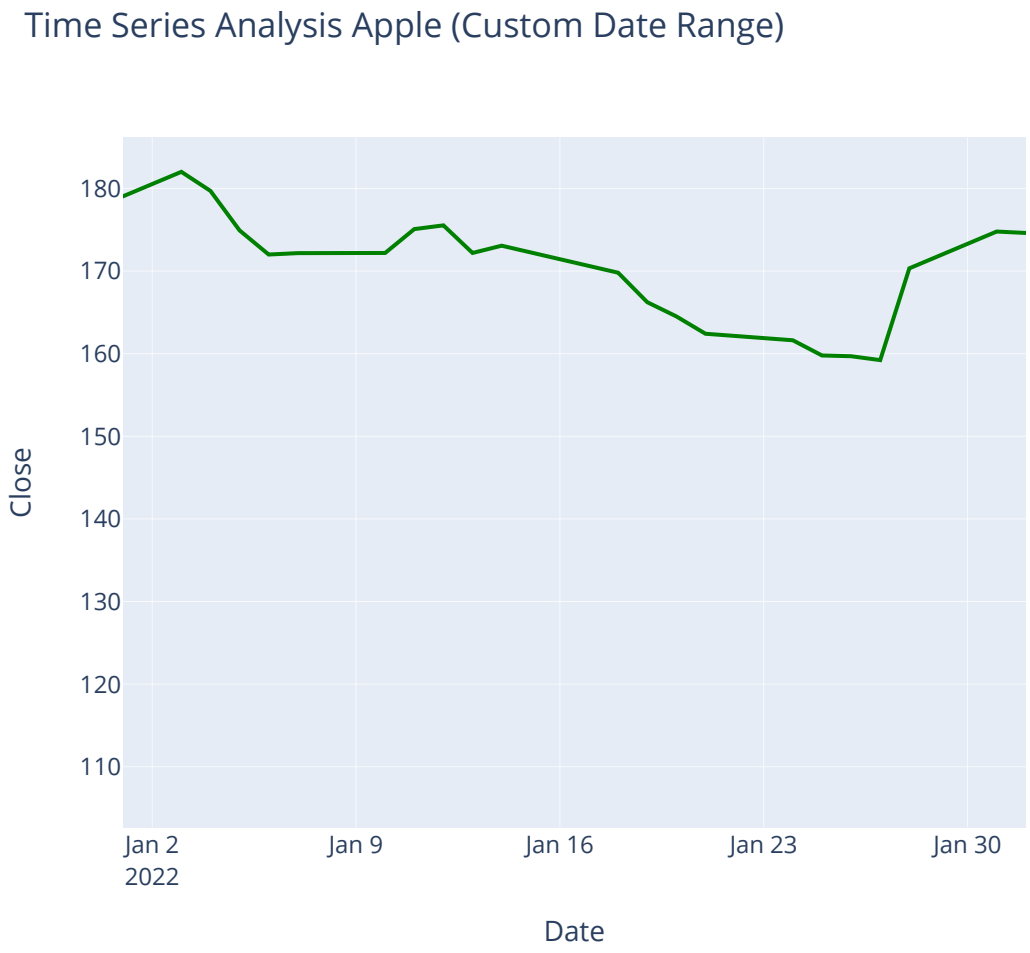
In [44]:

```
#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()
```



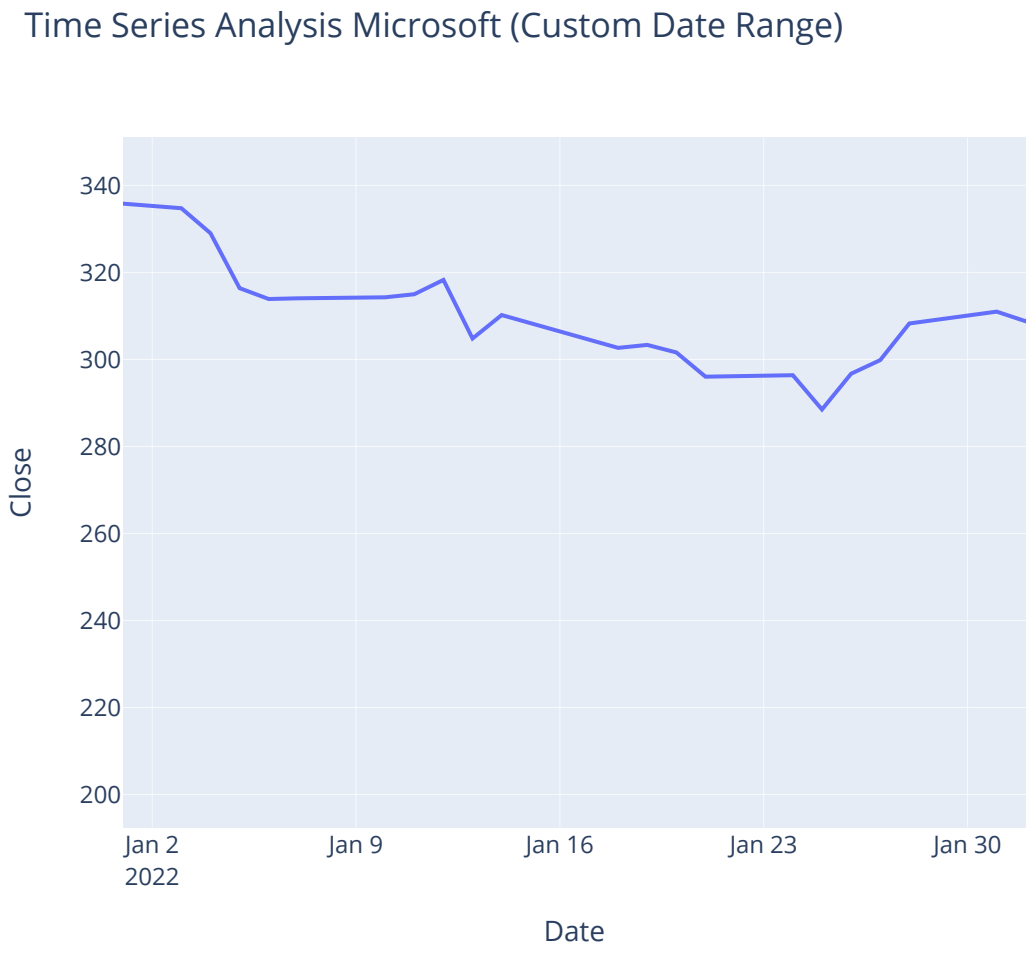
In [49]:

```
# 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()
```



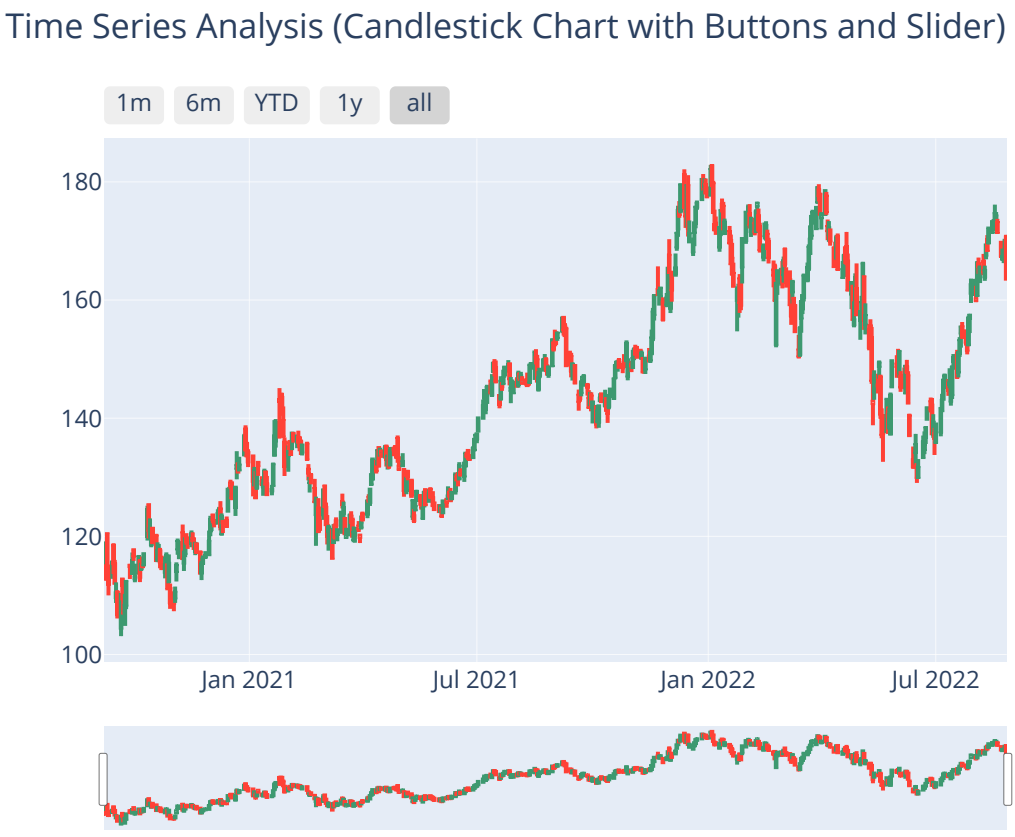
In [45]:

```
# 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()
```



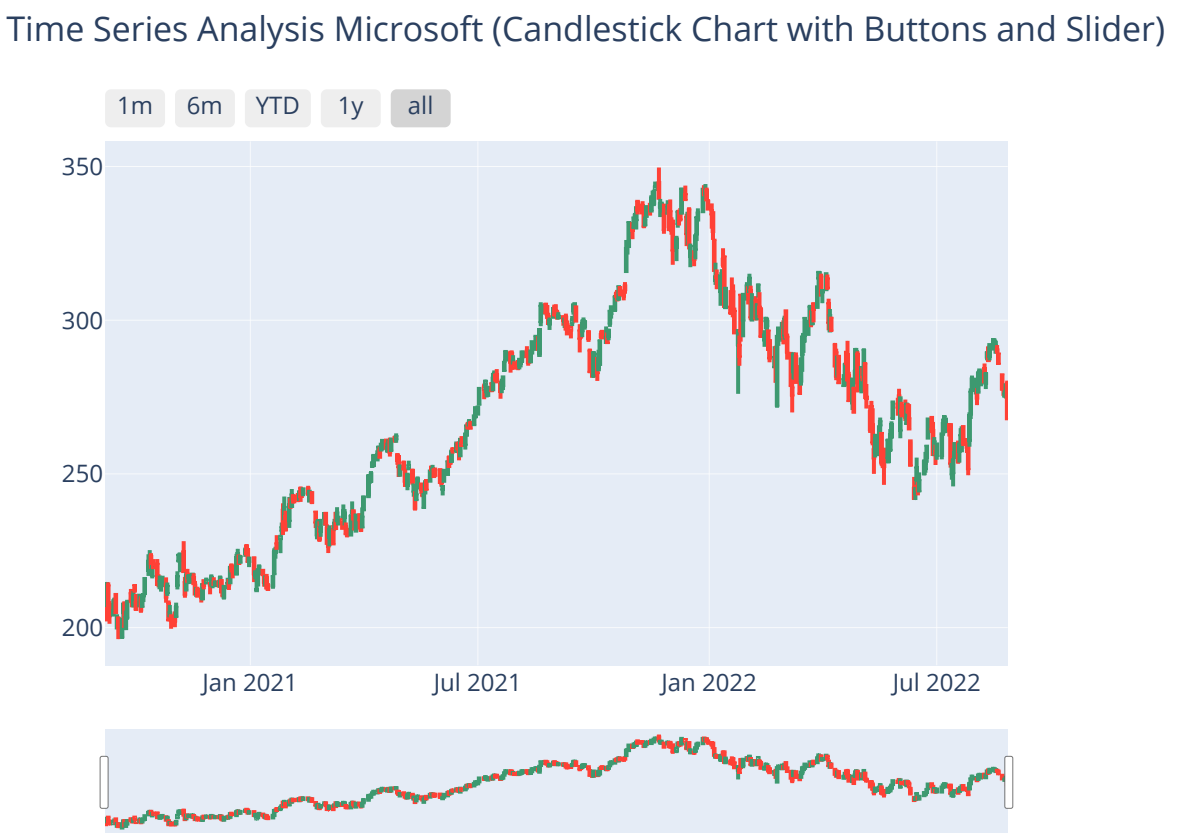
In [42]:

```
#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"])])
figure_a.update_layout(title = "Time Series Analysis Apple (Candlestick Char

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



In [47]:

```
#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"])])
figure_m.update_layout(title = "Time Series Analysis Microsoft (Candlestick

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



In [ ]:

```
1
```