Amazon (AMZN) Stock Price prediction using Facebook Prophet

In [1]:
```
## Switch to GPU mode for faster Computation (Runtime> Change runtime> GPU)
```

Importing all the necessary Libraries

In [2]:
```
#Necessary libraries = Pandas, fbprophet and plotly

#pandas= data Manipulation and analysis
#fbprophet = Forecasting
#plotly= data visualization
#!pip install prophet
```

In [3]:
```
import pandas as pd
import plotly.express as px
#from fbprophet import Prophet
from prophet import Prophet
#
#import ipywidgets as widgets
```

In [4]:
```
 #Initializing Plotly
import plotly.io as pio
pio.renderers.default='colab'
```

Importing the Dataset & Exploring it

In [5]:
```
df = pd.read_csv('AMZN.csv')
```

In [6]:
```
#read_csv function from pandas
```

In [7]:
```
df
```

Out[7]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2022-12-12	89.209999	90.580002	87.870003	90.550003	90.550003	61999800
1	2022-12-13	95.230003	96.250000	90.519997	92.489998	92.489998	100212000
2	2022-12-14	92.500000	93.459999	89.870003	91.580002	91.580002	70298000
3	2022-12-15	89.889999	89.970001	87.470001	88.449997	88.449997	84802900
4	2022-12-16	88.269997	89.349998	86.730003	87.860001	87.860001	146144100
...	...	...	...	...	...	...	...
247	2023-12-06	147.580002	147.850006	144.279999	144.520004	144.520004	39679000
248	2023-12-07	146.149994	147.919998	145.339996	146.880005	146.880005	52352800
249	2023-12-08	145.479996	147.839996	145.399994	147.419998	147.419998	41858000
250	2023-12-11	145.660004	146.190002	143.639999	145.889999	145.889999	50907300
251	2023-12-12	145.520004	147.500000	145.300003	147.479996	147.479996	44886600

252 rows × 7 columns

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 252 entries, 0 to 251
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       252 non-null    object 
 1   Open       252 non-null    float64
 2   High       252 non-null    float64
 3   Low        252 non-null    float64
 4   Close      252 non-null    float64
 5   Adj Close  252 non-null    float64
 6   Volume     252 non-null    int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 13.9+ KB
```

In [9]:
```
df.describe()
```

Out[9]:

	Open	High	Low	Close	Adj Close	Volume
count	252.000000	252.000000	252.000000	252.000000	252.000000	2.520000e+02
mean	117.831072	119.368730	116.353016	117.966469	117.966469	6.029315e+07
std	19.125679	19.142427	19.115752	19.131681	19.131681	2.135646e+07
min	82.800003	83.480003	81.430000	81.820000	81.820000	2.237840e+07
25%	99.725001	101.044998	98.094999	100.010000	100.010000	4.643740e+07
50%	124.774998	126.730003	123.689999	125.000000	125.000000	5.564410e+07
75%	133.597504	134.699997	131.982506	133.334996	133.334996	6.792652e+07
max	147.850006	149.259995	146.880005	147.729996	147.729996	1.581542e+08

Data Visualization using plotly express- Visualizing the historical performance of Tesla

In [10]:
```
#Line graph, Area graph , box plot (Analyzing price and volume)
```

In [11]:
```
px.area(df, x='Date',y='Close')
```

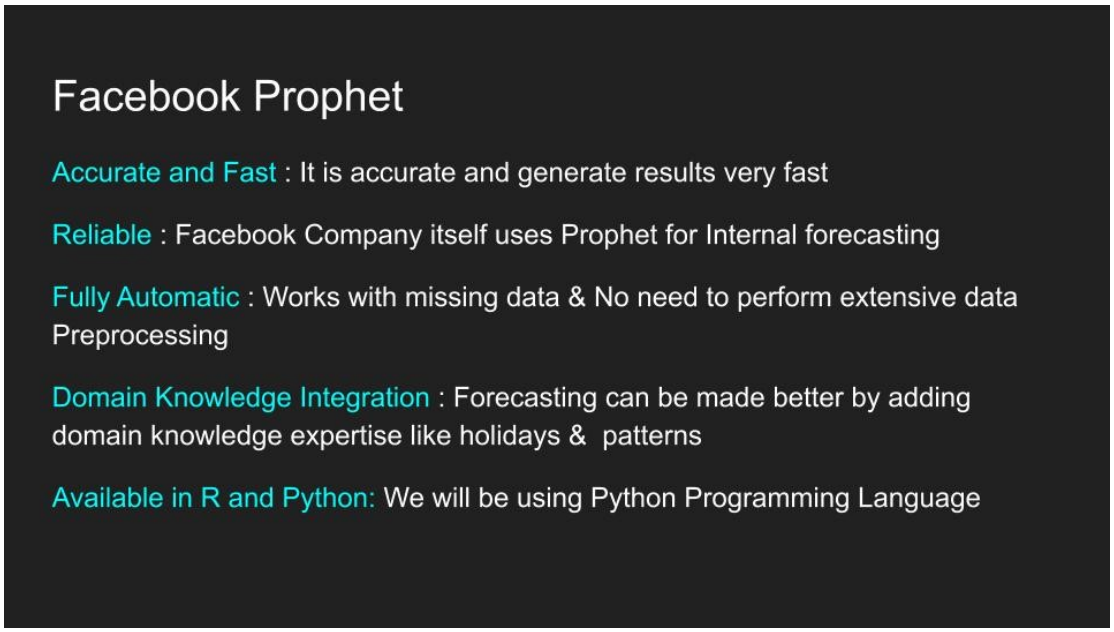
In [12]:
```
px.line(df, x='Date',y='Close')
```

In [13]:
```
px.area(df, x='Date',y='Volume')
```

In [14]:
```
px.bar(df, y='Volume')
```

In [15]:
```
px.box(df, y='Close')
```

Understanding Facebook Prophet



Data Preperation

In [16]:
```
df
```

Out[16]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2022-12-12	89.209999	90.580002	87.870003	90.550003	90.550003	61999800
1	2022-12-13	95.230003	96.250000	90.519997	92.489998	92.489998	100212000
2	2022-12-14	92.500000	93.459999	89.870003	91.580002	91.580002	70298000
3	2022-12-15	89.889999	89.970001	87.470001	88.449997	88.449997	84802900
4	2022-12-16	88.269997	89.349998	86.730003	87.860001	87.860001	146144100
...	...	...	...	...	...	...	...
247	2023-12-06	147.580002	147.850006	144.279999	144.520004	144.520004	39679000
248	2023-12-07	146.149994	147.919998	145.339996	146.880005	146.880005	52352800
249	2023-12-08	145.479996	147.839996	145.399994	147.419998	147.419998	41858000
250	2023-12-11	145.660004	146.190002	143.639999	145.889999	145.889999	50907300
251	2023-12-12	145.520004	147.500000	145.300003	147.479996	147.479996	44886600

252 rows × 7 columns

In [17]:
```
columns = ['Date','Close']
```

In [18]:
```
ndf = pd.DataFrame(df, columns = columns)
```

In [19]:
```
ndf
```

Out[19]:

	Date	Close
0	2022-12-12	90.550003
1	2022-12-13	92.489998
2	2022-12-14	91.580002
3	2022-12-15	88.449997
4	2022-12-16	87.860001
...	...	...
247	2023-12-06	144.520004
248	2023-12-07	146.880005
249	2023-12-08	147.419998
250	2023-12-11	145.889999
251	2023-12-12	147.479996

252 rows × 2 columns

In [20]:
```
prophet_df = ndf.rename(columns = {'Date':'ds', 'Close':'y'})
```

In [21]:
```
prophet_df
```

Out[21]:

	ds	y
0	2022-12-12	90.550003
1	2022-12-13	92.489998
2	2022-12-14	91.580002
3	2022-12-15	88.449997
4	2022-12-16	87.860001
...	...	...
247	2023-12-06	144.520004
248	2023-12-07	146.880005
249	2023-12-08	147.419998
250	2023-12-11	145.889999
251	2023-12-12	147.479996

252 rows × 2 columns

Creating Facebook Prophet Model

In [22]:
```
m = Prophet()
m.fit(prophet_df)
```

```
14:51:56 - cmdstanpy - INFO - Chain [1] start processing
14:51:56 - cmdstanpy - INFO - Chain [1] done processing
```

Out[22]: <prophet.forecaster.Prophet at 0x174b57e2900>

Forecasting

In [23]:
```
future = m.make_future_dataframe(periods=30) # 30: 30 days
future.tail()
```

Out[23]:

	ds
277	2024-01-07
278	2024-01-08
279	2024-01-09
280	2024-01-10
281	2024-01-11

In [24]:
```
forecast = m.predict(future)
forecast
```

Out[24]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additiv
0	2022-12-12	87.742902	81.764270	93.507129	87.742902	87.742902	-C
1	2022-12-13	87.895196	81.200064	93.329339	87.895196	87.895196	-C
2	2022-12-14	88.047490	81.866734	93.753686	88.047490	88.047490	-C
3	2022-12-15	88.199784	81.394424	93.450149	88.199784	88.199784	-C
4	2022-12-16	88.352079	82.073670	94.367781	88.352079	88.352079	-C
...	...	...	...	...	...	...	
277	2024-01-07	150.871199	145.895001	158.548315	149.766478	152.013956	1
278	2024-01-08	151.055727	144.717439	156.587977	149.878325	152.287437	-C
279	2024-01-09	151.240254	144.696398	156.666642	149.972061	152.573645	-C
280	2024-01-10	151.424782	144.736128	156.868868	150.086819	152.850873	-C
281	2024-01-11	151.609309	145.210476	157.095248	150.198002	153.099429	-C

282 rows × 16 columns

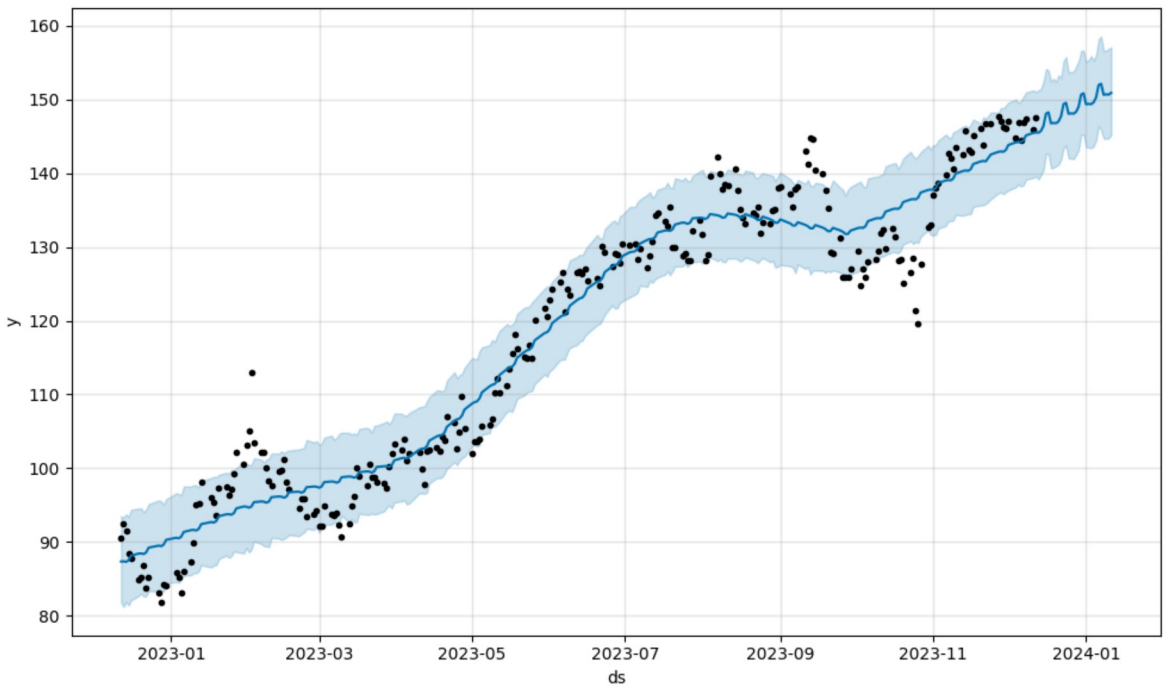
In [25]:
```
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

Out[25]:

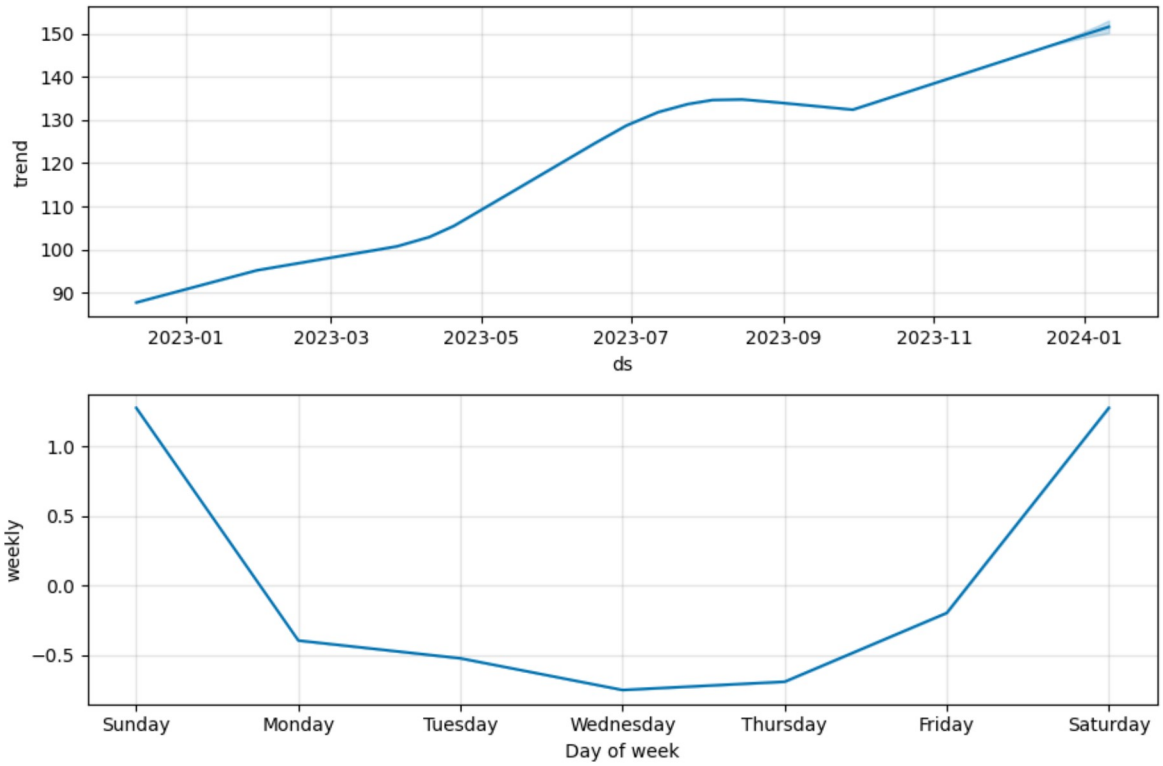
	ds	yhat	yhat_lower	yhat_upper
277	2024-01-07	152.150217	145.895001	158.548315
278	2024-01-08	150.660423	144.717439	156.587977
279	2024-01-09	150.717073	144.696398	156.666642
280	2024-01-10	150.673607	144.736128	156.868868
281	2024-01-11	150.917661	145.210476	157.095248

In [26]:
```
px.line(forecast, x='ds', y='yhat')
```

In [27]:
```
figure = m.plot(forecast, xlabel='ds',ylabel = 'y')
```



In [28]:
```
figure2 = m.plot_components(forecast)
```



In [29]:
```
from prophet.plot import plot_plotly, plot_components_plotly

plot_plotly(m, forecast)
```

In [30]:
```
plot_components_plotly(m, forecast)
```

Downloading the Forecast data

In [31]:
```
#from google.colab import files
#forecast.to_csv('forecast.csv')
#files.download('forecast.csv')
```

In [32]:
```
#forecast.to_csv('forecast.csv', index=False)
forecast.to_csv('forecast.csv', index=True)
```