In [11]:
```
#Import Needed Liberaries:
import numpy as np
import pandas as pd
import datetime
from datetime import date , timedelta
import plotly.graph_objects as go
import plotly.express as px
import plotly.io as pio

pio.templates.default = "plotly_white"
```

In [12]:
```
#Import our Data:
data = pd.read_csv("TWTR.csv")
data.head()
```

Out[12]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2013-11-07	45.099998	50.090000	44.000000	44.900002	44.900002	117701670.0
1	2013-11-08	45.930000	46.939999	40.685001	41.650002	41.650002	27925307.0
2	2013-11-11	40.500000	43.000000	39.400002	42.900002	42.900002	16113941.0
3	2013-11-12	43.660000	43.779999	41.830002	41.900002	41.900002	6316755.0
4	2013-11-13	41.029999	42.869999	40.759998	42.599998	42.599998	8688325.0

In [13]:
```
# Lets get some information on the data types of the columns and rows
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2264 entries, 0 to 2263
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       2264 non-null   object 
 1   Open       2259 non-null   float64
 2   High       2259 non-null   float64
 3   Low        2259 non-null   float64
 4   Close      2259 non-null   float64
 5   Adj Close  2259 non-null   float64
 6   Volume     2259 non-null   float64
dtypes: float64(6), object(1)
memory usage: 123.9+ KB
```

In [14]:
```
# Lets see the null values in our data
data.isnull().sum()
```

Out[14]:
```
Date         0
Open         5
High         5
Low          5
Close        5
Adj Close    5
Volume       5
dtype: int64
```

In [15]:
```
# Drop all the null values in the data
data = data.dropna()
```

In [16]:
```
# Visualize our data in the form of candlesticks:
figure = go.Figure(data =[go.Candlestick( x = data['Date'],
                                          open = data['Open'],
                                          close = data['Close'],
                                          high = data['High'],
                                          low = data['Low'])])
figure.update_layout(title = "Twitter Stock Price Over the years" , xaxis_rangeslider_visible = False )
figure.show()
```

In [20]:
```
#lets see the price of the stock a bit closer specially in 2021 when it starts to be profitable:
figure = px.bar(data, x = "Date", y= "Close", color = "Close" )
figure.update_xaxes(rangeslider_visible = True)
figure.show()
```

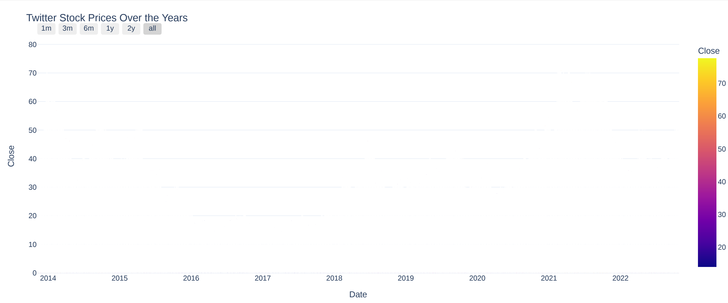
In [22]:
```
# lets update our visualization to more specific level including year and month:
# Create the bar chart
figure = px.bar(data, x="Date", y="Close", color="Close")

# Update x-axis with range slider
figure.update_xaxes(rangeslider_visible=True)

# Update layout with title and hide x-axis rangeslider
figure.update_layout(
    title="Twitter Stock Prices Over the Years",
    xaxis_rangeslider_visible=False)

# Update x-axis with rangeselector
figure.update_xaxes(
    rangeselector=dict(
        buttons=list([
            dict(count=1, label="1m", step="month", stepmode="backward"),
            dict(count=3, label="3m", step="month", stepmode="backward"),
            dict(count=6, label="6m", step="month", stepmode="backward"),
            dict(count=1, label="1y", step="year", stepmode="backward"),
            dict(count=2, label="2y", step="year", stepmode="backward"),
            dict(step="all")])))

# Show the figure
figure.show()
```



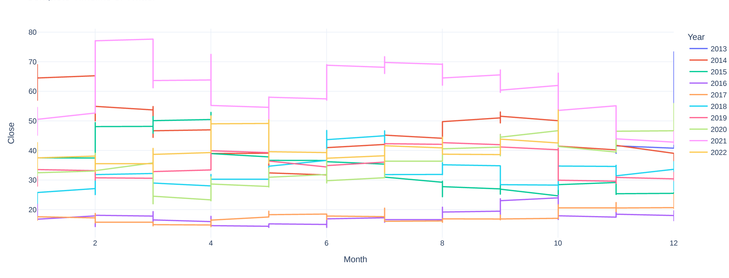
In [24]:
```
#lets visualize the trend of Twitter stock prices over time:

# Convert 'Date' column to datetime format
data["Date"] = pd.to_datetime(data["Date"], format='%Y-%m-%d')

# Extract 'Year' and 'Month' from the 'Date' column
data['Year'] = data['Date'].dt.year
data["Month"] = data["Date"].dt.month

# Create line chart
fig = px.line(data, x="Month", y="Close", color='Year', title="Complete Timeline of Twitter")

# Show the figure
fig.show()
```



In [ ]: