Time Series EDA

In [4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [5]:
```
df_tesla = yf.download('TSLA')
df_tesla.head()
```

```
[**********100%%**********]  1 of 1 completed
```

Out[5]:

Date	Open	High	Low	Close	Adj Close	Volume
2010-06-29	1.266667	1.666667	1.169333	1.592667	1.592667	281494500
2010-06-30	1.719333	2.028000	1.553333	1.588667	1.588667	257806500
2010-07-01	1.666667	1.728000	1.351333	1.464000	1.464000	123282000
2010-07-02	1.533333	1.540000	1.247333	1.280000	1.280000	77097000
2010-07-06	1.333333	1.333333	1.055333	1.074000	1.074000	103003500

In [6]:
```
df_tesla.tail()
```

Out[6]:

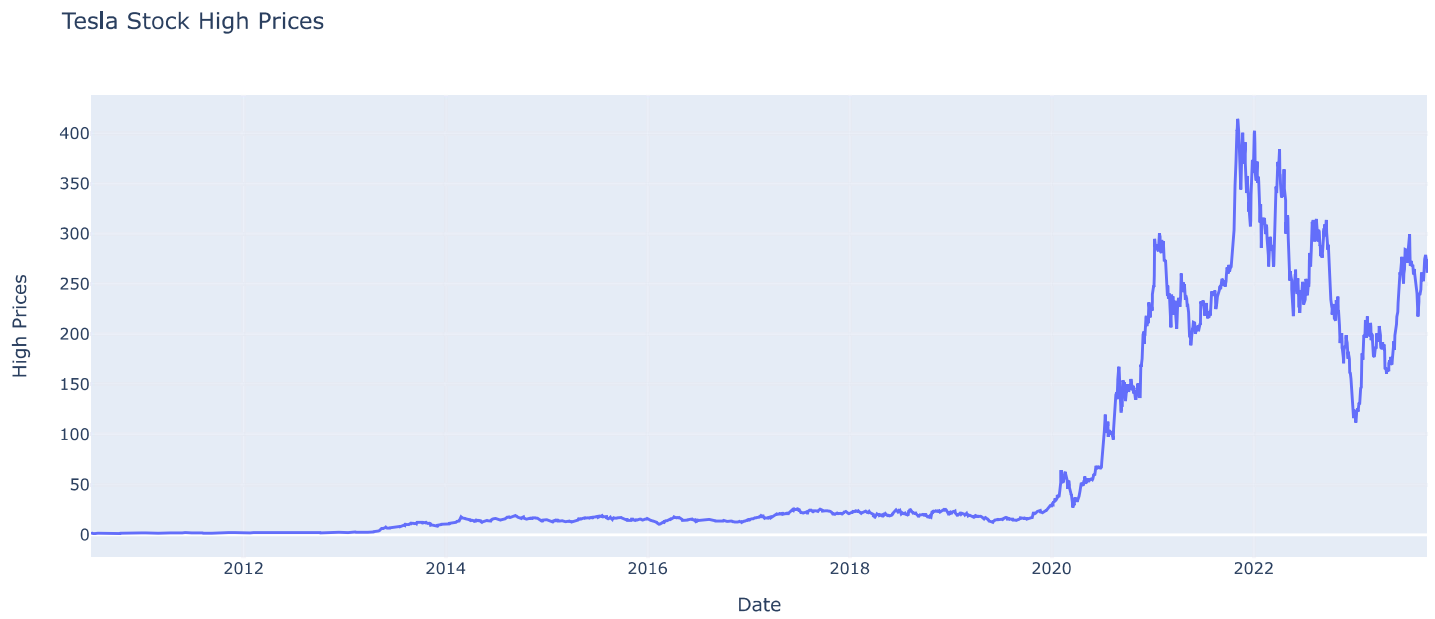
Date	Open	High	Low	Close	Adj Close	Volume
2023-09-15	277.549988	278.980011	271.000000	274.390015	274.390015	133422800
2023-09-18	271.160004	271.440002	263.760010	265.279999	265.279999	101543300
2023-09-19	264.350006	267.850006	261.200012	266.500000	266.500000	103704000
2023-09-20	267.040009	273.929993	262.459991	262.589996	262.589996	122225500
2023-09-21	257.850006	260.859985	254.210007	258.989899	258.989899	87036416

In [7]:
```
df_tesla.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 3331 entries, 2010-06-29 to 2023-09-21
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Open       3331 non-null   float64
 1   High       3331 non-null   float64
 2   Low        3331 non-null   float64
 3   Close      3331 non-null   float64
 4   Adj Close  3331 non-null   float64
 5   Volume     3331 non-null   int64  
dtypes: float64(5), int64(1)
memory usage: 182.2 KB
```

In [8]:
```
#plotting

fig = Figure(data=[Scatter(x=df_tesla.index, y = df_tesla['High'], mode ='lines', name='High')])

fig.update_layout(title = 'Tesla Stock High Prices',
                  xaxis_title ='Date',
                  yaxis_title = 'High Prices')
```

```
fig.show()
```



In [9]:
```
#xlimt and ylimt

fig = Figure(data=[Scatter(x=df_tesla.index, y = df_tesla['High'], mode ='lines', name='High')])

fig.update_layout(title = 'Tesla Stock High Prices',
                  xaxis_title ='Date',
                  yaxis_title = 'High Prices',
                  xaxis = dict(range=['2022-01-01', '2023-08-25']))

fig.show()
```

In [10]:
```
#xlimt and ylimt

fig = Figure(data=[Scatter(x=df_tesla.index, y = df_tesla['High'], mode ='lines', name='High', line=dict(color ='green'))])

fig.update_layout(title = 'Tesla Stock High Prices',
                  xaxis_title ='Date',
                  yaxis_title = 'High Prices',
                  xaxis = dict(range=['2022-01-01', '2023-08-25']))

fig.show()
```

In [11]:
```
df_tesla.head()
```

Out[11]:

Date	Open	High	Low	Close	Adj Close	Volume
2010-06-29	1.266667	1.666667	1.169333	1.592667	1.592667	281494500
2010-06-30	1.719333	2.028000	1.553333	1.588667	1.588667	257806500
2010-07-01	1.666667	1.728000	1.351333	1.464000	1.464000	123282000
2010-07-02	1.533333	1.540000	1.247333	1.280000	1.280000	77097000
2010-07-06	1.333333	1.333333	1.055333	1.074000	1.074000	103003500

In [12]:
```
fig = Figure()

fig.add_trace(Scatter(x=df_tesla.index, y=df_tesla['Open'], mode='lines', name='Open', line=dict(color='blue')))
fig.add_trace(Scatter(x=df_tesla.index, y=df_tesla['Close'], mode='lines', name='Close', line=dict(color='red')))

fig.update_layout(title='Tesla Open and Close Prices',
                  xaxis = dict(range=['2022-01-01', '2023-08-25']))

fig.show()
```

Time Resampling

In [13]:
```
df_tesla.head()
```

Out[13]:

Date	Open	High	Low	Close	Adj Close	Volume
2010-06-29	1.266667	1.666667	1.169333	1.592667	1.592667	281494500
2010-06-30	1.719333	2.028000	1.553333	1.588667	1.588667	257806500
2010-07-01	1.666667	1.728000	1.351333	1.464000	1.464000	123282000
2010-07-02	1.533333	1.540000	1.247333	1.280000	1.280000	77097000
2010-07-06	1.333333	1.333333	1.055333	1.074000	1.074000	103003500

In [14]:
```
df_tesla.resample(rule='A').min()
```

Out[14]:

Date	Open	High	Low	Close	Adj Close	Volume
2010-12-31	1.076000	1.108667	0.998667	1.053333	1.053333	1777500
2011-12-31	1.452000	1.484667	1.407333	1.455333	1.455333	3594000
2012-12-31	1.774667	1.790000	1.509333	1.519333	1.519333	5473500
2013-12-31	2.205333	2.225333	2.140667	2.194000	2.194000	6603000
2014-12-31	9.366667	9.800000	9.111333	9.289333	9.289333	19983000
2015-12-31	12.388667	12.619333	12.093333	12.333333	12.333333	10620000
2016-12-31	9.488000	10.331333	9.403333	9.578000	9.578000	24892500
2017-12-31	14.316667	14.688667	14.064000	14.466000	14.466000	32800500
2018-12-31	16.851999	17.355333	16.306000	16.704000	16.704000	46210500
2019-12-31	12.073333	12.445333	11.799333	11.931333	11.931333	36984000
2020-12-31	24.980000	26.990667	23.367332	24.081333	24.081333	52073100
2021-12-31	184.183334	188.736664	179.830002	187.666672	187.666672	29401800
2022-12-31	110.349998	116.269997	108.239998	109.099998	109.099998	41864700
2023-12-31	103.000000	111.750000	101.809998	108.099998	108.099998	83166000

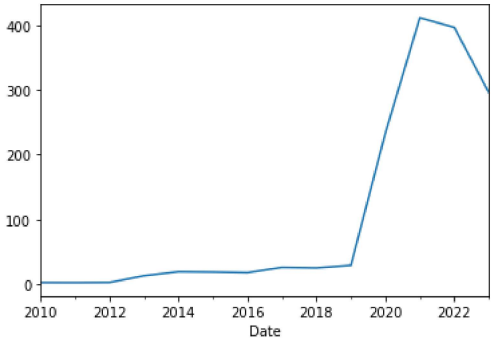
In [15]:
```
df_tesla.resample(rule='A').max()
```

Out[15]:

Date	Open	High	Low	Close	Adj Close	Volume
2010-12-31	2.391333	2.428000	2.316667	2.364667	2.364667	281494500
2011-12-31	2.308667	2.333333	2.268667	2.329333	2.329333	172767000
2012-12-31	2.546000	2.663333	2.474000	2.534000	2.534000	85206000
2013-12-31	12.930667	12.966667	12.558000	12.891333	12.891333	557458500
2014-12-31	19.177999	19.427999	18.693333	19.069332	19.069332	490225500
2015-12-31	18.680000	19.110001	18.420000	18.817333	18.817333	234744000
2016-12-31	17.763332	17.955999	16.967333	17.694668	17.694668	356136000
2017-12-31	25.779333	25.974001	25.290001	25.666668	25.666668	296871000
2018-12-31	25.000000	25.830667	24.474667	25.304667	25.304667	504745500
2019-12-31	29.000000	29.020666	28.423332	28.729334	28.729334	450091500
2020-12-31	233.330002	239.573334	230.373337	235.223328	235.223328	914082000
2021-12-31	411.470001	414.496674	405.666656	409.970001	409.970001	268189500
2022-12-31	396.516663	402.666656	378.679993	399.926666	399.926666	221923300
2023-12-31	296.040009	299.290009	289.519989	293.339996	293.339996	306590600

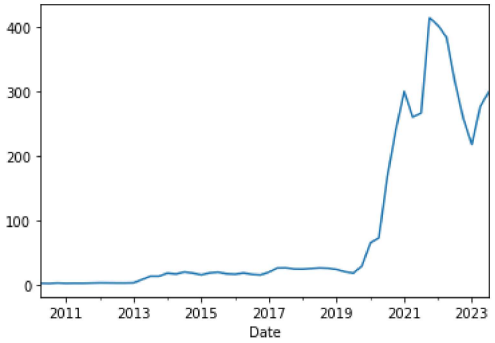
In [16]:
```
##year end frequency
df_tesla.resample(rule='A').max()['Open'].plot()
```

Out[16]:
```
<AxesSubplot:xlabel='Date'>
```



In [17]:
```
##quaterly start frequency
##https://towardsdatascience.com/resample-function-of-pandas-79b17ec82a78
df_tesla.resample(rule='QS').max()['High'].plot()
```

Out[17]:
```
<AxesSubplot:xlabel='Date'>
```



In [18]:
```
##Business End Frequency
##https://towardsdatascience.com/resample-function-of-pandas-79b17ec82a78
df_tesla.resample(rule='BA').max()
```

Out[18]:

Date	Open	High	Low	Close	Adj Close	Volume
2010-12-31	2.391333	2.428000	2.316667	2.364667	2.364667	281494500
2011-12-30	2.308667	2.333333	2.268667	2.329333	2.329333	172767000
2012-12-31	2.546000	2.663333	2.474000	2.534000	2.534000	85206000
2013-12-31	12.930667	12.966667	12.558000	12.891333	12.891333	557458500
2014-12-31	19.177999	19.427999	18.693333	19.069332	19.069332	490225500
2015-12-31	18.680000	19.110001	18.420000	18.817333	18.817333	234744000
2016-12-30	17.763332	17.955999	16.967333	17.694668	17.694668	356136000
2017-12-29	25.779333	25.974001	25.290001	25.666668	25.666668	296871000
2018-12-31	25.000000	25.830667	24.474667	25.304667	25.304667	504745500
2019-12-31	29.000000	29.020666	28.423332	28.729334	28.729334	450091500
2020-12-31	233.330002	239.573334	230.373337	235.223328	235.223328	914082000
2021-12-31	411.470001	414.496674	405.666656	409.970001	409.970001	268189500
2022-12-30	396.516663	402.666656	378.679993	399.926666	399.926666	221923300
2023-12-29	296.040009	299.290009	289.519989	293.339996	293.339996	306590600

In [19]:
```
df_tesla.resample(rule='BQS').max()
```

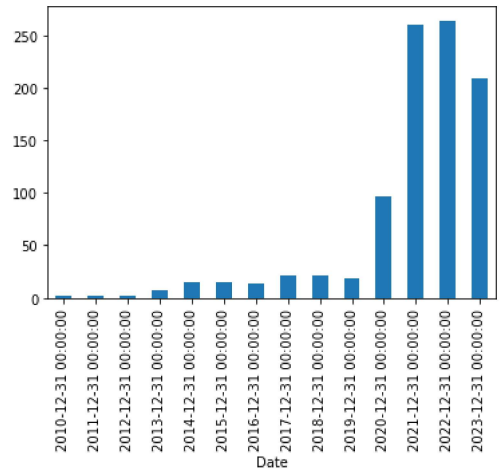
Out[19]:

Date	Open	High	Low	Close	Adj Close	Volume
2010-04-01	1.719333	2.028000	1.553333	1.592667	1.592667	281494500
2010-07-01	1.666667	1.728000	1.408667	1.465333	1.465333	123282000
2010-10-01	2.391333	2.428000	2.316667	2.364667	2.364667	139528500
2011-01-03	1.906000	1.914000	1.870000	1.896667	1.896667	172767000
2011-04-01	2.006667	2.100000	1.970000	2.009333	2.009333	93138000
2011-07-01	1.992667	2.029333	1.934000	1.982000	1.982000	45967500
2011-10-03	2.308667	2.333333	2.268667	2.329333	2.329333	58024500
2012-01-02	2.546000	2.663333	2.474000	2.529333	2.529333	82506000
2012-04-02	2.488667	2.564667	2.444667	2.534000	2.534000	83344500
2012-07-02	2.333333	2.400000	2.260000	2.397333	2.397333	85206000
2012-10-01	2.350667	2.386667	2.330000	2.352000	2.352000	34860000
2013-01-01	2.630000	2.666667	2.609333	2.632000	2.632000	135567000
2013-04-01	7.570000	7.660000	7.114000	7.355333	7.355333	557458500
2013-07-01	12.600000	12.966667	12.533333	12.891333	12.891333	485578500
2013-10-01	12.930667	12.948667	12.558000	12.866667	12.866667	466075500
2014-01-01	17.416668	17.666668	16.855333	16.989332	16.989332	490225500
2014-04-01	15.970000	16.299334	15.933333	16.004000	16.004000	300849000
2014-07-01	19.177999	19.427999	18.693333	19.069332	19.069332	246831000
2014-10-01	17.483334	17.702667	17.186666	17.374666	17.374666	230320500
2015-01-01	14.858000	15.032000	14.642667	14.732667	14.732667	234744000
2015-04-01	17.926001	18.094000	17.733334	17.919333	17.919333	186837000
2015-07-01	18.680000	19.110001	18.420000	18.817333	18.817333	219357000
2015-10-01	16.589333	16.656000	16.275333	16.504667	16.504667	223500000
2016-01-01	15.814000	15.992000	15.666667	15.888000	15.888000	213786000
2016-04-01	17.763332	17.955999	16.967333	17.694668	17.694668	356136000
2016-07-01	15.700000	15.775333	15.349333	15.652667	15.652667	119146500
2016-10-03	14.768667	14.920000	14.480000	14.649333	14.649333	196405500
2017-01-02	18.687332	19.159332	18.573999	18.732000	18.732000	223728000
2017-04-03	25.779333	25.799334	25.290001	25.563334	25.563334	258921000
2017-07-03	25.350000	25.974001	25.178667	25.666668	25.666668	289867500
2017-10-02	23.798668	24.200001	23.608667	23.976667	23.976667	296871000
2018-01-01	24.000000	24.033333	23.490667	23.827999	23.827999	315021000
2018-04-02	24.344000	24.915333	23.633333	24.722000	24.722000	335211000
2018-07-02	24.606001	25.830667	24.474667	25.304667	25.304667	504745500
2018-10-01	25.000000	25.299334	24.450001	25.119333	25.119333	411382500
2019-01-01	23.080667	23.466667	22.943333	23.153999	23.153999	362262000
2019-04-01	19.219999	19.744667	19.144667	19.454000	19.454000	398206500

Date	Open	High	Low	Close	Adj Close	Volume
2019-07-01	17.278000	17.738001	17.210667	17.658667	17.658667	336274500
2019-10-01	29.000000	29.020666	28.423332	28.729334	28.729334	450091500
2020-01-01	61.566666	64.599335	60.068001	61.161331	61.161331	914082000
2020-04-01	67.518669	72.512665	66.915337	71.987335	71.987335	487977000
2020-07-01	167.380005	167.496674	156.836670	166.106674	166.106674	584781000
2020-10-01	233.330002	239.573334	230.373337	235.223328	235.223328	666378600
2021-01-01	297.126678	300.133331	290.533325	294.363342	294.363342	268189500
2021-04-01	256.899994	260.263336	244.203339	254.106674	254.106674	147052200
2021-07-01	262.399994	266.333344	258.333344	263.786682	263.786682	100847400
2021-10-01	411.470001	414.496674	405.666656	409.970001	409.970001	188556300
2022-01-03	396.516663	402.666656	378.679993	399.926666	399.926666	151565700
2022-04-01	378.766663	384.290009	362.433319	381.816681	381.816681	144973200
2022-07-01	311.666656	314.666656	305.579987	309.320007	309.320007	142032300
2022-10-03	254.500000	257.500000	242.009995	249.440002	249.440002	221923300
2023-01-02	211.759995	217.649994	206.110001	214.240005	214.240005	306590600
2023-04-03	275.130005	276.989990	261.119995	274.450012	274.450012	211797100
2023-07-03	296.040009	299.290009	289.519989	293.339996	293.339996	175158300

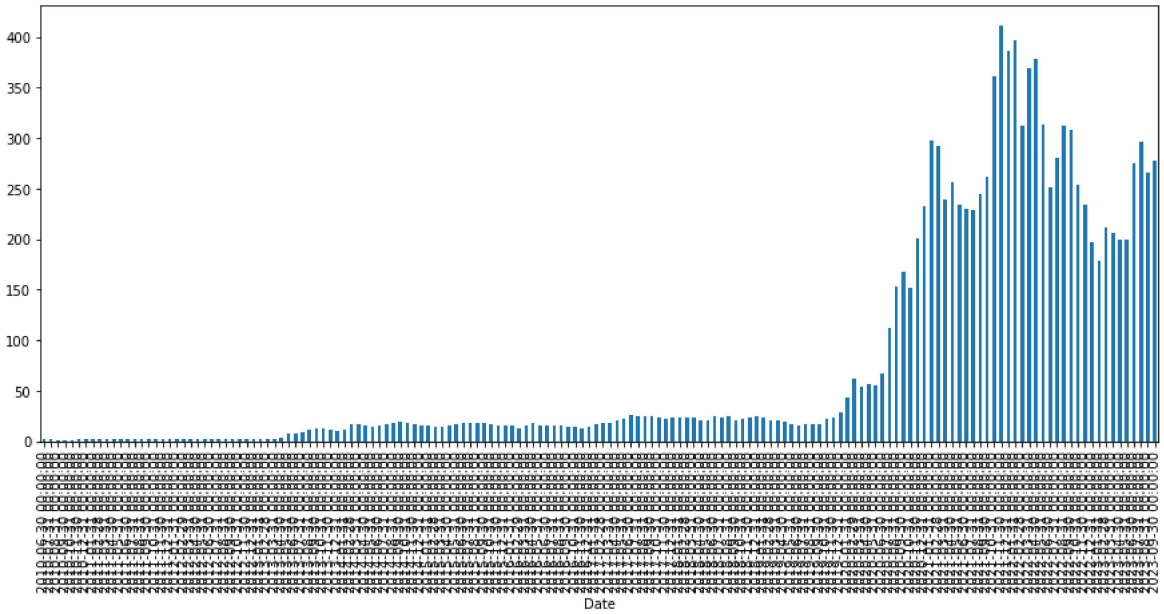
In [20]:
```
##plotting
df_tesla['Open'].resample(rule='A').mean().plot(kind='bar')
```

Out[20]:
```
<AxesSubplot:xlabel='Date'>
```



In [21]:
```
df_tesla['Open'].resample(rule='M').max().plot(kind='bar',figsize=(15,6))
```

Out[21]:
```
<AxesSubplot:xlabel='Date'>
```



In [22]:
```
df_tesla['High'].rolling(11).max().head(20)
```

Out[22]:
```
Date
2010-06-29         NaN
2010-06-30         NaN
2010-07-01         NaN
2010-07-02         NaN
2010-07-06         NaN
2010-07-07         NaN
2010-07-08         NaN
2010-07-09         NaN
2010-07-12         NaN
2010-07-13         NaN
2010-07-14    2.028000
2010-07-15    2.028000
2010-07-16    1.728000
2010-07-19    1.540000
2010-07-20    1.483333
2010-07-21    1.483333
2010-07-22    1.483333
2010-07-23    1.483333
2010-07-26    1.483333
2010-07-27    1.483333
Name: High, dtype: float64
```

In [23]:
```
df_tesla['Open:30 days rolling']=df_tesla['Open'].rolling(30).mean()
```

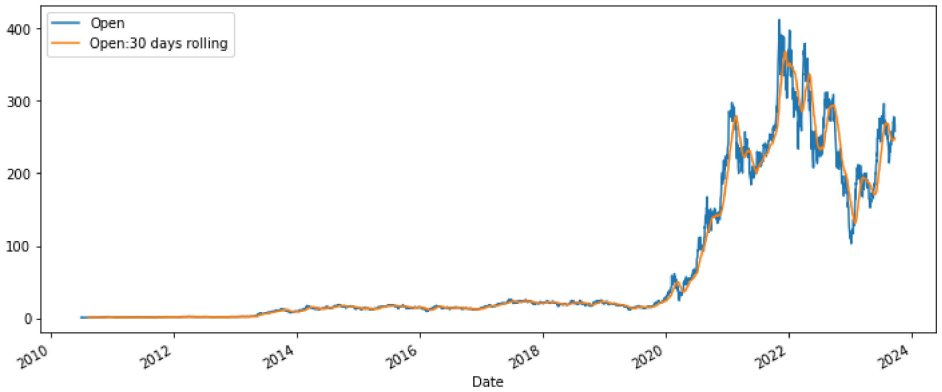
In [24]:
```
df_tesla.head(31)
```

Out[24]:

Date	Open	High	Low	Close	Adj Close	Volume	Open:30 days rolling
2010-06-29	1.266667	1.666667	1.169333	1.592667	1.592667	281494500	NaN
2010-06-30	1.719333	2.028000	1.553333	1.588667	1.588667	257806500	NaN
2010-07-01	1.666667	1.728000	1.351333	1.464000	1.464000	123282000	NaN
2010-07-02	1.533333	1.540000	1.247333	1.280000	1.280000	77097000	NaN
2010-07-06	1.333333	1.333333	1.055333	1.074000	1.074000	103003500	NaN
2010-07-07	1.093333	1.108667	0.998667	1.053333	1.053333	103825500	NaN
2010-07-08	1.076000	1.168000	1.038000	1.164000	1.164000	115671000	NaN
2010-07-09	1.172000	1.193333	1.103333	1.160000	1.160000	60759000	NaN
2010-07-12	1.196667	1.204667	1.133333	1.136667	1.136667	33037500	NaN
2010-07-13	1.159333	1.242667	1.126667	1.209333	1.209333	40201500	NaN
2010-07-14	1.196000	1.343333	1.184000	1.322667	1.322667	62928000	NaN
2010-07-15	1.329333	1.433333	1.266667	1.326000	1.326000	56097000	NaN
2010-07-16	1.380000	1.420000	1.336667	1.376000	1.376000	39319500	NaN
2010-07-19	1.424667	1.483333	1.394667	1.460667	1.460667	37297500	NaN
2010-07-20	1.456667	1.456667	1.336667	1.353333	1.353333	27379500	NaN
2010-07-21	1.377333	1.393333	1.300000	1.348000	1.348000	18787500	NaN
2010-07-22	1.366667	1.416667	1.358000	1.400000	1.400000	14367000	NaN
2010-07-23	1.412667	1.437333	1.404000	1.419333	1.419333	9804000	NaN
2010-07-26	1.433333	1.433333	1.353333	1.396667	1.396667	13833000	NaN
2010-07-27	1.394000	1.412000	1.350667	1.370000	1.370000	9295500	NaN
2010-07-28	1.370000	1.393333	1.367333	1.381333	1.381333	7008000	NaN
2010-07-29	1.384667	1.392000	1.333333	1.356667	1.356667	9240000	NaN
2010-07-30	1.346667	1.362667	1.303333	1.329333	1.329333	6403500	NaN
2010-08-02	1.366667	1.398000	1.355333	1.394667	1.394667	10771500	NaN
2010-08-03	1.400000	1.463333	1.388000	1.463333	1.463333	18457500	NaN
2010-08-04	1.463333	1.478667	1.390000	1.417333	1.417333	13695000	NaN
2010-08-05	1.436000	1.436667	1.336667	1.363333	1.363333	11943000	NaN
2010-08-06	1.340000	1.344000	1.301333	1.306000	1.306000	11128500	NaN
2010-08-09	1.326667	1.332000	1.296667	1.306667	1.306667	12190500	NaN
2010-08-10	1.310000	1.310000	1.254667	1.268667	1.268667	19219500	1.357711
2010-08-11	1.246000	1.258667	1.190000	1.193333	1.193333	11964000	1.357022

In [25]:
```
df_tesla[['Open','Open:30 days rolling']].plot(figsize=(12,5))
```

Out[25]:
```
<AxesSubplot:xlabel='Date'>
```



In [26]:
```
#xlimt and ylimt

open_mean = df_tesla['Open'].rolling(30).mean()
open_std = df_tesla['Open'].rolling(30).std()

fig = go.Figure()

fig.add_trace(go.Scatter(x=df_tesla.index, y=open_mean, mode='lines', name='moving avg', line=dict(color='blue')))
fig.add_trace(go.Scatter(x=df_tesla.index, y=open_std, mode='lines', name='moving std', line=dict(color='green')))
fig.add_trace(go.Scatter(x=df_tesla.index, y=df_tesla['Open'], mode='lines', name='Close', line=dict(color='red')))

fig.update_layout(title='Tesla Open and Close Prices',
                  xaxis = dict(range=['2022-01-01', '2023-08-25']))

fig.show()
```

In [27]:
```
#xlimt and ylimt

open_mean = df_tesla['Open'].rolling(5).mean()
open_std = df_tesla['Open'].rolling(5).std()

fig = go.Figure()

fig.add_trace(go.Scatter(x=df_tesla.index, y=open_mean, mode='lines', name='moving avg', line=dict(color='blue')))
fig.add_trace(go.Scatter(x=df_tesla.index, y=open_std, mode='lines', name='moving std', line=dict(color='green')))
fig.add_trace(go.Scatter(x=df_tesla.index, y=df_tesla['Open'], mode='lines', name='Close', line=dict(color='red')))

fig.update_layout(title='Tesla Open and Close Prices',
                  xaxis = dict(range=['2022-01-01', '2023-08-25']))
```

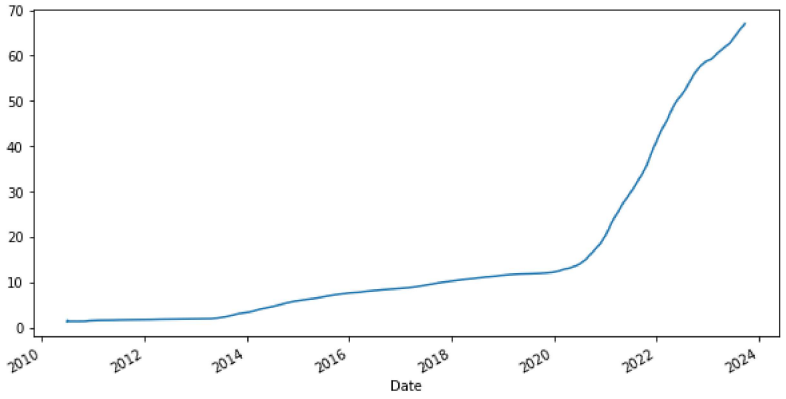
In [28]:
```
#xlimt and ylimt

open_mean = df_tesla['Open'].rolling(60).mean()
open_std = df_tesla['Open'].rolling(60).std()

fig = go.Figure()

fig.add_trace(go.Scatter(x=df_tesla.index, y=open_mean, mode='lines', name='moving avg', line=dict(color='blue')))
fig.add_trace(go.Scatter(x=df_tesla.index, y=open_std, mode='lines', name='moving std', line=dict(color='green')))
fig.add_trace(go.Scatter(x=df_tesla.index, y=df_tesla['Open'], mode='lines', name='Close', line=dict(color='red')))

fig.update_layout(title='Tesla Open and Close Prices',
                  xaxis = dict(range=['2022-01-01', '2023-08-25']))
```

In [29]:

```
#xlimt and ylimt

open_mean = df_tesla['Open'].rolling(15).mean()
open_std = df_tesla['Open'].rolling(15).std()

fig = go.Figure()

fig.add_trace(go.Scatter(x=df_tesla.index, y=open_mean, mode='lines', name='moving avg', line=dict(color='blue')))
fig.add_trace(go.Scatter(x=df_tesla.index, y=open_std, mode='lines', name='moving std', line=dict(color='green')))
fig.add_trace(go.Scatter(x=df_tesla.index, y=df_tesla['Open'], mode='lines', name='Close', line=dict(color='red')))

fig.update_layout(title='Tesla Open and Close Prices',
                  xaxis = dict(range=['2022-01-01', '2023-08-25']))
```

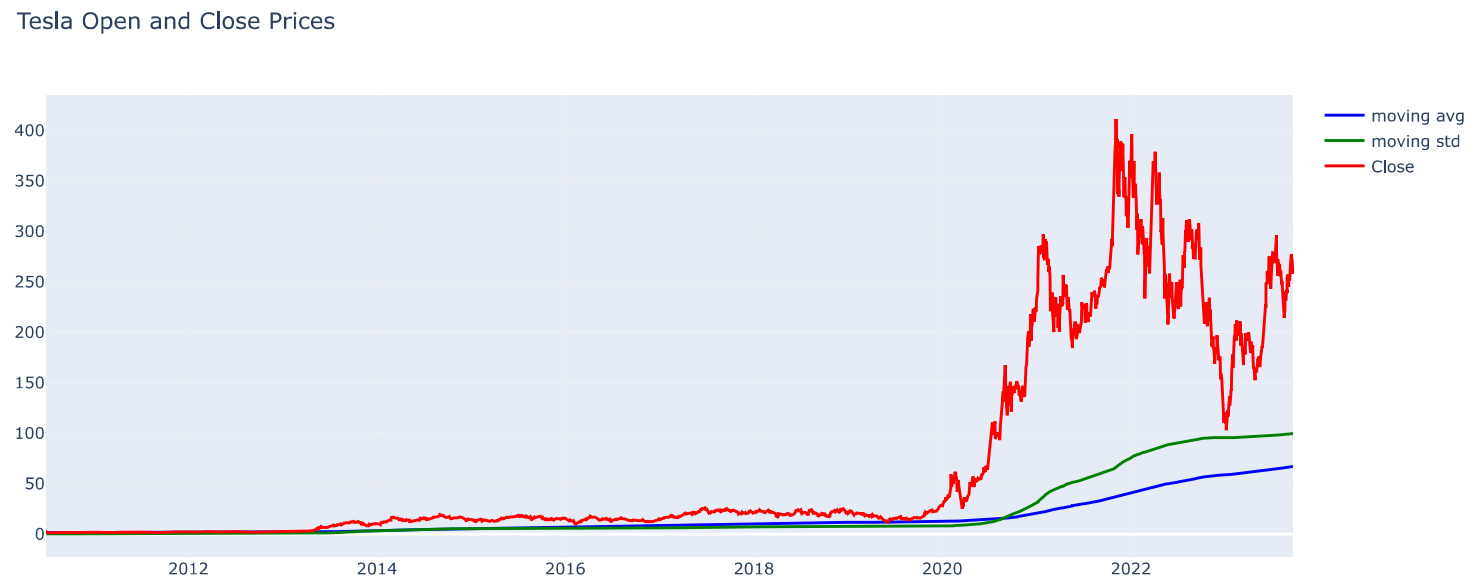
In [30]:
```
df_tesla['Open'].expanding().mean().plot(figsize=(10,5))
```

Out[30]:
```
<AxesSubplot:xlabel='Date'>
```



In [31]:
```
#xlimt and ylimt

open_mean = df_tesla['Open'].expanding().mean()
open_std = df_tesla['Open'].expanding().std()

fig = go.Figure()

fig.add_trace(go.Scatter(x=df_tesla.index, y=open_mean, mode='lines', name='moving avg', line=dict(color='blue')))
fig.add_trace(go.Scatter(x=df_tesla.index, y=open_std, mode='lines', name='moving std', line=dict(color='green')))
fig.add_trace(go.Scatter(x=df_tesla.index, y=df_tesla['Open'], mode='lines', name='Close', line=dict(color='red')))
```

```
fig.update_layout(title='Tesla Open and Close Prices')
```



In [ ]: