USD-GHS CURRENCY FORECASTING PROJECT

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from IPython.core.display import HTML

data = pd.read_csv("GHS=X.csv")

print(data.head())
```

```
         Date     Open     High      Low    Close  Adj Close  Volume
0  2008-01-01  0.97572  0.97702  0.97368  0.97415    0.97415     0.0
1  2008-01-02  0.97432  0.97565  0.94682  0.94853    0.94853     0.0
2  2008-01-03  0.94853  0.96018  0.94474  0.95754    0.95754     0.0
3  2008-01-04  0.95744  0.96050  0.94406  0.94778    0.94778     0.0
4  2008-01-07  0.94800  0.95346  0.94800  0.95117    0.95117     0.0
```

In [2]:
```
print(data.isnull().sum())
```

```
Date          0
Open         18
High         18
Low          18
Close        18
Adj Close    18
Volume       18
dtype: int64
```

In [3]:
```
data = data.dropna()
```

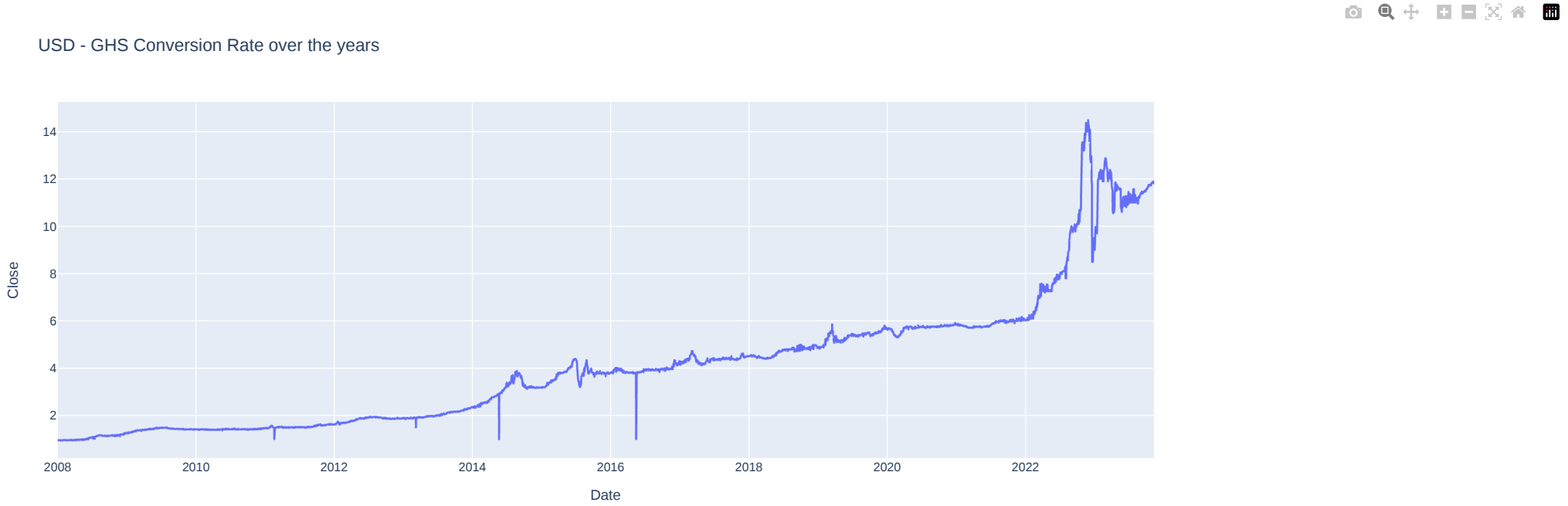
In [4]:
```
print(data.describe())
```

```
              Open         High          Low        Close    Adj Close  Volume
count  4120.000000  4120.000000  4120.000000  4120.000000  4120.000000  4120.0
mean      4.101485     4.231181     4.068988     4.084199     4.084199     0.0
std       2.834193     6.538584     2.793049     2.794460     2.794460     0.0
min       0.947130     0.950510     0.600000     0.946510     0.946510     0.0
25%       1.628125     1.636575     1.624500     1.629675     1.629675     0.0
50%       3.838800     3.858050     3.820450     3.837250     3.837250     0.0
75%       5.484880     5.510000     5.463922     5.490000     5.490000     0.0
max      14.583687   380.000000    14.482678    14.480690    14.480690     0.0
```

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4120 entries, 0 to 4137
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       4120 non-null   object 
 1   Open       4120 non-null   float64
 2   High       4120 non-null   float64
 3   Low        4120 non-null   float64
 4   Close      4120 non-null   float64
 5   Adj Close  4120 non-null   float64
 6   Volume     4120 non-null   float64
dtypes: float64(6), object(1)
memory usage: 257.5+ KB
```

In [6]:
```
figure = px.line(data, x="Date",
                 y="Close",
                 title='USD - GHS Conversion Rate over the years')
figure.show()
```



In [7]:

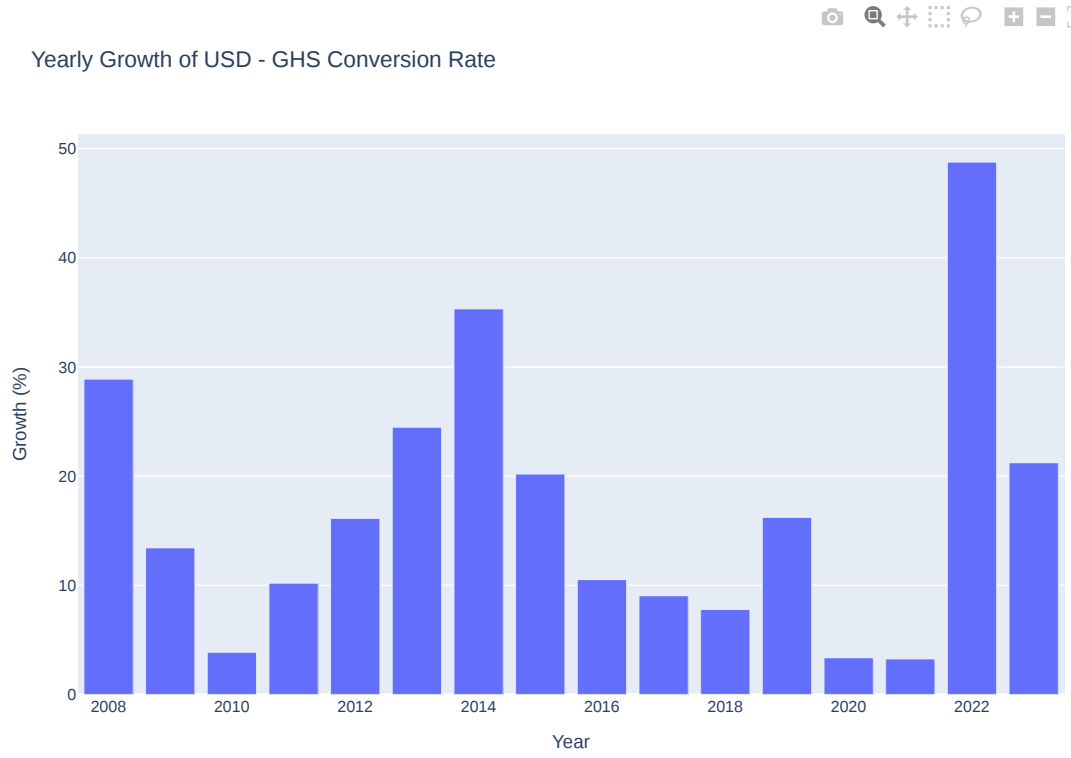
```
data["Date"] = pd.to_datetime(data["Date"], format = '%Y-%m-%d')
data['Year'] = data['Date'].dt.year
data["Month"] = data["Date"].dt.month
print(data.head())
```

```
        Date     Open     High      Low    Close  Adj Close  Volume  Year  \
0 2008-01-01  0.97572  0.97702  0.97368  0.97415    0.97415     0.0  2008
1 2008-01-02  0.97432  0.97565  0.94682  0.94853    0.94853     0.0  2008
2 2008-01-03  0.94853  0.96018  0.94474  0.95754    0.95754     0.0  2008
3 2008-01-04  0.95744  0.96050  0.94406  0.94778    0.94778     0.0  2008
4 2008-01-07  0.94800  0.95346  0.94800  0.95117    0.95117     0.0  2008

   Month
0      1
1      1
2      1
3      1
4      1
```

In [8]:

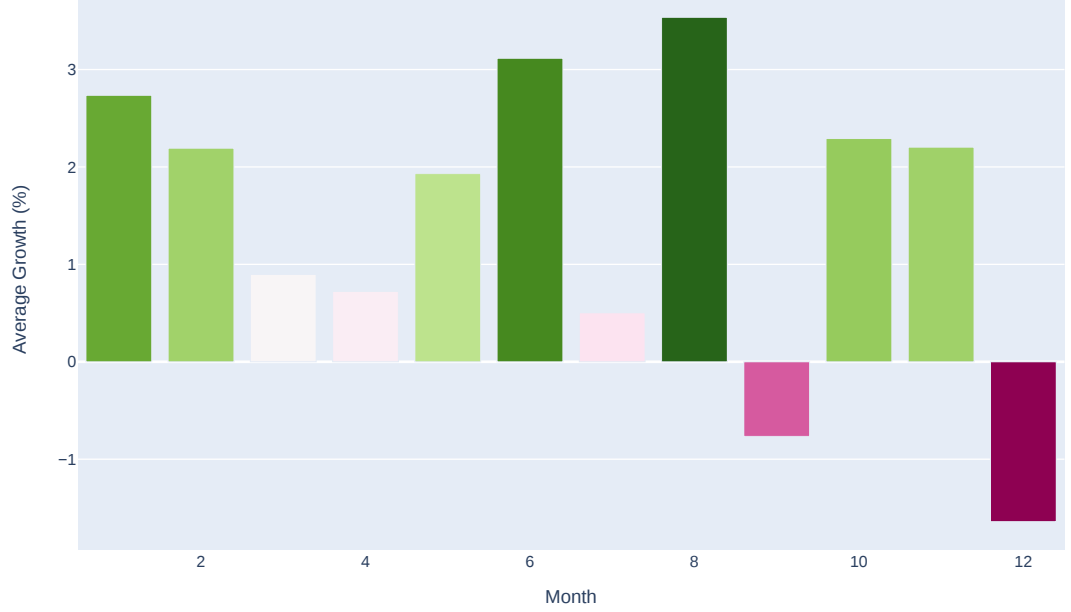
```
import plotly.graph_objs as go
import plotly.io as pio

# Calculate yearly growth
growth = data.groupby('Year').agg({'Close': lambda x: (x.iloc[-1]-x.iloc[0])/x.iloc[0]*100})

fig = go.Figure()
fig.add_trace(go.Bar(x=growth.index,
                     y=growth['Close'],
                     name='Yearly Growth'))

fig.update_layout(title="Yearly Growth of USD - GHS Conversion Rate",
                  xaxis_title="Year",
                  yaxis_title="Growth (%)",
                  width=900,
                  height=600)
```

```
pio.show(fig)
```



In [9]:

```
# Calculate monthly growth
data['Growth'] = data.groupby(['Year', 'Month'])['Close'].transform(lambda x: (x.iloc[-1] - x.iloc[0]) / x.iloc[0] * 100)

# Group data by Month and calculate average growth
grouped_data = data.groupby('Month').mean().reset_index()

fig = go.Figure()

fig.add_trace(go.Bar(
    x=grouped_data['Month'],
    y=grouped_data['Growth'],
    marker_color=grouped_data['Growth'],
    hovertemplate='Month: %{x}<br>Average Growth: %{y:.2f}%<extra></extra>'
))

fig.update_layout(
    title="Aggregated Monthly Growth of USD - GHS Conversion Rate",
    xaxis_title="Month",
    yaxis_title="Average Growth (%)",
    width=900,
    height=600
)

pio.show(fig)
```

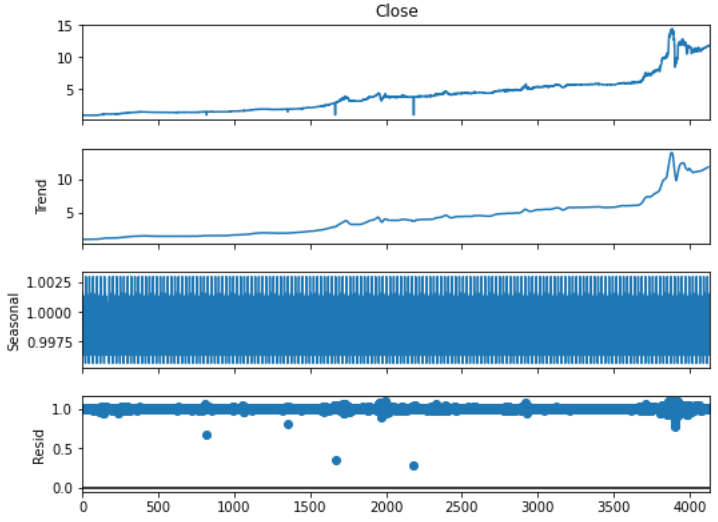


In [10]:

```
from statsmodels.tsa.seasonal import seasonal_decompose
result = seasonal_decompose(data["Close"], model='multiplicative', period=24)
fig = plt.figure()
fig = result.plot()
fig.set_size_inches(8, 6)
fig.show()
```

```
C:\Users\KINGSL~1\AppData\Local\Temp/ipykernel_15616/3687509498.py:6: UserWarning:

Matplotlib is currently using module://matplotlib_inline.backend_inline, which is a non-GUI backend, so cannot show the figure.
```

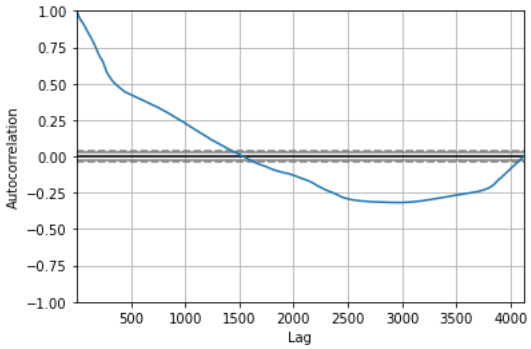
```
<Figure size 432x288 with 0 Axes>
```



In [11]:

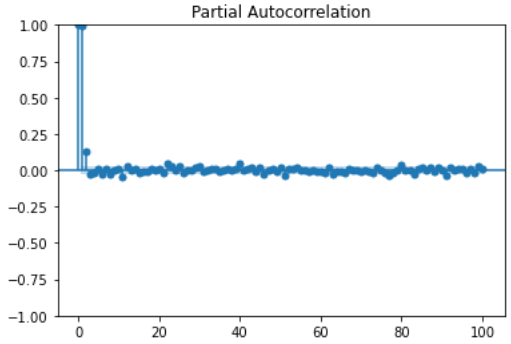
```
pd.plotting.autocorrelation_plot(data["Close"])
```

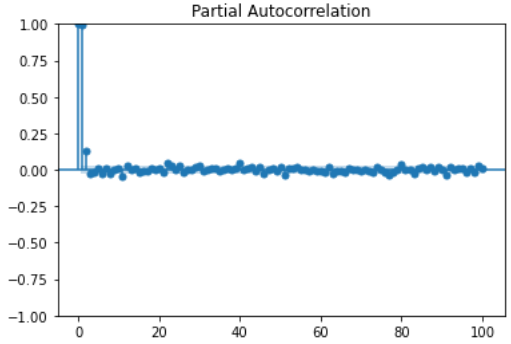
Out[11]: <AxesSubplot:xlabel='Lag', ylabel='Autocorrelation'>



In [12]:
```
from statsmodels.graphics.tsaplots import plot_pacf
plot_pacf(data["Close"], lags = 100)
```

Out[12]:





In [13]:
```
p, d, q = 5, 1, 2
from statsmodels.tsa.arima.model import ARIMA
model = ARIMA(data["Close"], order=(p, d, q))
fitted = model.fit()
print(fitted.summary())
```

```
C:\Users\Kingsley Mills\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning:

An unsupported index was provided and will be ignored when e.g. forecasting.

C:\Users\Kingsley Mills\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning:

An unsupported index was provided and will be ignored when e.g. forecasting.

C:\Users\Kingsley Mills\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning:
```

```
An unsupported index was provided and will be ignored when e.g. forecasting.
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                  Close   No. Observations:                 4120
Model:                 ARIMA(5, 1, 2)   Log Likelihood                2991.453
Date:                Sat, 11 Nov 2023   AIC                          -5966.905
Time:                        15:05:07   BIC                          -5916.318
Sample:                             0   HQIC                         -5949.000
                                 - 4120
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.0437      0.152      0.287      0.774      -0.255       0.342
ar.L2          0.4342      0.090      4.822      0.000       0.258       0.611
ar.L3          0.1143      0.039      2.896      0.004       0.037       0.192
ar.L4         -0.0208      0.010     -2.120      0.034      -0.040      -0.002
ar.L5          0.0426      0.009      4.941      0.000       0.026       0.060
ma.L1         -0.3131      0.152     -2.057      0.040      -0.611      -0.015
ma.L2         -0.3195      0.123     -2.598      0.009      -0.560      -0.078
sigma2         0.0137   3.29e-05    416.141      0.000       0.014       0.014
===================================================================================
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):          13563507.91
Prob(Q):                              0.97   Prob(JB):                         0.00
Heteroskedasticity (H):              39.99   Skew:                            -6.59
Prob(H) (two-sided):                  0.00   Kurtosis:                       283.81
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

In [14]:

```
predictions = fitted.predict(len(data), len(data)+60)
print(predictions)
```

```
4120    11.884286
4121    11.887765
4122    11.883702
4123    11.889185
4124    11.887542
          ...
4176    11.892266
4177    11.892266
4178    11.892266
4179    11.892266
4180    11.892266
Name: predicted_mean, Length: 61, dtype: float64
```

```
C:\Users\Kingsley Mills\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:836: ValueWarning:

No supported index is available. Prediction results will be given with an integer index beginning at `start`.

C:\Users\Kingsley Mills\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:836: FutureWarning:

No supported index is available. In the next version, calling this method in a model without a supported index will result in an exception.
```

In [15]:

```
# Create figure
fig = go.Figure()

# Add training data line plot
fig.add_trace(go.Scatter(
    x=data.index,
    y=data['Close'],
    mode='lines',
    name='Training Data',
    line=dict(color='blue')
))

# Add predictions line plot
fig.add_trace(go.Scatter(
```

```
    x=predictions.index,
    y=predictions,
    mode='lines',
    name='Predictions',
    line=dict(color='green')
))

fig.update_layout(
    title="GHS Rate - Training Data and Predictions",
    xaxis_title="Date",
    yaxis_title="Close",
    legend_title="Data",
    width=900,
    height=600
)

pio.show(fig)
```