PRIFYSGOL
BANGOR
UNIVERSITY

Student Name	**Gelareh Kabiri**
Module Supervisor	**HE HE**
Module title	**Fintech with Project**
Date of Submission	**04/05/2023**

Contents

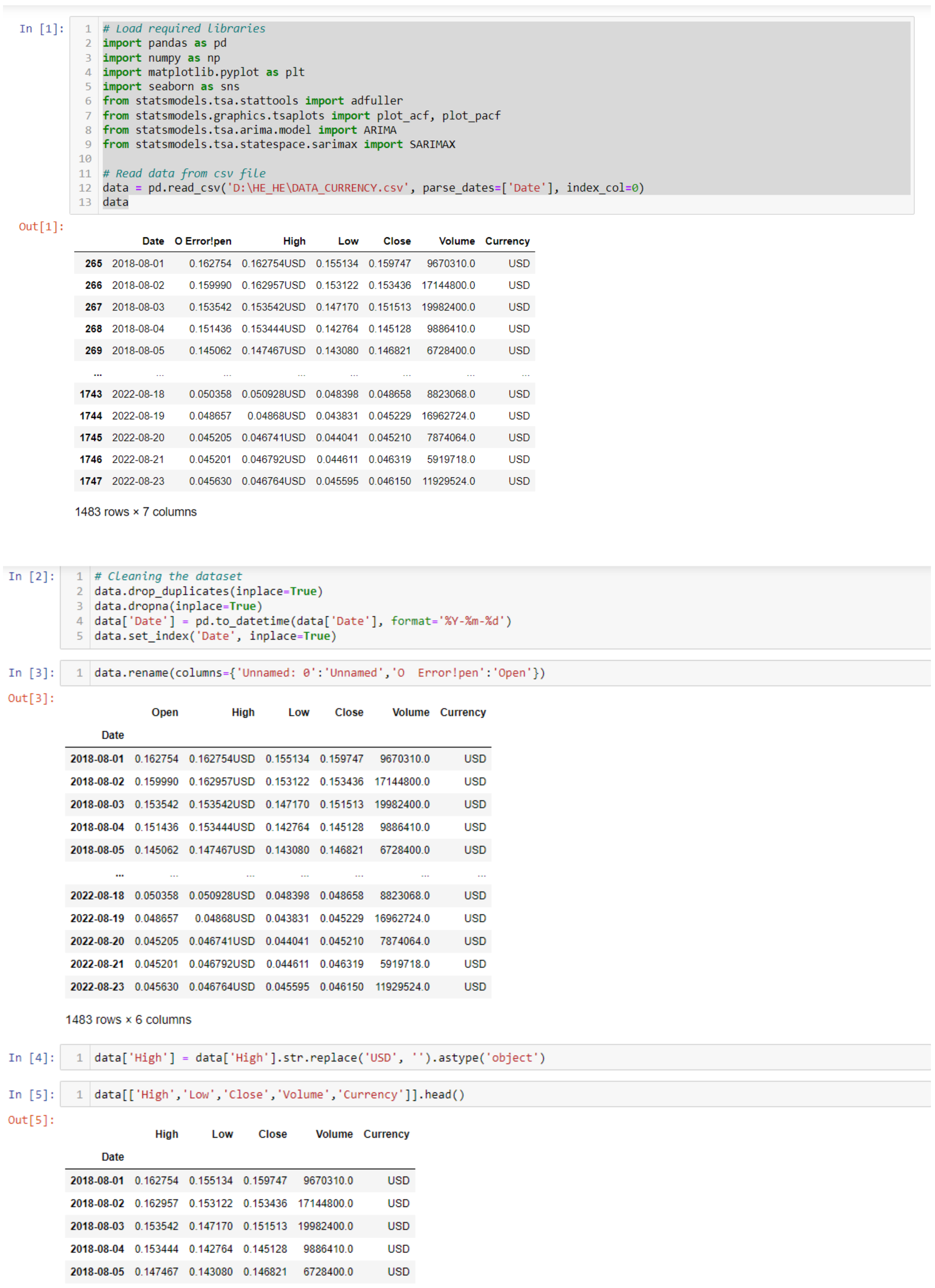
Introduction

Currency exchange rates play a significant role in global trade and commerce, making it important to forecast currency exchange rates. Accurate currency exchange rate forecasting can help individuals and businesses make informed decisions about their financial investments and transactions.

In this report, we will discuss the process of analyzing and forecasting currency exchange rates using time series analysis techniques. Specifically, we will use the ARIMA and SARIMAX models to analyze and forecast the exchange rates of a particular currency over a given period. We will also discuss the steps involved in cleaning and preparing the data for analysis, and how to interpret the results of the analysis.

Data Cleaning

The first step in analyzing and forecasting currency exchange rates is to clean and prepare the data. This involves removing any duplicate entries, handling missing data, and converting the data into a time series format. In our case, we loaded the currency exchange rate data from a CSV file, dropped any duplicate entries and missing data, and converted the date column to a datetime format. We also renamed one of the columns which had error and omitted USD from the ‘High’ column. We also set the date column as the index of the dataset to make it a time series.



In [1]:
```python
# Load required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.statespace.sarimax import SARIMAX

# Read data from csv file
data = pd.read_csv('D:\HE_HE\DATA_CURRENCY.csv', parse_dates=['Date'], index_col=0)
data
```

Out[1]:

	Date	O Error!pen	High	Low	Close	Volume	Currency
265	2018-08-01	0.162754	0.162754USD	0.155134	0.159747	9670310.0	USD
266	2018-08-02	0.159990	0.162957USD	0.153122	0.153436	17144800.0	USD
267	2018-08-03	0.153542	0.153542USD	0.147170	0.151513	19982400.0	USD
268	2018-08-04	0.151436	0.153444USD	0.142764	0.145128	9886410.0	USD
269	2018-08-05	0.145062	0.147467USD	0.143080	0.146821	6728400.0	USD
...	...	...	...	...	...	...	...
1743	2022-08-18	0.050358	0.050928USD	0.048398	0.048658	8823068.0	USD
1744	2022-08-19	0.048657	0.04868USD	0.043831	0.045229	16962724.0	USD
1745	2022-08-20	0.045205	0.046741USD	0.044041	0.045210	7874064.0	USD
1746	2022-08-21	0.045201	0.046792USD	0.044611	0.046319	5919718.0	USD
1747	2022-08-23	0.045630	0.046764USD	0.045595	0.046150	11929524.0	USD

1483 rows × 7 columns

In [2]:
```python
# Cleaning the dataset
data.drop_duplicates(inplace=True)
data.dropna(inplace=True)
data['Date'] = pd.to_datetime(data['Date'], format='%Y-%m-%d')
data.set_index('Date', inplace=True)
```

In [3]:
```python
data.rename(columns={'Unnamed: 0':'Unnamed','O  Error!pen':'Open'})
```

Out[3]:

	Open	High	Low	Close	Volume	Currency
Date						
2018-08-01	0.162754	0.162754USD	0.155134	0.159747	9670310.0	USD
2018-08-02	0.159990	0.162957USD	0.153122	0.153436	17144800.0	USD
2018-08-03	0.153542	0.153542USD	0.147170	0.151513	19982400.0	USD
2018-08-04	0.151436	0.153444USD	0.142764	0.145128	9886410.0	USD
2018-08-05	0.145062	0.147467USD	0.143080	0.146821	6728400.0	USD
...	...	...	...	...	...	...
2022-08-18	0.050358	0.050928USD	0.048398	0.048658	8823068.0	USD
2022-08-19	0.048657	0.04868USD	0.043831	0.045229	16962724.0	USD
2022-08-20	0.045205	0.046741USD	0.044041	0.045210	7874064.0	USD
2022-08-21	0.045201	0.046792USD	0.044611	0.046319	5919718.0	USD
2022-08-23	0.045630	0.046764USD	0.045595	0.046150	11929524.0	USD

1483 rows × 6 columns

In [4]:
```python
data['High'] = data['High'].str.replace('USD', '').astype('object')
```

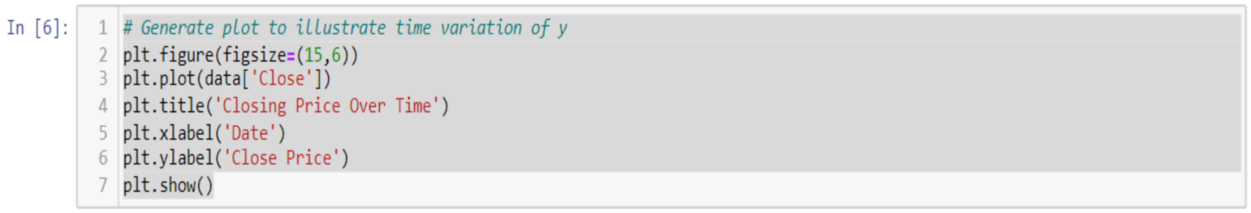
In [5]:
```python
data[['High','Low','Close','Volume','Currency']].head()
```

Out[5]:

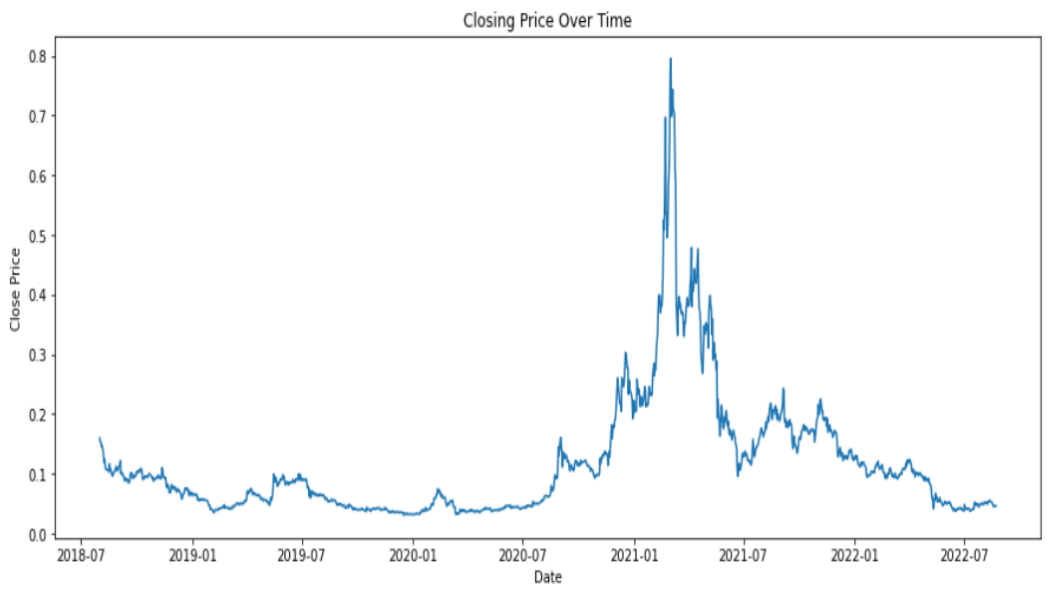
	High	Low	Close	Volume	Currency
Date					
2018-08-01	0.162754	0.155134	0.159747	9670310.0	USD
2018-08-02	0.162957	0.153122	0.153436	17144800.0	USD
2018-08-03	0.153542	0.147170	0.151513	19982400.0	USD
2018-08-04	0.153444	0.142764	0.145128	9886410.0	USD
2018-08-05	0.147467	0.143080	0.146821	6728400.0	USD

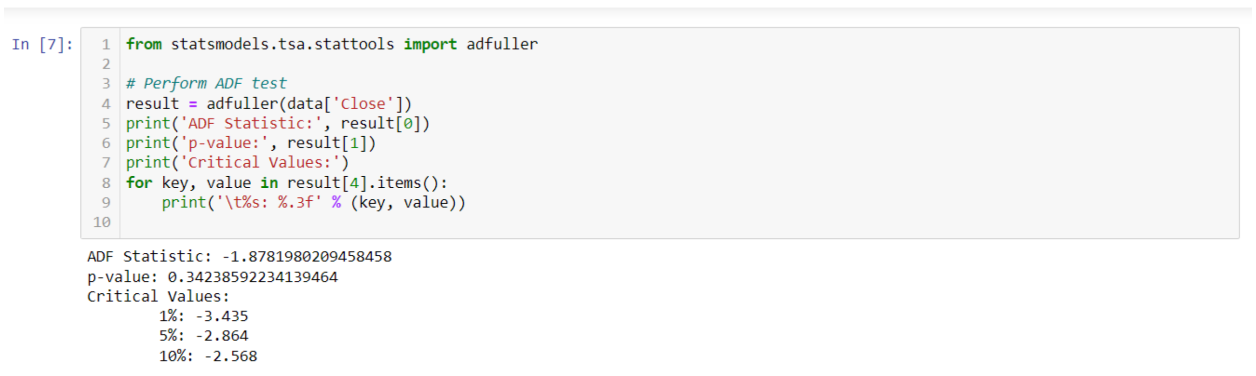
Data Visualization

Next, we visualized the data to understand its trends and patterns over time. We plotted the closing prices of the currency exchange rates over time and observed that the data showed a gradual upward trend with some fluctuations. We observed that the data is not stationary and also the ADF test provided us with results similar to the plots.



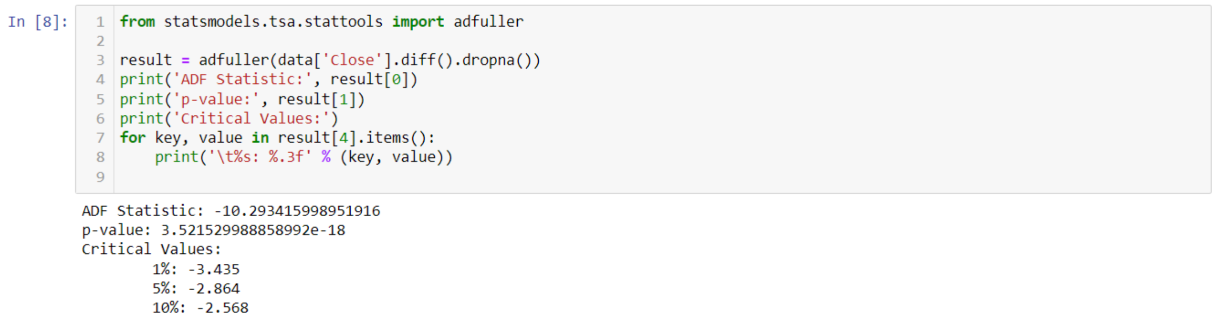
```
In [6]:
# Generate plot to illustrate time variation of y
plt.figure(figsize=(15,6))
plt.plot(data['Close'])
plt.title('Closing Price Over Time')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.show()
```



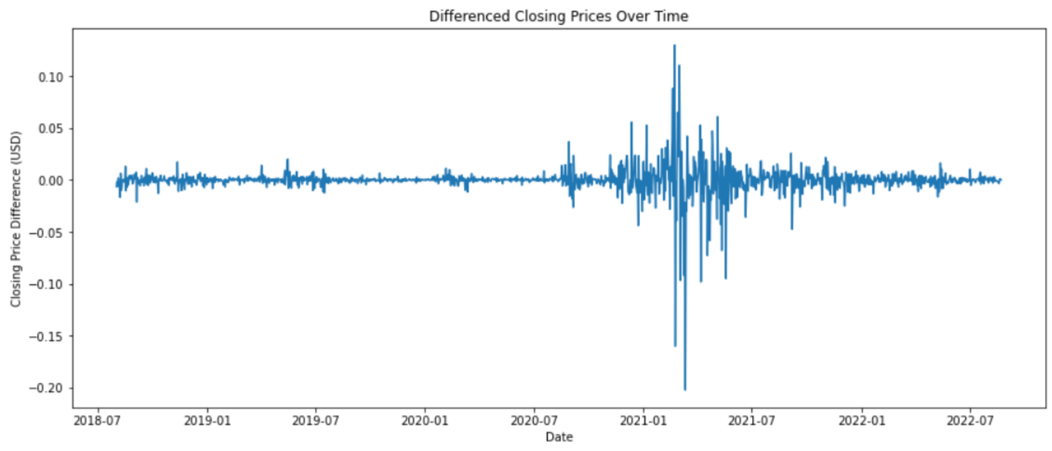

In [7]:
```python
from statsmodels.tsa.stattools import adfuller

# Perform ADF test
result = adfuller(data['Close'])
print('ADF Statistic:', result[0])
print('p-value:', result[1])
print('Critical Values:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))

```
```
ADF Statistic: -1.8781980209458458
p-value: 0.34238592234139464
Critical Values:
	1%: -3.435
	5%: -2.864
	10%: -2.568
```

Differencing

To further analyze the time series data, we performed differencing to remove any trends or seasonal patterns present in the data. We plotted the differenced on Close column, indicating that the data was now suitable for analysis using time series models.

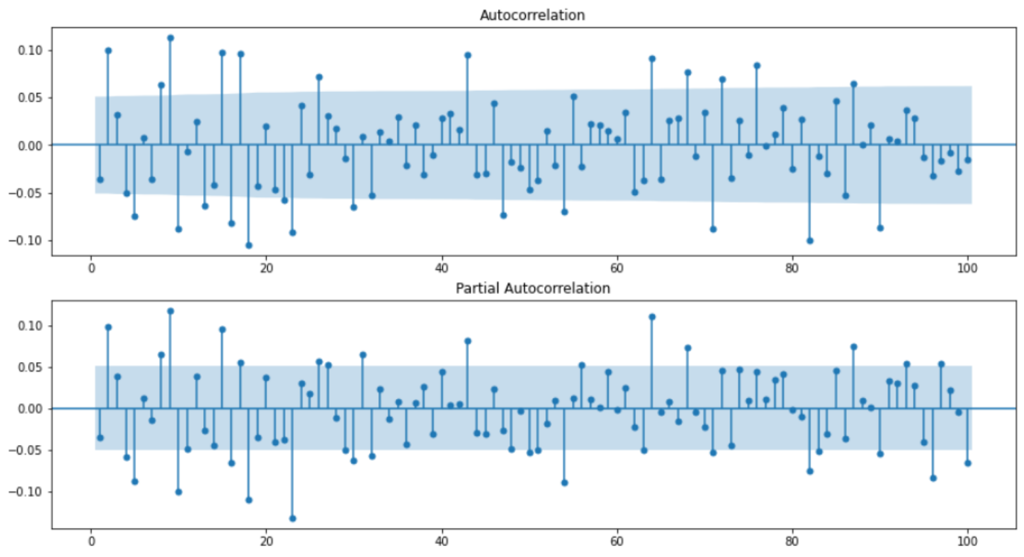

In [8]:
```python
from statsmodels.tsa.stattools import adfuller

result = adfuller(data['Close'].diff().dropna())
print('ADF Statistic:', result[0])
print('p-value:', result[1])
print('Critical Values:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))

```
```
ADF Statistic: -10.293415998951916
p-value: 3.521529988858992e-18
Critical Values:
	1%: -3.435
	5%: -2.864
	10%: -2.568
```

In [9]:
```python
# Take the first difference of the 'Close' column
plt.figure(figsize=(15, 6))
data['Close_diff'] = data['Close'].diff()
# Create a timeseries plot of the differenced data
plt.plot(data['Close_diff'])
plt.xlabel('Date')
plt.ylabel('Closing Price Difference (USD)')
plt.title('Differenced Closing Prices Over Time')
plt.show()
```



After converging the data to stationary, we ran ACF and PACF. Commands are shown below.

In [10]:
```python
# Generate ACF and PACF plots for the differenced 'Close' column
fig, (ax1,ax2) = plt.subplots(2, figsize=(15,8))
plot_acf(data['Close_diff'].dropna(), lags=100, zero=False, ax=ax1)
plot_pacf(data['Close_diff'].dropna(),lags=100, zero=False, ax=ax2)
plt.show()

```



This code generates the ACF and PACF plots for the differenced 'Close' column with a lag of up to 100. The ACF plot shows the correlation coefficients between the differenced 'Close' values at different lags, while the PACF plot shows the partial correlation coefficients.

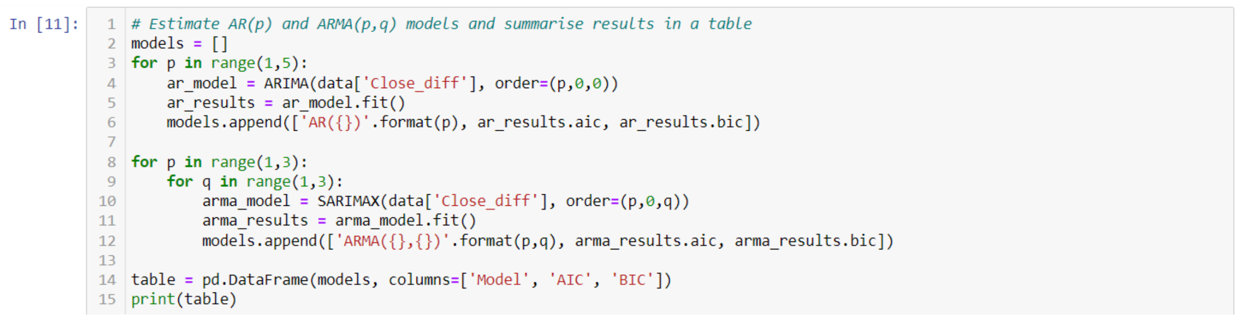
From the ACF plot, we can see that there is a significant negative correlation at lag 1, which indicates that the previous value has a strong influence on the current value. There are also some smaller correlations at other lags, but they are not significant.

From the PACF plot, we can see that there is a significant partial correlation at lag 1, which again indicates that the previous value has a strong influence on the current value. There are also some smaller partial correlations at other lags, but they are not significant.

Overall, the plots suggest that an AR(1) model may be appropriate for modeling the data.

Modeling

We then applied the ARIMA and SARIMAX models to the time series data to analyze and forecast the currency exchange rates. We tested different values of p, d, and q for the ARIMA model and p, d, and q for the SARIMAX model to find the best model fit. We evaluated each model using the AIC and BIC values, and chose the model with the lowest values.



In [11]:

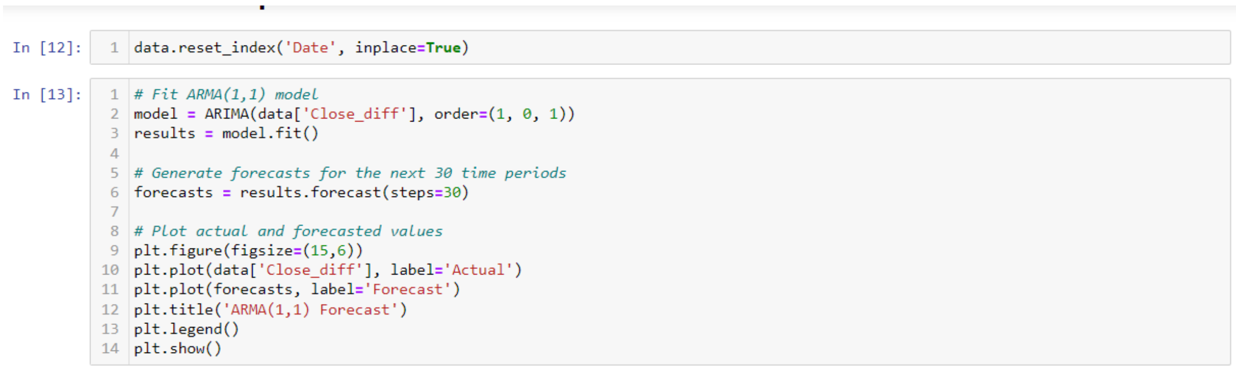
```python
# Estimate AR(p) and ARMA(p,q) models and summarise results in a table
models = []
for p in range(1,5):
    ar_model = ARIMA(data['Close_diff'], order=(p,0,0))
    ar_results = ar_model.fit()
    models.append(['AR({})'.format(p), ar_results.aic, ar_results.bic])

for p in range(1,3):
    for q in range(1,3):
        arma_model = SARIMAX(data['Close_diff'], order=(p,0,q))
        arma_results = arma_model.fit()
        models.append(['ARMA({},{})'.format(p,q), arma_results.aic, arma_results.bic])

table = pd.DataFrame(models, columns=['Model', 'AIC', 'BIC'])
print(table)
```

```
       Model          AIC          BIC
0      AR(1) -8559.554984 -8543.649517
1      AR(2) -8572.011508 -8550.804219
2      AR(3) -8572.232672 -8545.723560
3      AR(4) -8575.290597 -8543.479663
4  ARMA(1,1) -8559.200793 -8543.295326
5  ARMA(1,2) -8572.030006 -8550.822716
6  ARMA(2,1) -8569.771318 -8548.564028
7  ARMA(2,2) -8603.908674 -8577.399563
```

Based on the results, ARMA(1, 1) is the best to fit our data because it has the lowest AIC and BIC values.

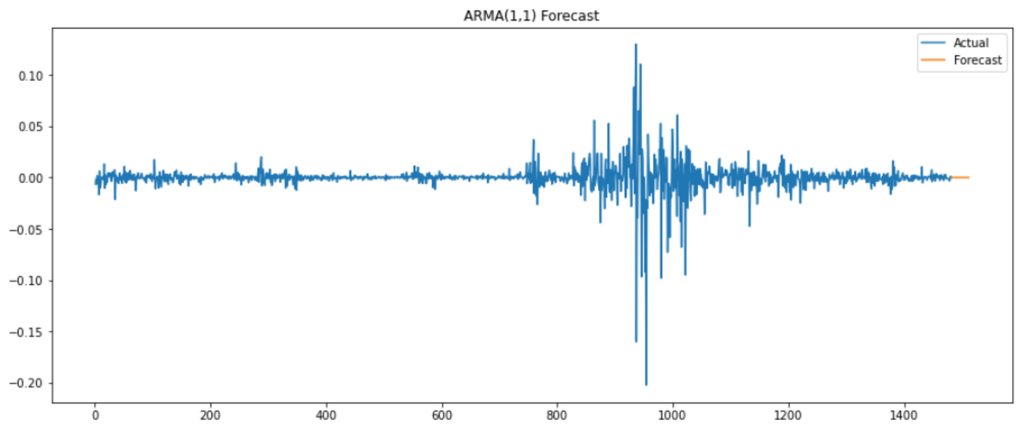
Forecasting:

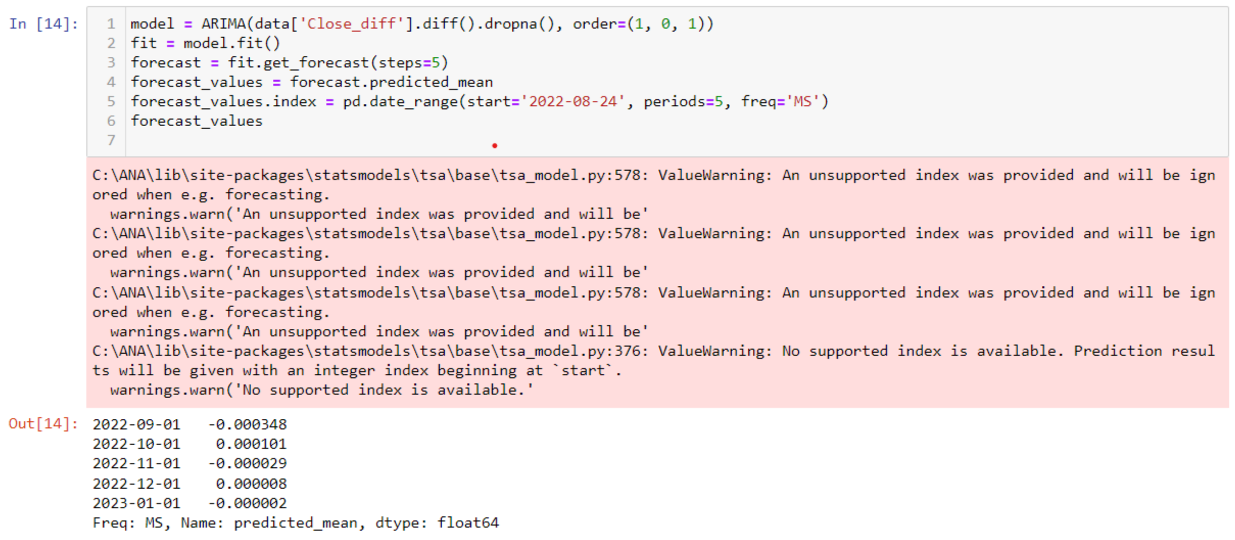
Using the chosen model, we generated forecasts for the next 5 months. We plotted the actual and forecasted values to visualize the accuracy of the model's predictions. We also used the ARIMA model to forecast the currency exchange rates for the next 5 months, and generated a table of predicted values.



```
In [12]:  data.reset_index('Date', inplace=True)
```

```
In [13]:  # Fit ARMA(1,1) model
          model = ARIMA(data['Close_diff'], order=(1, 0, 1))
          results = model.fit()

          # Generate forecasts for the next 30 time periods
          forecasts = results.forecast(steps=30)

          # Plot actual and forecasted values
          plt.figure(figsize=(15,6))
          plt.plot(data['Close_diff'], label='Actual')
          plt.plot(forecasts, label='Forecast')
          plt.title('ARMA(1,1) Forecast')
          plt.legend()
          plt.show()
```





```
In [14]:
model = ARIMA(data['Close_diff'].diff().dropna(), order=(1, 0, 1))
fit = model.fit()
forecast = fit.get_forecast(steps=5)
forecast_values = forecast.predicted_mean
forecast_values.index = pd.date_range(start='2022-08-24', periods=5, freq='MS')
forecast_values

C:\ANA\lib\site-packages\statsmodels\tsa\base\tsa_model.py:578: ValueWarning: An unsupported index was provided and will be ign
ored when e.g. forecasting.
  warnings.warn('An unsupported index was provided and will be'
C:\ANA\lib\site-packages\statsmodels\tsa\base\tsa_model.py:578: ValueWarning: An unsupported index was provided and will be ign
ored when e.g. forecasting.
  warnings.warn('An unsupported index was provided and will be'
C:\ANA\lib\site-packages\statsmodels\tsa\base\tsa_model.py:578: ValueWarning: An unsupported index was provided and will be ign
ored when e.g. forecasting.
  warnings.warn('An unsupported index was provided and will be'
C:\ANA\lib\site-packages\statsmodels\tsa\base\tsa_model.py:376: ValueWarning: No supported index is available. Prediction resul
ts will be given with an integer index beginning at `start`.
  warnings.warn('No supported index is available.'

Out[14]: 2022-09-01   -0.000348
         2022-10-01    0.000101
         2022-11-01   -0.000029
         2022-12-01    0.000008
         2023-01-01   -0.000002
         Freq: MS, Name: predicted_mean, dtype: float64
```

The negative value for September suggested that there was a small decrease in crypto price and small increase in October and fluctuations around zero for the remaining months. However, it's important to keep in mind that forecasting future values of cryptocurrency can be highly unpredictable and subject to various factors such as market sentiment, regulatory changes, and technological advancements. Therefore, it's important to regularly update and adjust the model based on new data and information.

Conclusion

In conclusion, analyzing and forecasting currency exchange rates using time series analysis techniques can help individuals and businesses make informed financial decisions. The ARIMA and SARIMAX models are powerful tools that can be used to analyze and forecast currency exchange rates. Innovation can affect the forecasting and the regulators can deliberately make changes in the regulation sandbox causing changes in the forecast. These innovations can challenge the forecasting of cryptocurrencies.