EX -8
17/04/2025

Create an ARIMA model for time series forecasting.

AIM :
Create an ARIMA model for time series forecasting.

Procedure and Code :

Step 1 - Import the Files and Libraries .

```
import pandas as pd
import numpy as np
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
import matplotlib.pyplot as plt
```

Step 2 - Describe and Read the Data

```
df = pd.read_csv('/content/drive/MyDrive/TimeSereisDatasets/Ex-8/Copy of
daily-website-visitors.csv')


df['Date'] = pd.to_datetime(df['Date'])
df.set_index('Date', inplace=True)


df['Unique.Visits'] = df['Unique.Visits'].str.replace(',', '').astype(int)
df['Page.Loads'] = df['Page.Loads'].str.replace(',', '').astype(int)


ts = df['Unique.Visits']
```

EX -8
17/04/2025

Step 3 -`Stationary Models`

```
from statsmodels.tsa.stattools import adfuller

result = adfuller(ts)
print('ADF Statistic:', result[0])
print('p-value:', result[1])
print('Critical Values:', result[4])
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(12,8))
plot_acf(ts.diff().dropna(), ax=ax1)
plot_pacf(ts.diff().dropna(), ax=ax2)
plt.show()
```

Step 4 - `ARIMA Model training and Evaluation`

```
    train_size = int(len(ts) * 0.8)
train, test = ts[:train_size], ts[train_size:]


model = ARIMA(train, order=(2,1,2))
model_fit = model.fit()
print(model_fit.summary())

forecast = model_fit.forecast(steps=len(test))


plt.figure(figsize=(12,6))
plt.plot(train.index, train, label='Training Data')
```

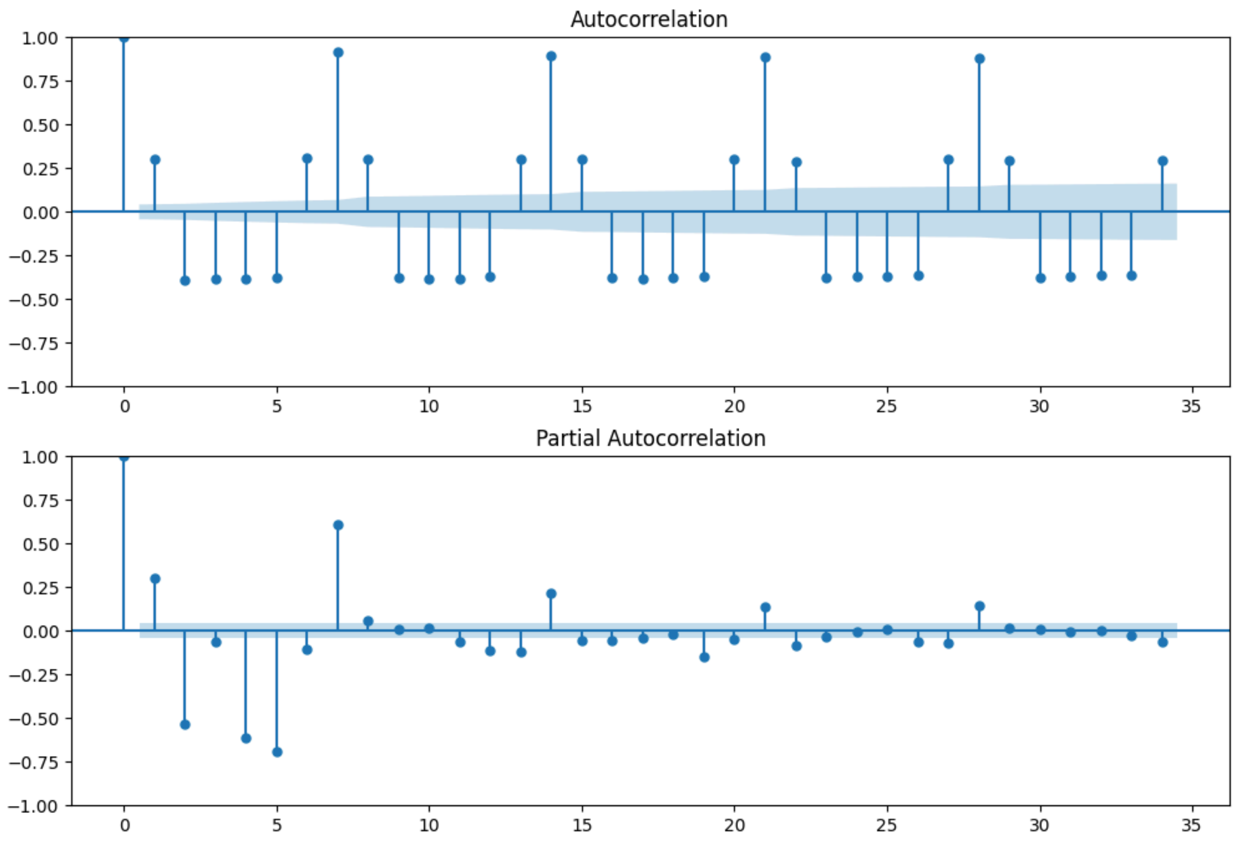
EX -8
17/04/2025

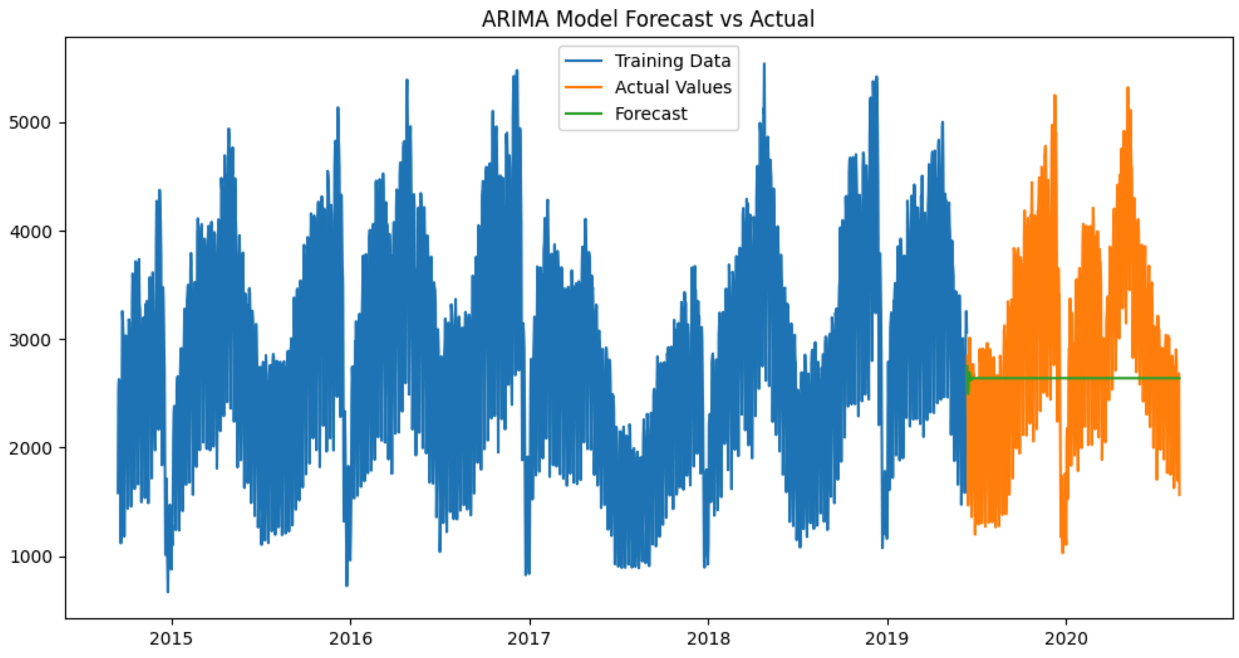
```
plt.plot(test.index, test, label='Actual Values')
plt.plot(test.index, forecast, label='Forecast')
plt.title('ARIMA Model Forecast vs Actual')
plt.legend()
plt.show()


from sklearn.metrics import mean_squared_error
rmse = np.sqrt(mean_squared_error(test, forecast))
print(f'Test RMSE: {rmse:.2f}')
```

Step 5 - `Forecasting Using ARIMA Model`

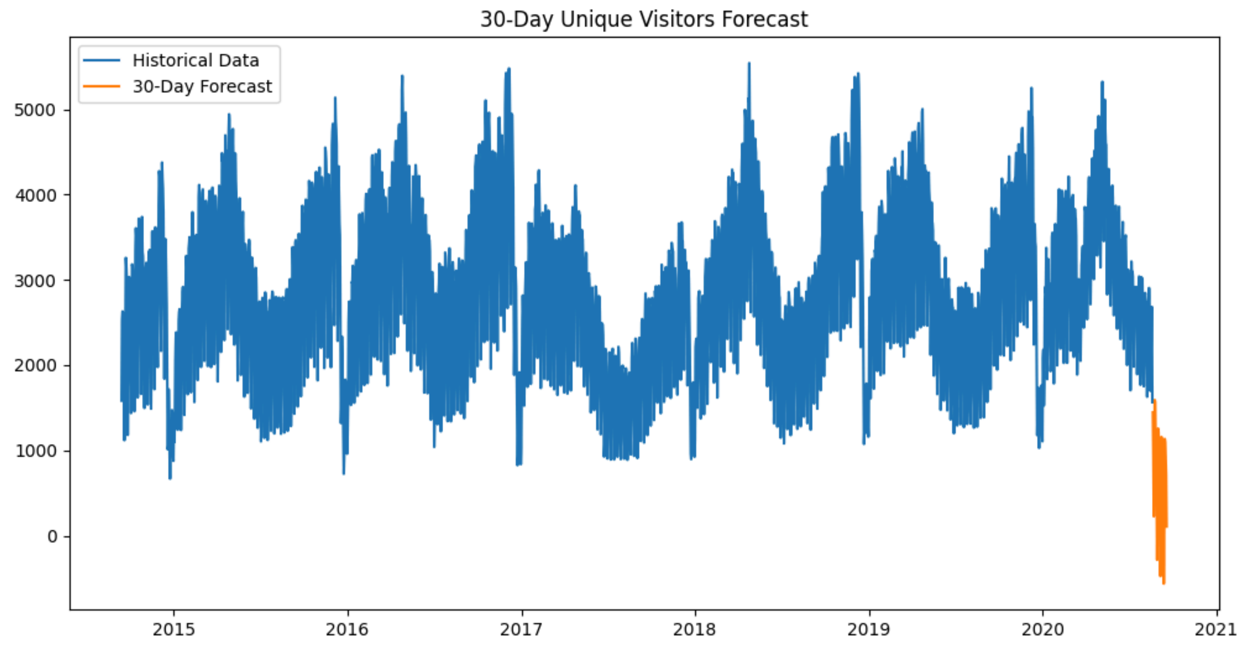
```
        final_model = SARIMAX(ts, order=(2,1,2), seasonal_order=(1,1,1,7))
        final_fit = final_model.fit()


        forecast_days = 30
        forecast = final_fit.forecast(steps=forecast_days)


        plt.figure(figsize=(12,6))
        plt.plot(ts.index, ts, label='Historical Data')
        plt.plot(pd.date_range(ts.index[-1], periods=forecast_days+1)[1:], forecast,
label='30-Day Forecast')
        plt.title('30-Day Unique Visitors Forecast')
        plt.legend()
        plt.show()
```

EX -8
17/04/2025





EX -8
17/04/2025



Result:

Thus the Program has been Executed Successfully.