In [1]:

```
!pip install pandas numpy matplotlib scikit-learn statsmodels
```

```
Requirement already satisfied: pandas in c:\users\harshini ts\appdata\local\programs\python\python314\lib\site-p
ackages (2.3.3)
Requirement already satisfied: numpy in c:\users\harshini ts\appdata\local\programs\python\python314\lib\site-pa
ckages (2.3.4)
Requirement already satisfied: matplotlib in c:\users\harshini ts\appdata\local\programs\python\python314\lib\si
te-packages (3.10.7)
Requirement already satisfied: scikit-learn in c:\users\harshini ts\appdata\local\programs\python\python314\lib\
site-packages (1.7.2)
Requirement already satisfied: statsmodels in c:\users\harshini ts\appdata\local\programs\python\python314\lib\s
ite-packages (0.14.6)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\harshini ts\appdata\local\programs\python\pyth
on314\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\harshini ts\appdata\local\programs\python\python314\lib\
site-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\harshini ts\appdata\local\programs\python\python314\li
b\site-packages (from pandas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\harshini ts\appdata\local\programs\python\python314\
lib\site-packages (from matplotlib) (1.3.3)
Requirement already satisfied: cycler>=0.10 in c:\users\harshini ts\appdata\local\programs\python\python314\lib\
site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\harshini ts\appdata\local\programs\python\python314
\lib\site-packages (from matplotlib) (4.60.1)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\harshini ts\appdata\local\programs\python\python314
\lib\site-packages (from matplotlib) (1.4.9)
Requirement already satisfied: packaging>=20.0 in c:\users\harshini ts\appdata\local\programs\python\python314\l
ib\site-packages (from matplotlib) (25.0)
Requirement already satisfied: pillow>=8 in c:\users\harshini ts\appdata\local\programs\python\python314\lib\sit
e-packages (from matplotlib) (12.0.0)
Requirement already satisfied: pyparsing>=3 in c:\users\harshini ts\appdata\local\programs\python\python314\lib\
site-packages (from matplotlib) (3.2.5)
Requirement already satisfied: scipy>=1.8.0 in c:\users\harshini ts\appdata\local\programs\python\python314\lib\
site-packages (from scikit-learn) (1.16.3)
Requirement already satisfied: joblib>=1.2.0 in c:\users\harshini ts\appdata\local\programs\python\python314\lib
\site-packages (from scikit-learn) (1.5.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\harshini ts\appdata\local\programs\python\python
314\lib\site-packages (from scikit-learn) (3.6.0)
Requirement already satisfied: patsy>=0.5.6 in c:\users\harshini ts\appdata\local\programs\python\python314\lib\
site-packages (from statsmodels) (1.0.2)
Requirement already satisfied: six>=1.5 in c:\users\harshini ts\appdata\local\programs\python\python314\lib\site
-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
```

```
[notice] A new release of pip is available: 25.2 -> 26.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [2]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.graphics.tsaplots import plot_acf

from sklearn.metrics import mean_absolute_error, mean_squared_error
```

In [3]:

```
np.random.seed(42)

date_range = pd.date_range(start="2015-01-01", periods=96, freq="M")

electricity_consumption = (
    200 +
    20 * np.sin(2 * np.pi * date_range.month / 12) +
    np.random.normal(0, 5, len(date_range))
)

temperature = (
    25 +
    10 * np.sin(2 * np.pi * (date_range.month + 3) / 12) +
    np.random.normal(0, 2, len(date_range))
)

df = pd.DataFrame({
    "Date": date_range,
    "Electricity_Consumption": electricity_consumption,
    "Temperature": temperature
})

df.set_index("Date", inplace=True)
df.head()
```

```
C:\Users\Harshini TS\AppData\Local\Temp\ipykernel_9188\3916590776.py:3: FutureWarning: 'M' is deprecated and wil
l be removed in a future version, please use 'ME' instead.
  date_range = pd.date_range(start="2015-01-01", periods=96, freq="M")
```

Out[3]:

	Electricity_Consumption	Temperature
Date		
2015-01-31	212.483571	34.252495
2015-02-28	216.629187	30.522111
2015-03-31	223.238443	25.010227
2015-04-30	224.935657	19.530826
2015-05-31	208.829233	13.509004

In [4]:

```
df.info()
df.describe()
```

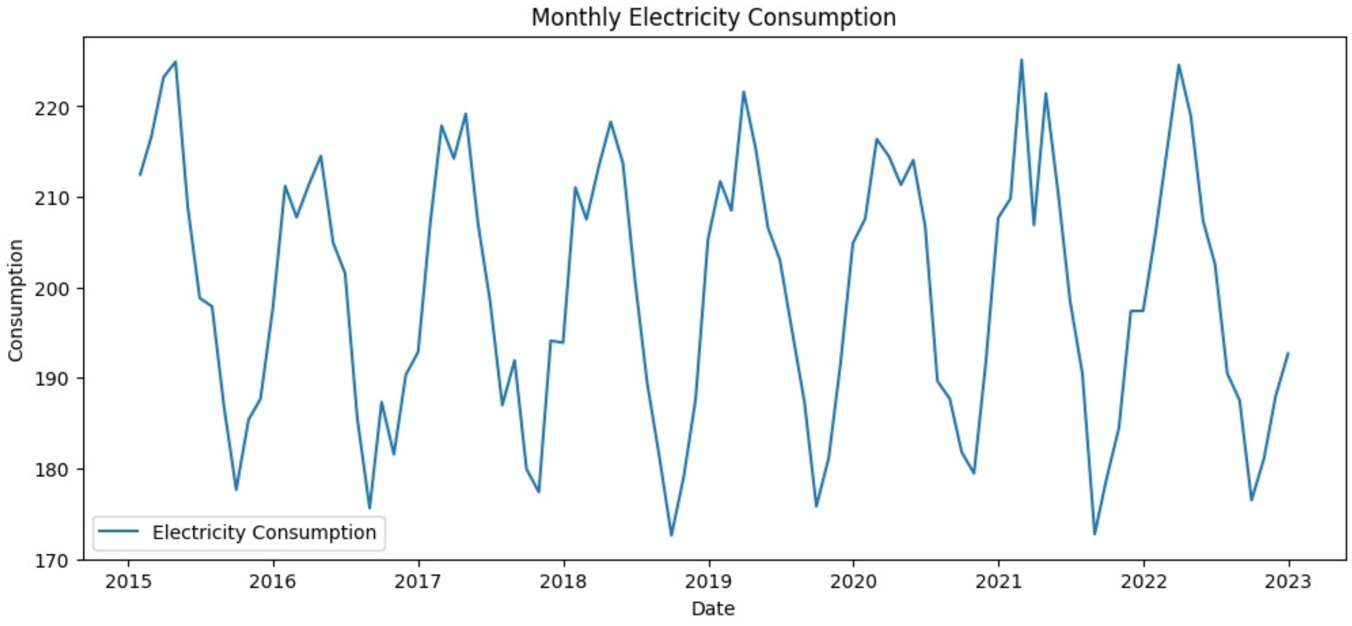
```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 96 entries, 2015-01-31 to 2022-12-31
Data columns (total 2 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Electricity_Consumption  96 non-null     float64
 1   Temperature              96 non-null     float64
dtypes: float64(2)
memory usage: 2.2 KB
```

Out[4]:

	Electricity_Consumption	Temperature
count	96.000000	96.000000
mean	199.442065	25.100963
std	14.282574	7.496634
min	172.607390	13.509004
25%	187.642863	18.260672
50%	199.843078	24.158690
75%	211.242227	31.542646
max	225.143726	40.440338

In [5]:

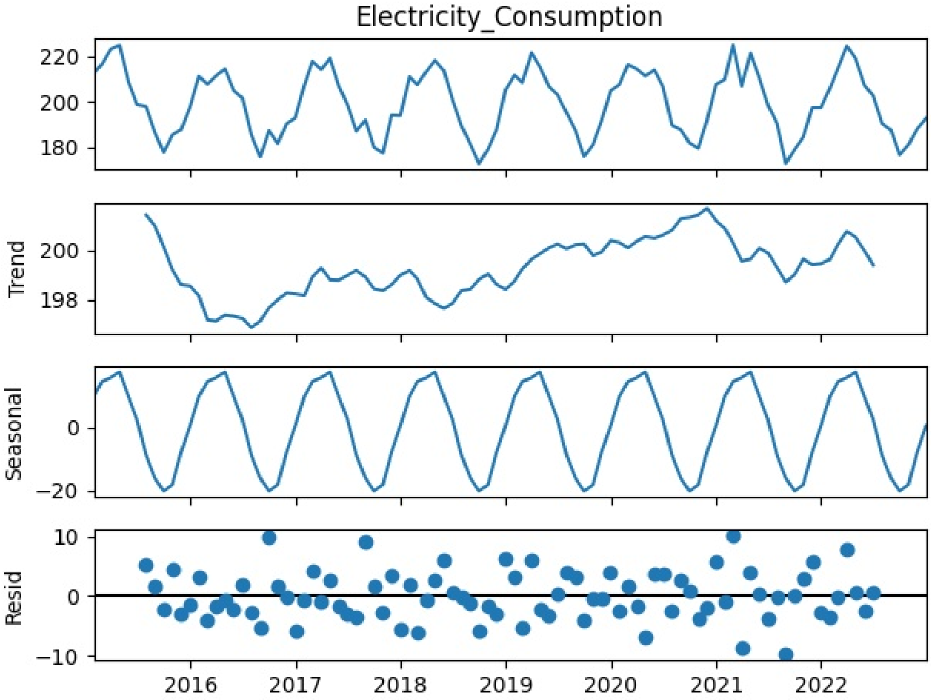
```
plt.figure(figsize=(12,5))
plt.plot(df["Electricity_Consumption"], label="Electricity Consumption")
plt.title("Monthly Electricity Consumption")
plt.xlabel("Date")
plt.ylabel("Consumption")
plt.legend()
plt.show()
```



In [6]:

```
decomposition = seasonal_decompose(
    df["Electricity_Consumption"],
    model="additive",
    period=12
)
```

```
decomposition.plot()
plt.show()
```



In [7]:

```
train_size = int(len(df) * 0.8)

train = df.iloc[:train_size]
test = df.iloc[train_size:]
```

In [8]:

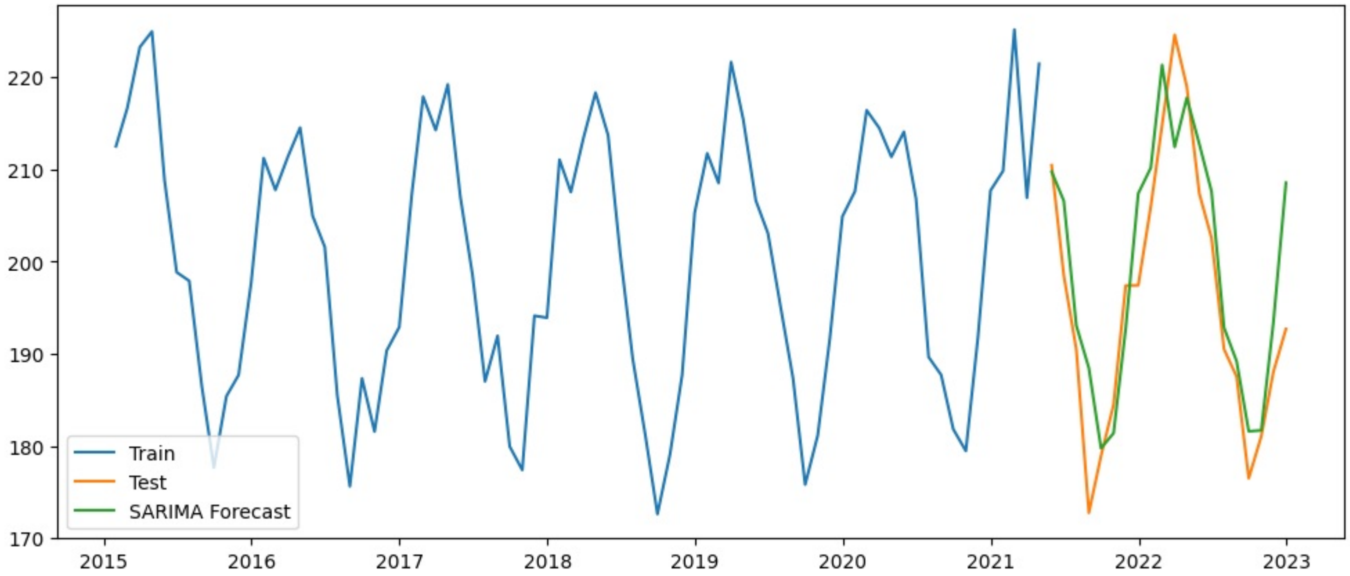
```
sarima_model = SARIMAX(
    train["Electricity_Consumption"],
    order=(1,1,1),
    seasonal_order=(1,1,1,12),
    enforce_stationarity=False,
    enforce_invertibility=False
)

sarima_result = sarima_model.fit()
print(sarima_result.summary())
```

```
C:\Users\Harshini TS\AppData\Local\Programs\Python\Python314\Lib\site-packages\statsmodels\tsa\base\tsa_model.py
:473: ValueWarning: No frequency information was provided, so inferred frequency ME will be used.
  self._init_dates(dates, freq)
C:\Users\Harshini TS\AppData\Local\Programs\Python\Python314\Lib\site-packages\statsmodels\tsa\base\tsa_model.py
:473: ValueWarning: No frequency information was provided, so inferred frequency ME will be used.
  self._init_dates(dates, freq)
                                     SARIMAX Results                                      
==========================================================================================
Dep. Variable:            Electricity_Consumption   No. Observations:                   76
Model:             SARIMAX(1, 1, 1)x(1, 1, 1, 12)   Log Likelihood                -149.488
Date:                            Fri, 06 Feb 2026   AIC                            308.977
Time:                                    13:54:11   BIC                            318.436
Sample:                                01-31-2015   HQIC                           312.566
                                     - 04-30-2021                                         
Covariance Type:                              opg                                         
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.3207      0.153     -2.095      0.036      -0.621      -0.021
ma.L1         -0.9738      0.119     -8.193      0.000      -1.207      -0.741
ar.S.L12      -0.3831      0.187     -2.047      0.041      -0.750      -0.016
ma.S.L12      -0.1318      0.320     -0.412      0.680      -0.758       0.495
sigma2        24.9091      5.600      4.448      0.000      13.934      35.884
===================================================================================
Ljung-Box (L1) (Q):                   0.26   Jarque-Bera (JB):                 0.01
Prob(Q):                              0.61   Prob(JB):                         0.99
Heteroskedasticity (H):               1.32   Skew:                             0.03
Prob(H) (two-sided):                  0.59   Kurtosis:                         3.05
===================================================================================

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

In [9]:

```
sarima_forecast = sarima_result.forecast(steps=len(test))
```

```
plt.figure(figsize=(12,5))
plt.plot(train.index, train["Electricity_Consumption"], label="Train")
plt.plot(test.index, test["Electricity_Consumption"], label="Test")
plt.plot(test.index, sarima_forecast, label="SARIMA Forecast")
plt.legend()
plt.show()
```

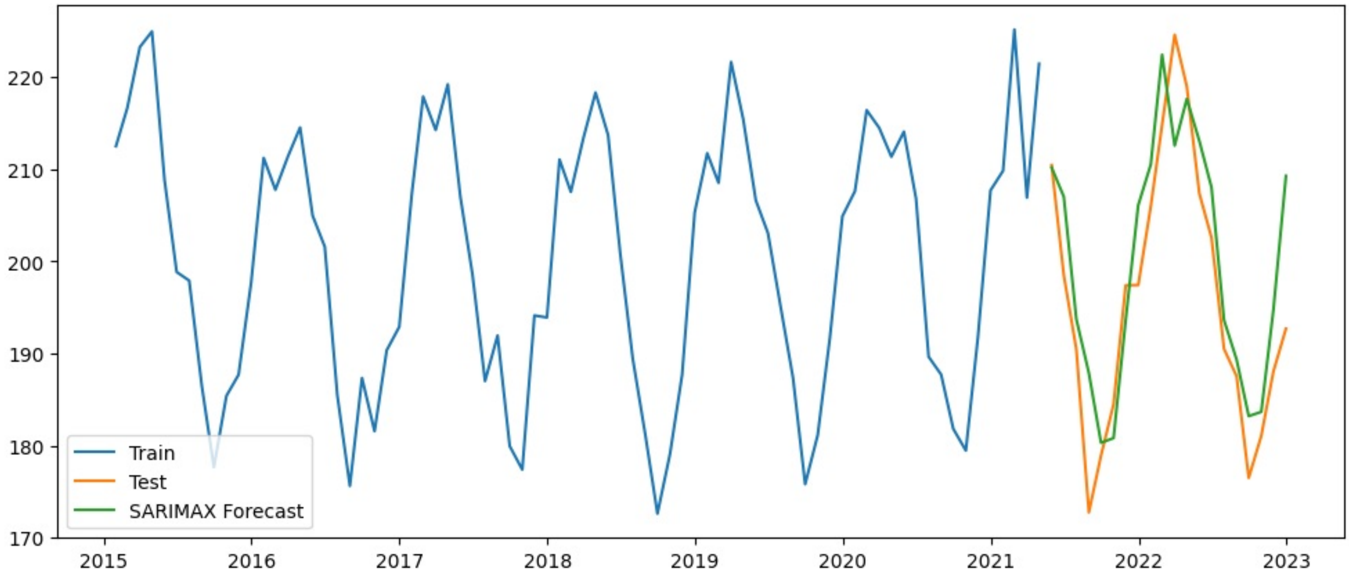


```
In [10]: sarimax_model = SARIMAX(
             train["Electricity_Consumption"],
             exog=train[["Temperature"]],
             order=(1,1,1),
             seasonal_order=(1,1,1,12),
             enforce_stationarity=False,
             enforce_invertibility=False
         )

         sarimax_result = sarimax_model.fit()
         print(sarimax_result.summary())
```

```
C:\Users\Harshini TS\AppData\Local\Programs\Python\Python314\Lib\site-packages\statsmodels\tsa\base\tsa_model.py
:473: ValueWarning: No frequency information was provided, so inferred frequency ME will be used.
  self._init_dates(dates, freq)
C:\Users\Harshini TS\AppData\Local\Programs\Python\Python314\Lib\site-packages\statsmodels\tsa\base\tsa_model.py
:473: ValueWarning: No frequency information was provided, so inferred frequency ME will be used.
  self._init_dates(dates, freq)
                                     SARIMAX Results
===========================================================================================
Dep. Variable:            Electricity_Consumption   No. Observations:                    76
Model:             SARIMAX(1, 1, 1)x(1, 1, 1, 12)   Log Likelihood                 -148.648
Date:                            Fri, 06 Feb 2026   AIC                             309.297
Time:                                    13:55:07   BIC                             320.648
Sample:                                01-31-2015   HQIC                            313.603
                                     - 04-30-2021
Covariance Type:                              opg
================================================================================
                  coef    std err          z      P>|z|       [0.025      0.975]
--------------------------------------------------------------------------------
Temperature    -0.3978      0.478     -0.831      0.406       -1.336       0.540
ar.L1          -0.3136      0.153     -2.055      0.040       -0.613      -0.015
ma.L1          -0.9521      0.083    -11.465      0.000       -1.115      -0.789
ar.S.L12       -0.3869      0.209     -1.849      0.065       -0.797       0.023
ma.S.L12       -0.1122      0.344     -0.326      0.744       -0.787       0.562
sigma2         24.3719      6.085      4.005      0.000       12.446      36.298
====================================================================================
Ljung-Box (L1) (Q):                   0.18   Jarque-Bera (JB):                  0.04
Prob(Q):                              0.67   Prob(JB):                          0.98
Heteroskedasticity (H):               1.34   Skew:                             -0.04
Prob(H) (two-sided):                  0.56   Kurtosis:                          2.89
====================================================================================

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

```
In [11]: sarimax_forecast = sarimax_result.predict(
             start=test.index[0],
             end=test.index[-1],
             exog=test[["Temperature"]]
         )

         plt.figure(figsize=(12,5))
```

```python
plt.plot(train.index, train["Electricity_Consumption"], label="Train")
plt.plot(test.index, test["Electricity_Consumption"], label="Test")
plt.plot(test.index, sarimax_forecast, label="SARIMAX Forecast")
plt.legend()
plt.show()
```



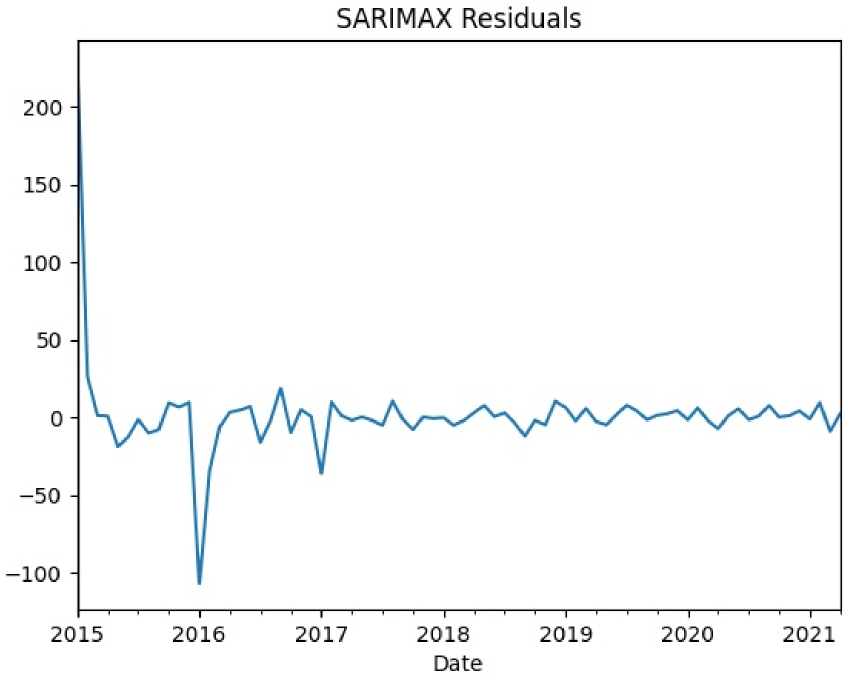
In [12]:

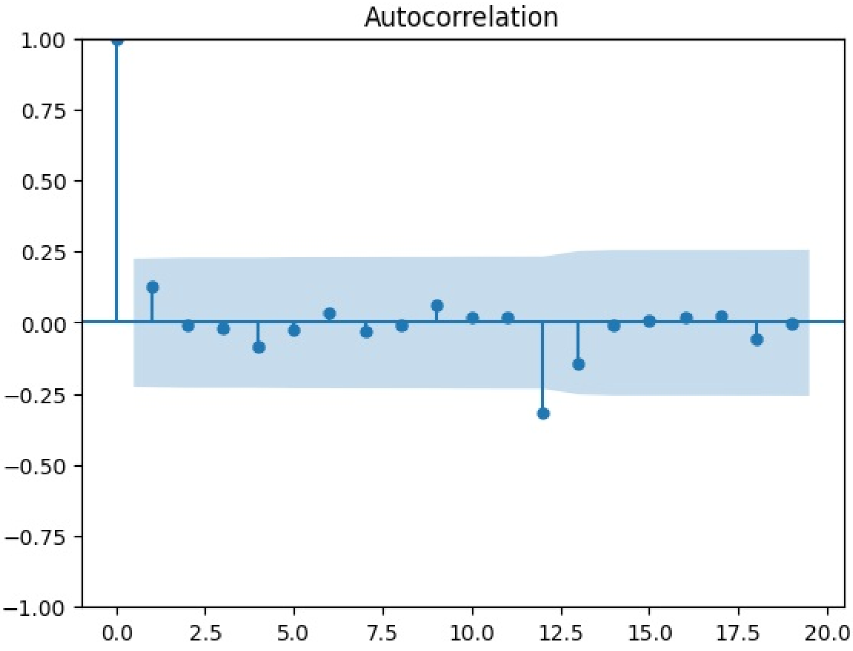
```python
def evaluate(actual, predicted, name):
    mae = mean_absolute_error(actual, predicted)
    rmse = np.sqrt(mean_squared_error(actual, predicted))
    print(f"{name} Results")
    print("MAE :", round(mae, 2))
    print("RMSE:", round(rmse, 2))
    print("-" * 30)

evaluate(test["Electricity_Consumption"], sarima_forecast, "SARIMA")
evaluate(test["Electricity_Consumption"], sarimax_forecast, "SARIMAX")
```

```
SARIMA Results
MAE : 5.56
RMSE: 7.17
------------------------------
SARIMAX Results
MAE : 5.94
RMSE: 7.38
------------------------------
```

In [13]:

```python
sarimax_result.resid.plot(title="SARIMAX Residuals")
plt.show()

plot_acf(sarimax_result.resid.dropna())
plt.show()
```




Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

In [ ]: