In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.graphics.tsaplots import plot_acf
from sklearn.metrics import mean_absolute_error, mean_squared_error
```

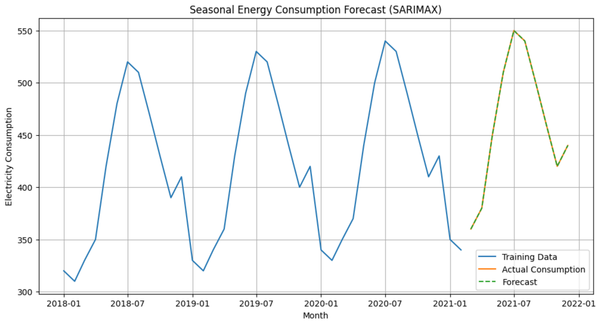
In [2]:
```
df = pd.read_csv(r"C:\Users\Logesh\Downloads\Month,Electricity_Consumption,Tempe.csv")
```

In [3]:
```
print("Columns in CSV:", df.columns)
df['Month'] = pd.to_datetime(df['Month'])
df.set_index('Month', inplace=True)
print("\nData Preview:")
print(df.head())
y = df['Electricity_Consumption']
exog = df[['Temperature', 'Holiday']]
train_size = int(len(df) * 0.8)
y_train = y.iloc[:train_size]
y_test = y.iloc[train_size:]
exog_train = exog.iloc[:train_size]
exog_test = exog.iloc[train_size:]
model = SARIMAX(
    y_train,
    exog=exog_train,
    order=(1, 1, 1),
    seasonal_order=(1, 1, 1, 12),
    enforce_stationarity=False,
    enforce_invertibility=False
)
results = model.fit()
print("\nMODEL SUMMARY:")
print(results.summary())
forecast = results.predict(
    start=y_test.index[0],
    end=y_test.index[-1],
    exog=exog_test
)
mae = mean_absolute_error(y_test, forecast)
rmse = np.sqrt(mean_squared_error(y_test, forecast))
print("\nEvaluation Metrics:")
print("MAE :", mae)
print("RMSE:", rmse)
plt.figure(figsize=(12, 6))
plt.plot(y_train, label='Training Data')
plt.plot(y_test, label='Actual Consumption')
plt.plot(forecast, label='Forecast', linestyle='--')
plt.title('Seasonal Energy Consumption Forecast (SARIMAX)')
plt.xlabel('Month')
plt.ylabel('Electricity Consumption')
plt.legend()
plt.grid(True)
plt.show()
```

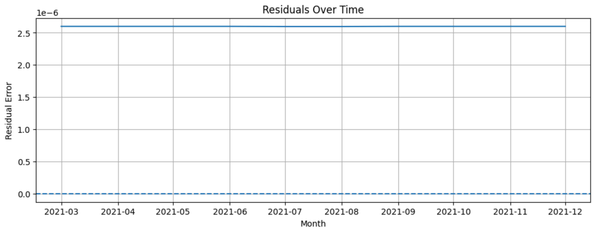
```
Columns in CSV: Index(['Month', 'Electricity_Consumption', 'Temperature', 'Holiday'], dtype='object')

Data Preview:
            Electricity_Consumption  Temperature  Holiday
Month
2018-01-01                      320           18        1
2018-02-01                      310           20        0
2018-03-01                      330           25        0
2018-04-01                      350           30        0
2018-05-01                      420           35        0
```

```
C:\Users\Logesh\AppData\Roaming\Python\Python313\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
  self._init_dates(dates, freq)
C:\Users\Logesh\AppData\Roaming\Python\Python313\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred frequency MS will be used.
  self._init_dates(dates, freq)
C:\Users\Logesh\AppData\Roaming\Python\Python313\site-packages\statsmodels\tsa\statespace\sarimax.py:866: UserWarning: Too few observations to estimate starting parameters for seasonal ARMA. All parameters except for variances will be set to zeros.
  warn('Too few observations to estimate starting parameters%s.'
C:\Users\Logesh\AppData\Roaming\Python\Python313\site-packages\statsmodels\base\model.py:607: ConvergenceWarning: Maximum Likelihood optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

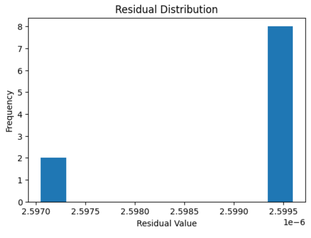
```
MODEL SUMMARY:
                                     SARIMAX Results
==========================================================================================
Dep. Variable:              Electricity_Consumption   No. Observations:                   38
Model:             SARIMAX(1, 1, 1)x(1, 1, 1, 12)   Log Likelihood                 131.604
Date:                            Sat, 07 Feb 2026   AIC                           -249.207
Time:                                    14:06:51   BIC                           -246.422
Sample:                                01-01-2018   HQIC                          -250.963
                                     - 02-01-2021
Covariance Type:                               opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Temperature    -2.6e-06   1.07e-06     -2.422      0.015    -4.7e-06   -4.96e-07
Holiday               0        nan        nan        nan         nan         nan
ar.L1            0.0003   2.38e-16   1.08e+12      0.000       0.000       0.000
ma.L1         -1.499e-13   4.76e-16   -314.879      0.000   -1.51e-13   -1.49e-13
ar.S.L12         0.0010   1.84e-12   5.31e+08      0.000       0.001       0.001
ma.S.L12      -5.533e-12   2.04e-16  -2.71e+04      0.000   -5.53e-12   -5.53e-12
sigma2         4.131e-12   6.72e-10      0.006      0.995   -1.31e-09    1.32e-09
===================================================================================
Ljung-Box (L1) (Q):                   0.02   Jarque-Bera (JB):                 0.21
Prob(Q):                              0.90   Prob(JB):                         0.90
Heteroskedasticity (H):               1.00   Skew:                             0.13
Prob(H) (two-sided):                  1.00   Kurtosis:                         3.62
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
[2] Covariance matrix is singular or near-singular, with condition number    inf. Standard errors may be unstable.

Evaluation Metrics:
MAE : 2.5990875712977876e-06
RMSE: 2.5990877707878483e-06
```



In [4]:
```
residuals = (y_test - forecast).dropna()
plt.figure(figsize=(12, 4))
plt.plot(residuals)
plt.axhline(0, linestyle='--')
plt.title('Residuals Over Time')
plt.xlabel('Month')
plt.ylabel('Residual Error')
plt.grid(True)
plt.show()
```

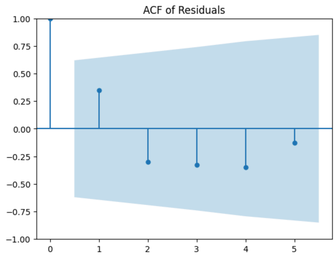


In [5]:
```
plt.figure(figsize=(6, 4))
plt.hist(residuals, bins=10)
plt.title('Residual Distribution')
plt.xlabel('Residual Value')
plt.ylabel('Frequency')
plt.show()
```



In [6]:
```
if len(residuals) > 5:
    max_lags = min(5, len(residuals) - 1)

    plt.figure(figsize=(8, 4))
    plot_acf(residuals, lags=max_lags)
    plt.title('ACF of Residuals')
    plt.show()
else:
    print("⚠️ ACF plot skipped due to very small residual size")


print("\n✅ ALL OUTPUTS GENERATED SUCCESSFULLY")
```

```
<Figure size 800x400 with 0 Axes>
```




```
✅ ALL OUTPUTS GENERATED SUCCESSFULLY
```

In [ ]: