In [435]:

```
from pmdarima import auto_arima
import pandas as pd
import matplotlib.pyplot as plt
import requests
import io
import seaborn as sns
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.stattools import adfuller
import statsmodels.api as sm
from statsmodels.tsa.seasonal import seasonal_decompose




# URL of the dataset
url = "https://fred.stlouisfed.org/graph/fredgraph.csv?id=ICSA"

# Fetching the data from the URL
response = requests.get(url)

# Reading the data into a DataFrame
df = pd.read_csv(io.StringIO(response.text))

print(df.head(10))  # Displays the first 10 rows
```

```
         DATE    ICSA
0  1967-01-07  208000
1  1967-01-14  207000
2  1967-01-21  217000
3  1967-01-28  204000
4  1967-02-04  216000
5  1967-02-11  229000
6  1967-02-18  229000
7  1967-02-25  242000
8  1967-03-04  310000
9  1967-03-11  241000
```

In [436]:

```
df.shape
```

Out[436]:

```
(2980, 2)
```

In [437]:

```python
# Convert 'DATE' column to datetime
print('DATE data type before conversion:')
print(df['DATE'].dtypes)
df['DATE'] = pd.to_datetime(df['DATE'])
print('DATE data type after conversion:')
print(df['DATE'].dtypes)
```

```
DATE data type before conversion:
object
DATE data type after conversion:
datetime64[ns]
```

In [438]:

```python
# Check for missing values
missing_values = df.isnull().sum()
# Check for duplicates
duplicates = df.duplicated().sum()

# Handling missing values (if any, here we just print them)
print(f"Missing values:\n{missing_values}")
print(f"Duplicate values:\n{duplicates}")
```

```
Missing values:
DATE    0
ICSA    0
dtype: int64
Duplicate values:
0
```

Here there are no missing or duplicate values

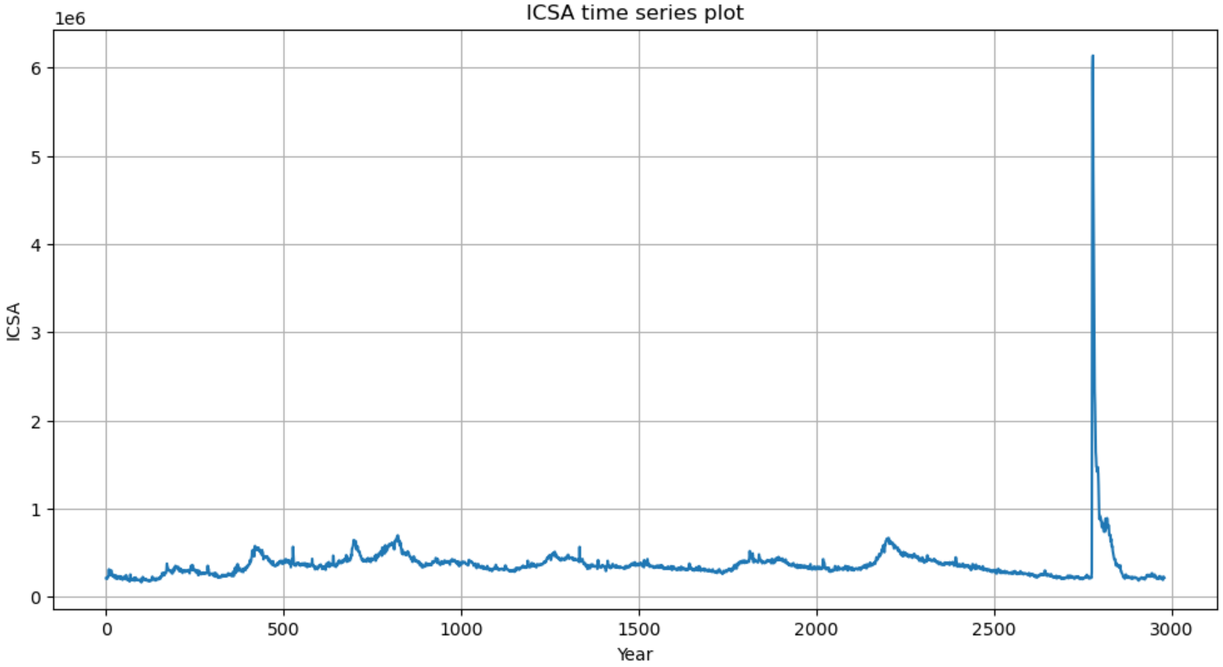
In [439]:

```python
print(f"Statistical Summary:")
print(df.describe())
```

```
Statistical Summary:
                      DATE          ICSA
count                 2980  2.980000e+03
mean   1995-07-25 12:00:00  3.654225e+05
min    1967-01-07 00:00:00  1.620000e+05
25%    1981-04-16 06:00:00  2.910000e+05
50%    1995-07-25 12:00:00  3.420000e+05
75%    2009-11-01 18:00:00  3.990000e+05
max    2024-02-10 00:00:00  6.137000e+06
std                    NaN  2.418473e+05
```

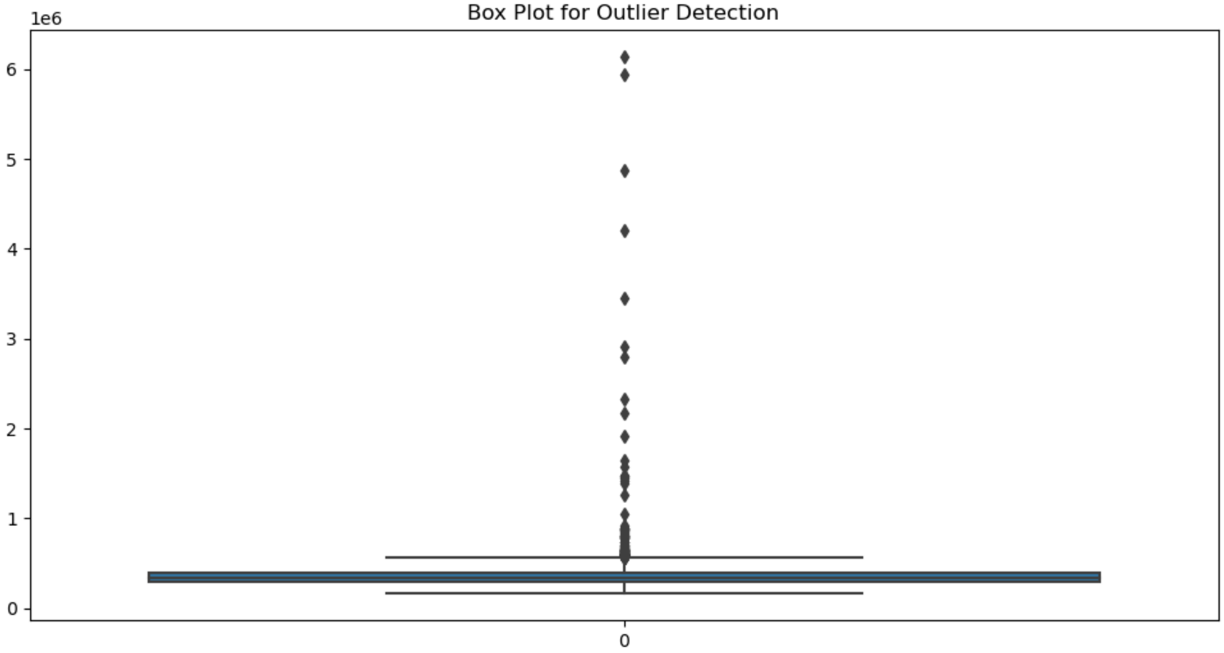
In [440]:

```
plt.figure(figsize=(12, 6))
plt.plot(df.index, df['ICSA'])
plt.title('ICSA time series plot')
plt.xlabel('Year')
plt.ylabel('ICSA')
plt.grid(True)
plt.show()
```



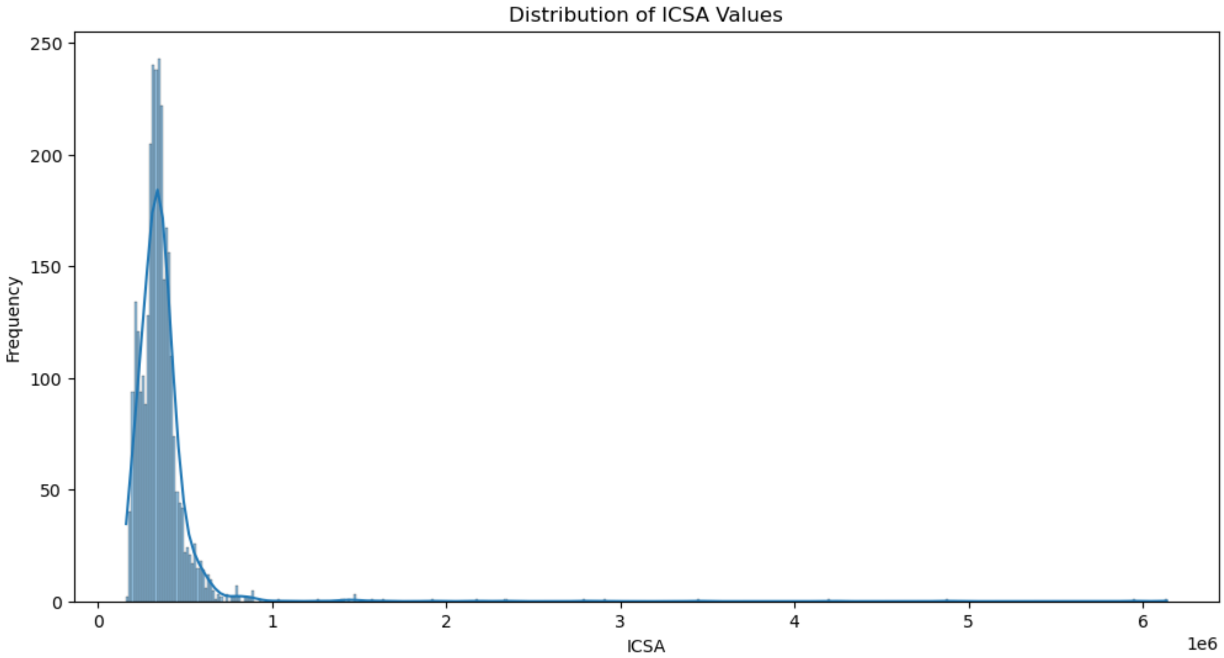
In [441]:

```
# Outlier Detection
plt.figure(figsize=(12, 6))
sns.boxplot(df['ICSA'])
plt.title('Box Plot for Outlier Detection')
plt.show()
```



In [442]:

```
# Distribution of ICSA values
plt.figure(figsize=(12, 6))
sns.histplot(df['ICSA'], kde=True)
plt.title('Distribution of ICSA Values')
plt.xlabel('ICSA')
plt.ylabel('Frequency')
plt.show()
```

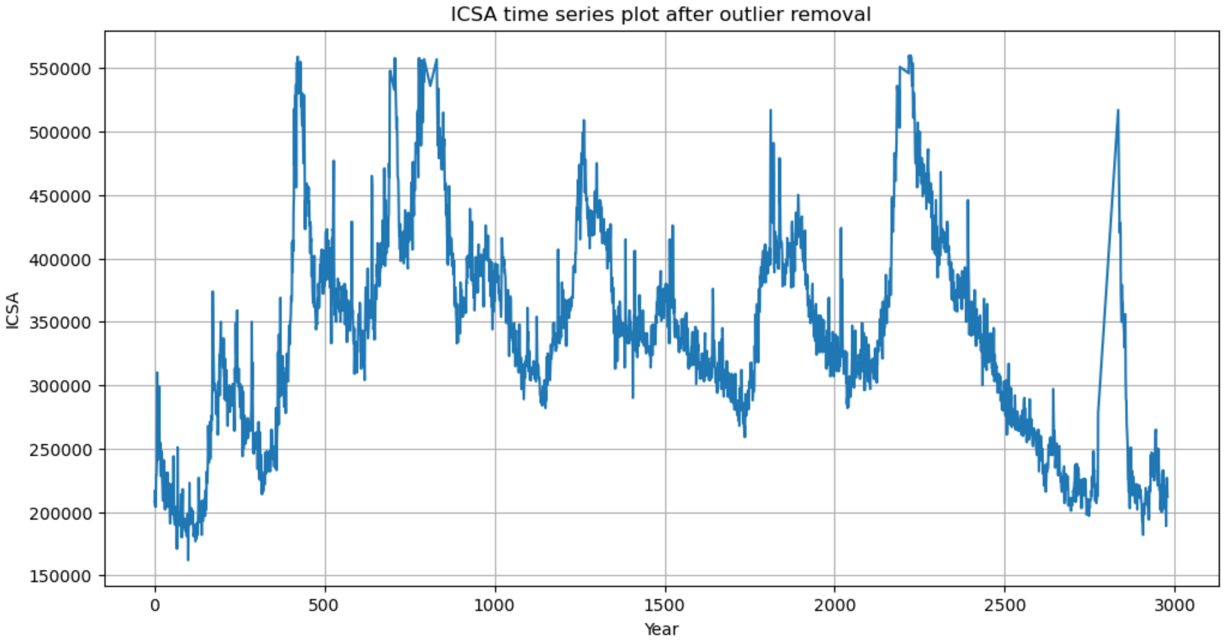


In [443]:

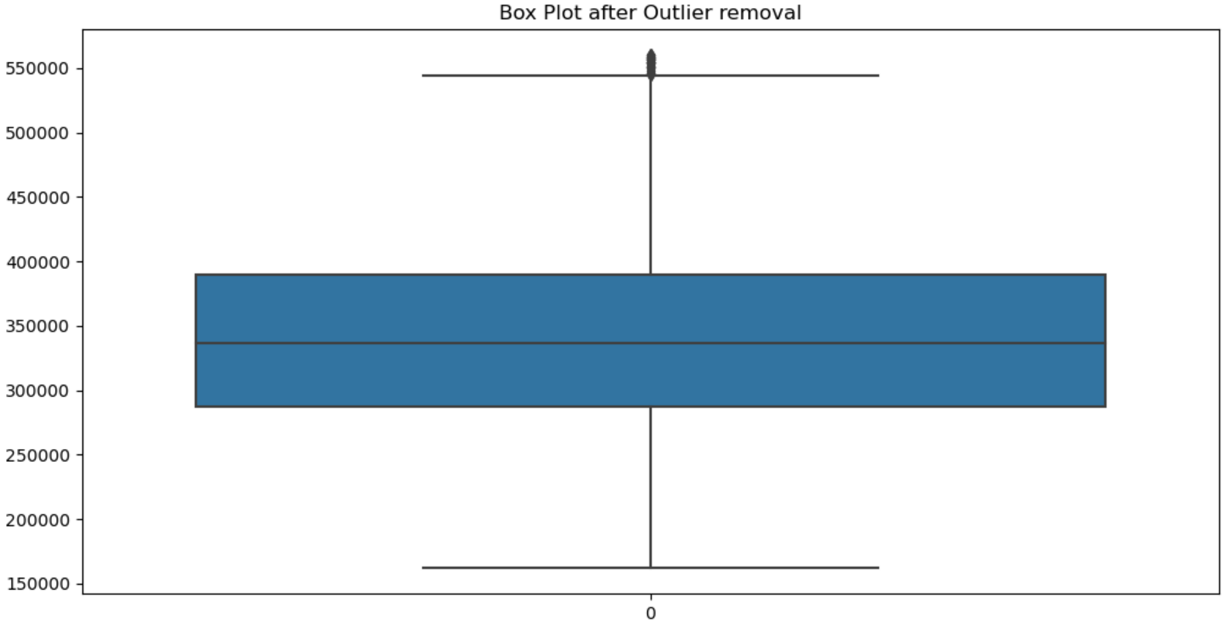
```
# removing outliers
Q1 = df['ICSA'].quantile(0.25)
Q3 = df['ICSA'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
print(lower_bound,upper_bound)
df_filt = df[(df['ICSA'] >= lower_bound) & (df['ICSA'] <= upper_bound)]
```

```
129000.0 561000.0
```

In [444]:
```
plt.figure(figsize=(12, 6))
plt.plot(df_filt.index, df_filt['ICSA'])
plt.title('ICSA time series plot after outlier removal')
plt.xlabel('Year')
plt.ylabel('ICSA')
plt.grid(True)
plt.show()
```

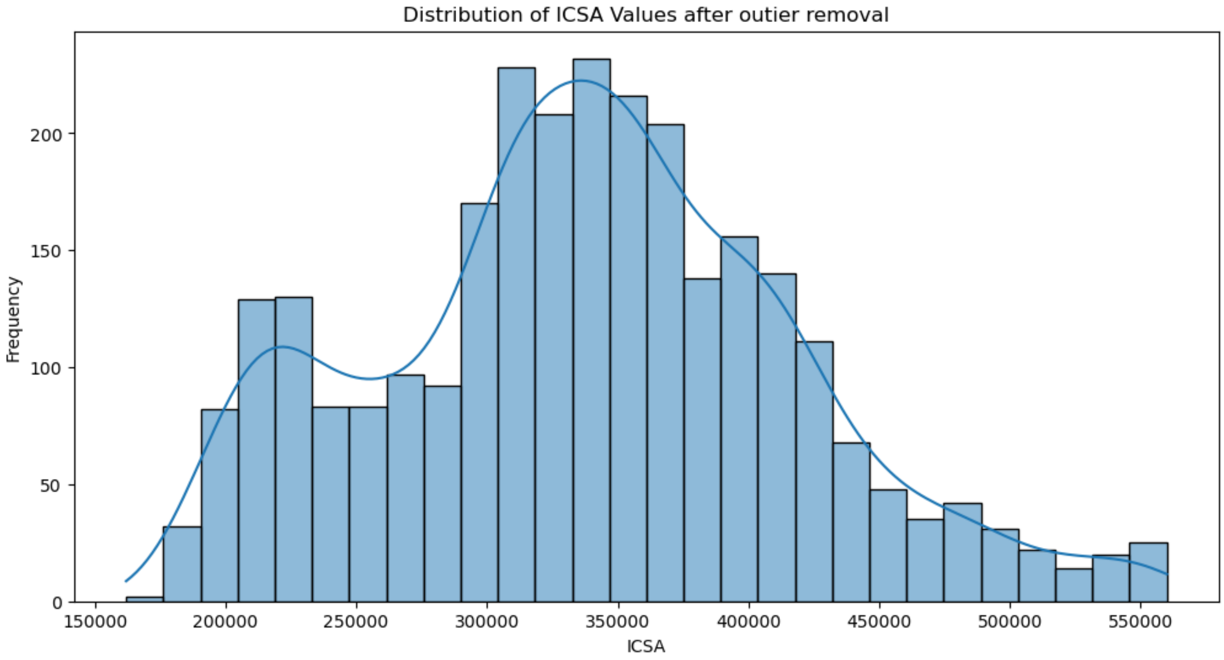


In [445]:
```
# Boxpot after outlier removal
plt.figure(figsize=(12, 6))
sns.boxplot(df_filt['ICSA'])
plt.title('Box Plot after Outlier removal')
plt.show()
```



In [446]:

```
# Distribution of ICSA values after outlier removal
plt.figure(figsize=(12, 6))
sns.histplot(df_filt['ICSA'], kde=True)
plt.title('Distribution of ICSA Values after outier removal')
plt.xlabel('ICSA')
plt.ylabel('Frequency')
plt.show()
```



In [447]:

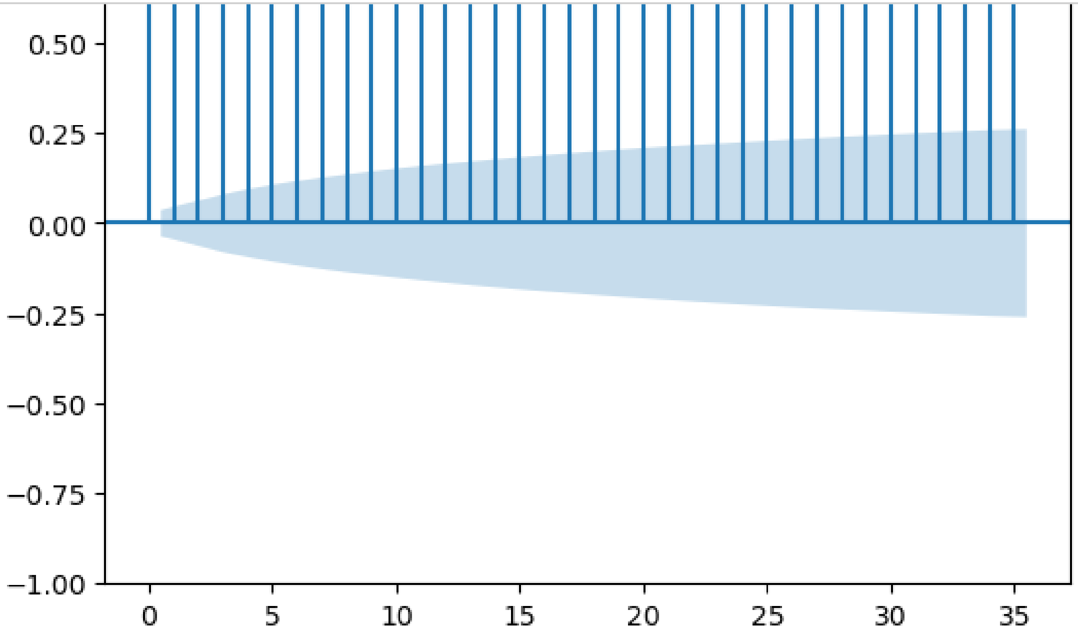
```
# Checking for stationarity and making the Series Stationary if necessary
result = adfuller(df_filt['ICSA'])
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print('Critical Values:')
for key, value in result[4].items():
    print(f'    {key}: {value}')
```

```
ADF Statistic: -3.382131
p-value: 0.011583
Critical Values:
    1%: -3.4326595388027648
    5%: -2.8625603948435945
    10%: -2.5673131867249634
```

As p-value is less than 0.05 and absolute of ADF Statistic (-3.38) < absolute of Critical Value (5%) (-2.862), time series is stationary.

In [448]:

```
plot_acf(df_filt['ICSA'])
plot_pacf(df_filt['ICSA'])
plt.show()
```



In [449]:

```
df.dropna(inplace=True)
```

In [450]:

```
from sklearn.model_selection import train_test_split
train, test = train_test_split(df_filt['ICSA'], test_size=0.2, random_state=45)
```

In [451]:

```
model = ARIMA(train,order=(2, 0, 2))
results = model.fit()
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473:
ValueWarning: An unsupported index was provided and will be ignored when e.g. fore
casting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473:
ValueWarning: An unsupported index was provided and will be ignored when e.g. fore
casting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473:
ValueWarning: An unsupported index was provided and will be ignored when e.g. fore
casting.
  self._init_dates(dates, freq)
```

In [452]:
```
print(results.summary())
```

```
                               SARIMAX Results                                
==============================================================================
Dep. Variable:                   ICSA   No. Observations:                 2270
Model:                 ARIMA(2, 0, 2)   Log Likelihood              -28808.227
Date:                Thu, 22 Feb 2024   AIC                          57628.454
Time:                        06:35:34   BIC                          57662.820
Sample:                             0   HQIC                         57640.992
                               - 2270                                         
Covariance Type:                  opg                                         
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const       3.375e+05   1775.535    190.079      0.000    3.34e+05    3.41e+05
ar.L1          0.0492      0.427      0.115      0.908      -0.787       0.886
ar.L2          0.7103      0.309      2.297      0.022       0.104       1.316
ma.L1         -0.0232      0.428     -0.054      0.957      -0.862       0.816
ma.L2         -0.7189      0.317     -2.267      0.023      -1.340      -0.097
sigma2      6.184e+09      0.002   2.71e+12      0.000    6.18e+09    6.18e+09
==================================================================================
=
Ljung-Box (L1) (Q):                   0.55   Jarque-Bera (JB):                16.8
1
Prob(Q):                              0.46   Prob(JB):                         0.0
0
Heteroskedasticity (H):               0.89   Skew:                             0.2
0
Prob(H) (two-sided):                  0.11   Kurtosis:                         2.8
4
==================================================================================
=

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-ste
p).
[2] Covariance matrix is singular or near-singular, with condition number 4.6e+27.
Standard errors may be unstable.
```

In [453]:

```
for i in range(1, 4):
    df_filt[f'ICSA_lag{i}'] = df_filt['ICSA'].shift(i)

df_filt.dropna(inplace=True)
X = df_filt[['ICSA_lag1', 'ICSA_lag2', 'ICSA_lag3']]
Y = df_filt['ICSA']

X = sm.add_constant(X)

Xtrain, Xtest, Ytrain, Ytest = train_test_split(X, Y, test_size=0.2, random_state=4

model = sm.OLS(Ytrain, Xtrain)
results = model.fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   ICSA   R-squared:                       0.958
Model:                            OLS   Adj. R-squared:                  0.958
Method:                 Least Squares   F-statistic:                 1.716e+04
Date:                Thu, 22 Feb 2024   Prob (F-statistic):               0.00
Time:                        06:35:35   Log-Likelihood:                -25211.
No. Observations:                2268   AIC:                         5.043e+04
Df Residuals:                    2264   BIC:                         5.045e+04
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       4862.2616   1506.593      3.227      0.001    1907.813    7816.710
ICSA_lag1      0.6993      0.020     34.372      0.000       0.659       0.739
ICSA_lag2      0.1031      0.026      4.020      0.000       0.053       0.153
ICSA_lag3      0.1834      0.021      8.635      0.000       0.142       0.225
==============================================================================
Omnibus:                     1463.714   Durbin-Watson:                   1.971
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            85826.171
Skew:                           2.341   Prob(JB):                         0.00
Kurtosis:                      32.771   Cond. No.                     2.64e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly s
pecified.
[2] The condition number is large, 2.64e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

```
C:\Users\Akhila Markunda\AppData\Local\Temp\ipykernel_15852\3266025240.py:2: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.o
rg/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df_filt[f'ICSA_lag{i}'] = df_filt['ICSA'].shift(i)
C:\Users\Akhila Markunda\AppData\Local\Temp\ipykernel_15852\3266025240.py:2: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.o
rg/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df_filt[f'ICSA_lag{i}'] = df_filt['ICSA'].shift(i)
C:\Users\Akhila Markunda\AppData\Local\Temp\ipykernel_15852\3266025240.py:2: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.o
rg/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df_filt[f'ICSA_lag{i}'] = df_filt['ICSA'].shift(i)
C:\Users\Akhila Markunda\AppData\Local\Temp\ipykernel_15852\3266025240.py:4: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.o
rg/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df_filt.dropna(inplace=True)
```

In [454]:

```python
Ypred = results.predict(Xtest)
print("The forecasted values are")
print(Ypred)
```

```
The forecasted values are
2666    227364.099630
1817    480688.972285
1464    329879.210121
261     274443.463596
487     378117.046020
            ...
1753    304574.522611
85      199371.191380
104     216353.720673
2362    378954.875396
1190    357709.960042
Length: 567, dtype: float64
```

In [455]:

```
url3 = "https://fred.stlouisfed.org/graph/fredgraph.csv?id=IC4WSA"
```

In [456]:

```
response2 = requests.get(url3)
df_2 = pd.read_csv(io.StringIO(response2.text))
print(df_2.head(10))
```

```
         DATE  IC4WSA
0  1967-01-28  209000
1  1967-02-04  211000
2  1967-02-11  216500
3  1967-02-18  219500
4  1967-02-25  229000
5  1967-03-04  252500
6  1967-03-11  255500
7  1967-03-18  259500
8  1967-03-25  260750
9  1967-04-01  248000
```

In [457]:

```
# Check for missing values
missing_values = df_2.isnull().sum()
# Check for duplicates
duplicates = df_2.duplicated().sum()

# Handling missing values (if any, here we just print them)
print(f"Missing values:\n{missing_values}")
print(f"Duplicate values:\n{duplicates}")
```

```
Missing values:
DATE      0
IC4WSA    0
dtype: int64
Duplicate values:
0
```

Here there are no missing or duplicate values in IC4WSA data.

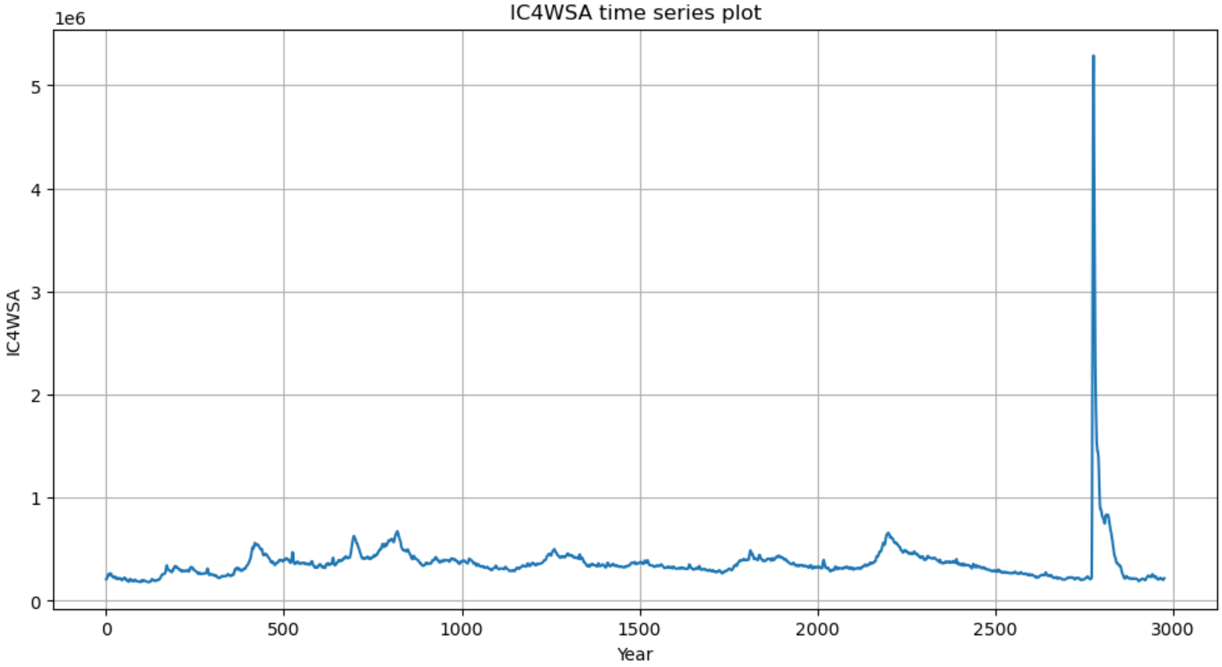
In [458]:

```
print(f"Statistical Summary of IC4WSA data:")
print(df_2.describe())
```

```
Statistical Summary of IC4WSA data:
             IC4WSA
count  2.977000e+03
mean   3.655887e+05
std    2.289940e+05
min    1.790000e+05
25%    2.910000e+05
50%    3.415000e+05
75%    3.990000e+05
max    5.288250e+06
```

In [459]:

```
plt.figure(figsize=(12, 6))
plt.plot(df_2.index, df_2['IC4WSA'])
plt.title('IC4WSA time series plot')
plt.xlabel('Year')
plt.ylabel('IC4WSA')
plt.grid(True)
plt.show()
```



In [460]:

```
df_2['DATE'] = pd.to_datetime(df_2['DATE'])
df_merge = pd.merge(df_filt, df_2, on='DATE', how='inner')
print(df_2.dtypes)
```

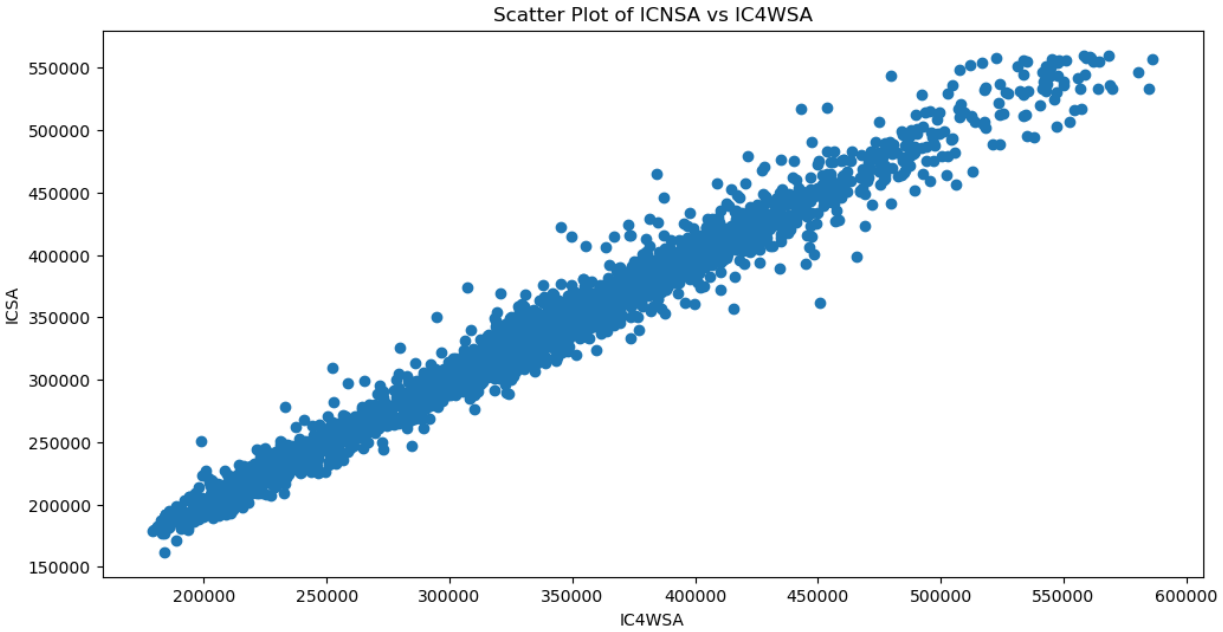
```
DATE      datetime64[ns]
IC4WSA             int64
dtype: object
```

In [461]:

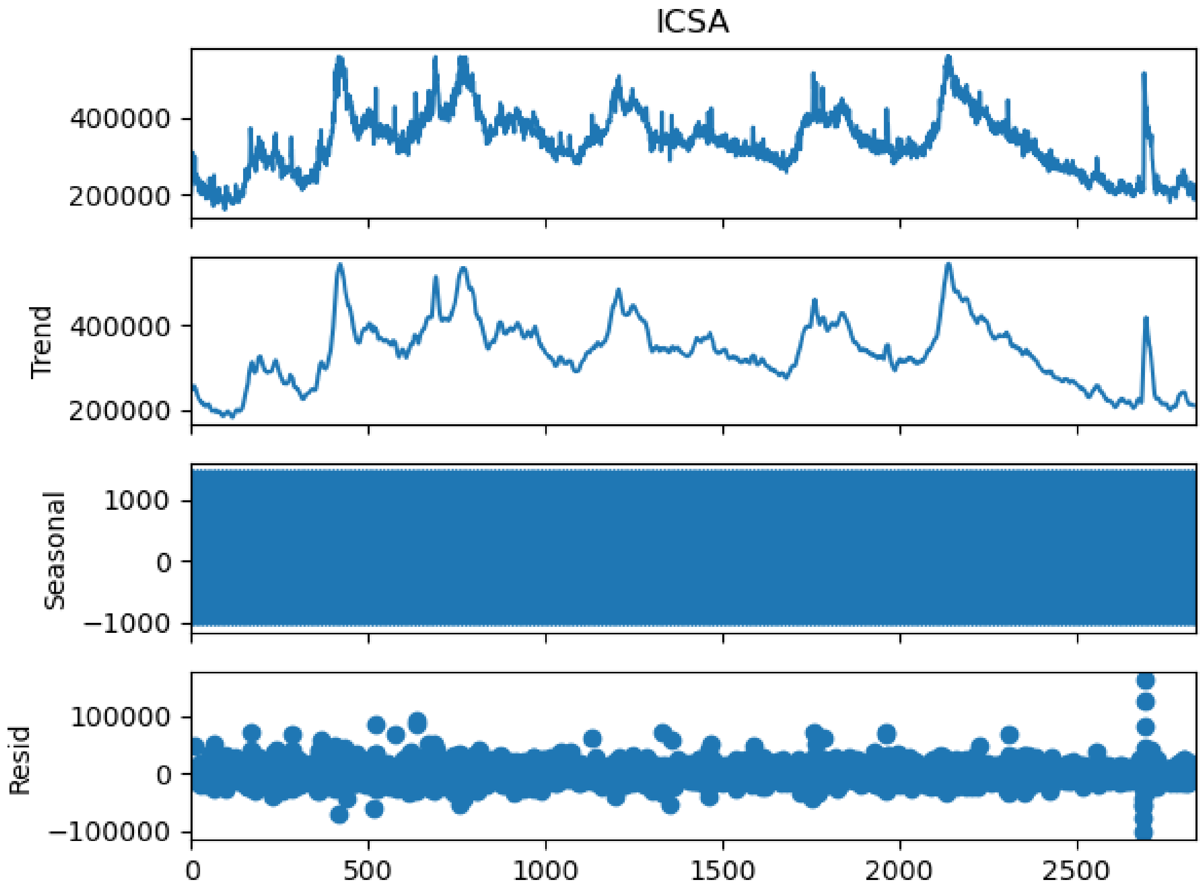
```
corr = df_merge[['ICSA','IC4WSA']].corr()
```

In [462]:

```
plt.figure(figsize=(12, 6))
plt.scatter(df_merge['IC4WSA'], df_merge['ICSA'])
plt.xlabel('IC4WSA')
plt.ylabel('ICSA')
plt.title('Scatter Plot of ICNSA vs IC4WSA')
plt.show()
```



In [463]:
```
period_s = 12
decomp_result = seasonal_decompose(df_merge['ICSA'], model='additive', period=perio
decomp_result.plot()
plt.show()
```



In [469]:
```
X = df_merge['IC4WSA']
Y = df_merge['ICSA']

Xtrain, Xtest, Ytrain, Ytest = train_test_split(X, Y, test_size=0.2, random_state=4
```

In [472]:
```
model = ARIMA(Xtrain,order=(2, 0, 2))
result = model.fit()
print(result.summary())
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473:
ValueWarning: An unsupported index was provided and will be ignored when e.g. fore
casting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473:
ValueWarning: An unsupported index was provided and will be ignored when e.g. fore
casting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473:
ValueWarning: An unsupported index was provided and will be ignored when e.g. fore
casting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\statespace\sarimax.py:9
66: UserWarning: Non-stationary starting autoregressive parameters found. Using ze
ros as starting parameters.
  warn('Non-stationary starting autoregressive parameters'
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\statespace\sarimax.py:9
78: UserWarning: Non-invertible starting MA parameters found. Using zeros as start
ing parameters.
  warn('Non-invertible starting MA parameters found.'
                               SARIMAX Results
==============================================================================
Dep. Variable:                 IC4WSA   No. Observations:                 2268
Model:                 ARIMA(2, 0, 2)   Log Likelihood              -28799.071
Date:                Thu, 22 Feb 2024   AIC                          57610.142
Time:                        06:37:49   BIC                          57644.502
Sample:                             0   HQIC                         57622.679
                               - 2268
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const       3.374e+05   1852.790    182.087      0.000    3.34e+05    3.41e+05
ar.L1         -0.1285      0.197     -0.651      0.515      -0.515       0.258
ar.L2          0.8486      0.192      4.423      0.000       0.473       1.225
ma.L1          0.1467      0.206      0.712      0.477      -0.257       0.551
ma.L2         -0.8359      0.201     -4.149      0.000      -1.231      -0.441
sigma2       6.28e+09      0.003    2.1e+12      0.000    6.28e+09    6.28e+09
==================================================================================
=
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):                21.4
3
Prob(Q):                              0.98   Prob(JB):                         0.0
0
Heteroskedasticity (H):               0.90   Skew:                             0.2
3
Prob(H) (two-sided):                  0.13   Kurtosis:                         2.9
0
==================================================================================
=

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-ste
p).
[2] Covariance matrix is singular or near-singular, with condition number 1.8e+27.
Standard errors may be unstable.
```

In [473]:

```
f = result.get_forecast(1)
f_v = f.predicted_mean
c_i = f.conf_int()
print("Forecast values:")
print(f_v)
print("\nConfidence intervals:")
print(c_i)
```

```
Forecast values:
2268    334217.179433
dtype: float64

Confidence intervals:
       lower IC4WSA  upper IC4WSA
2268  178899.052795  489535.30607
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:836:
ValueWarning: No supported index is available. Prediction results will be given wi
th an integer index beginning at `start`.
  return get_prediction_index(
C:\ProgramData\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.
  return get_prediction_index(
```

In [ ]: