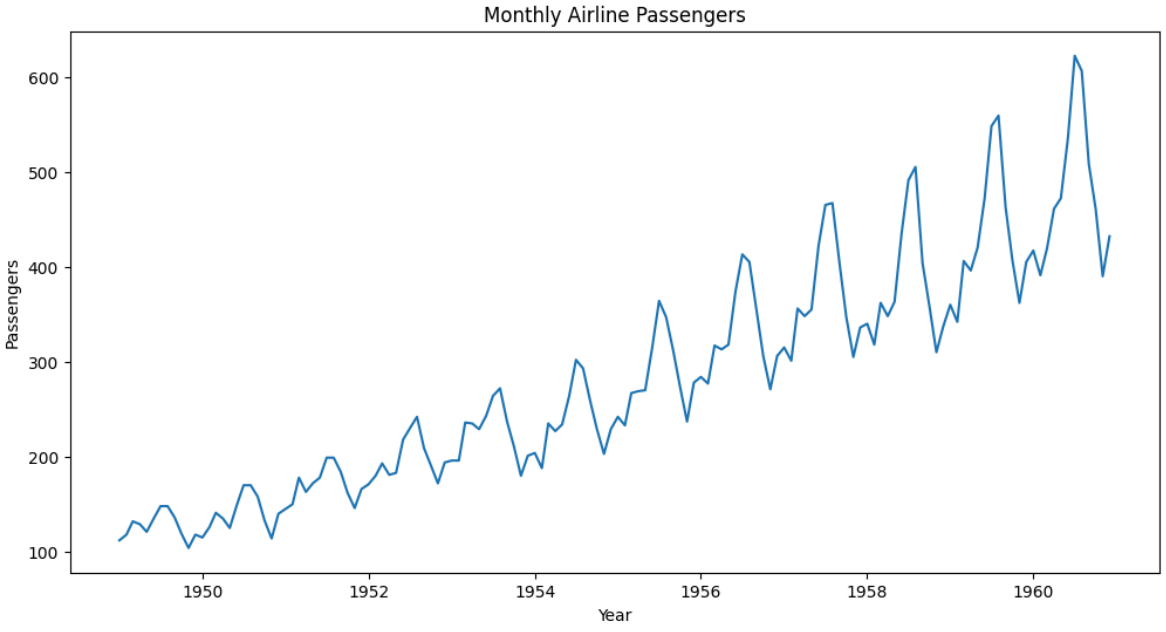
In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_absolute_error, mean_squared_error
```

In [18]:
```
data = pd.read_csv(r"E:\NLP_DATASET\airline_passengers_dataset\airline_passenger
print(data.columns)
```

```
Index(['Month', 'Passengers'], dtype='object')
```

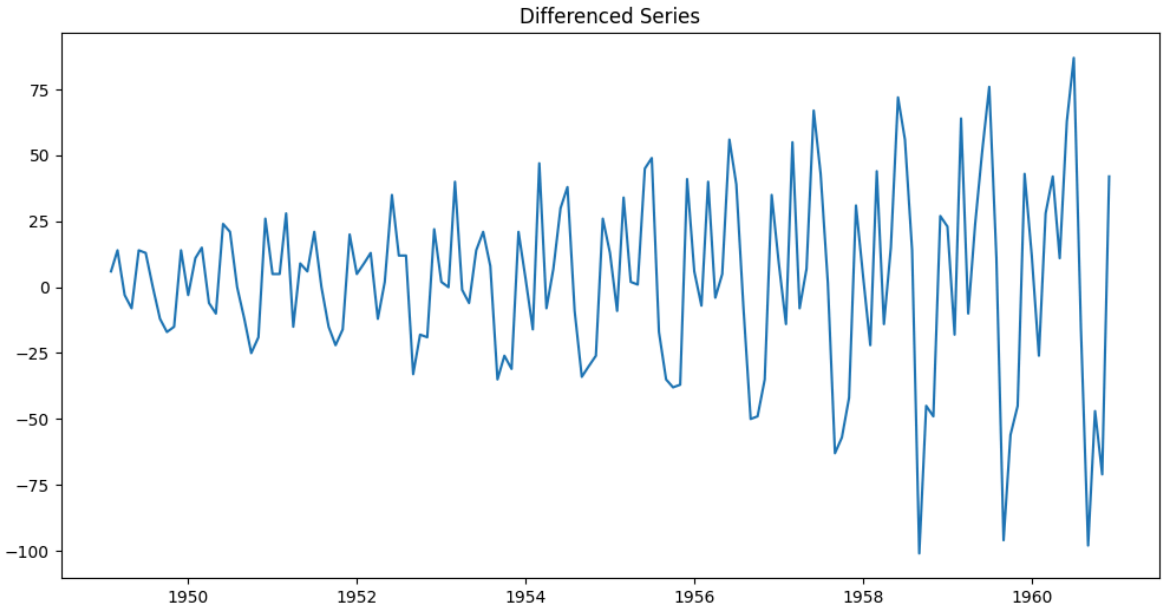
In [19]:
```
data["Month"] = pd.to_datetime(data["Month"])

data = data.set_index("Month")

data = data.asfreq("MS")

plt.figure(figsize=(12,6))
plt.plot(data["Passengers"])
plt.title("Monthly Airline Passengers")
plt.xlabel("Year")
plt.ylabel("Passengers")
plt.show()
```



In [20]:
```
result = adfuller(data["Passengers"])
print("ADF Statistic:", result[0])
print("p-value:", result[1])
```

```
ADF Statistic: 0.8153688792060482
p-value: 0.991880243437641
```

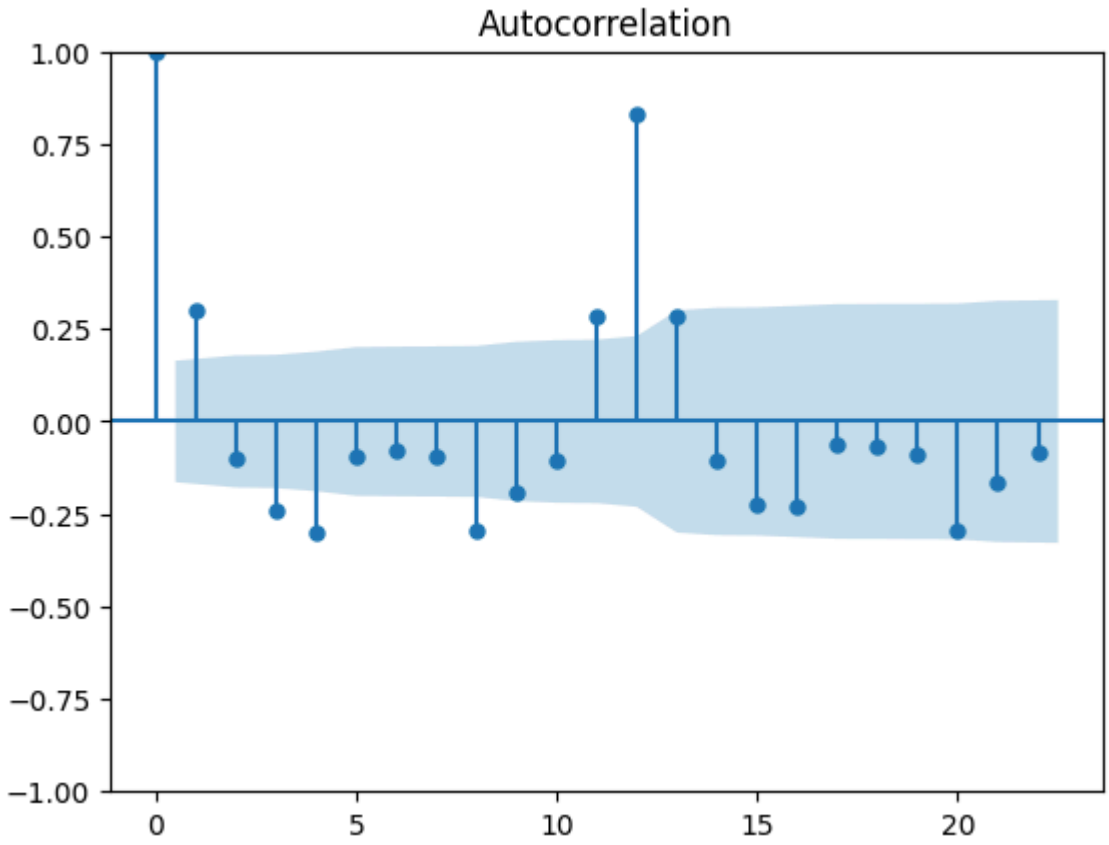
In [21]:
```
diff_data = data["Passengers"].diff().dropna()

plt.figure(figsize=(12,6))
plt.plot(diff_data)
plt.title("Differenced Series")
plt.show()
```



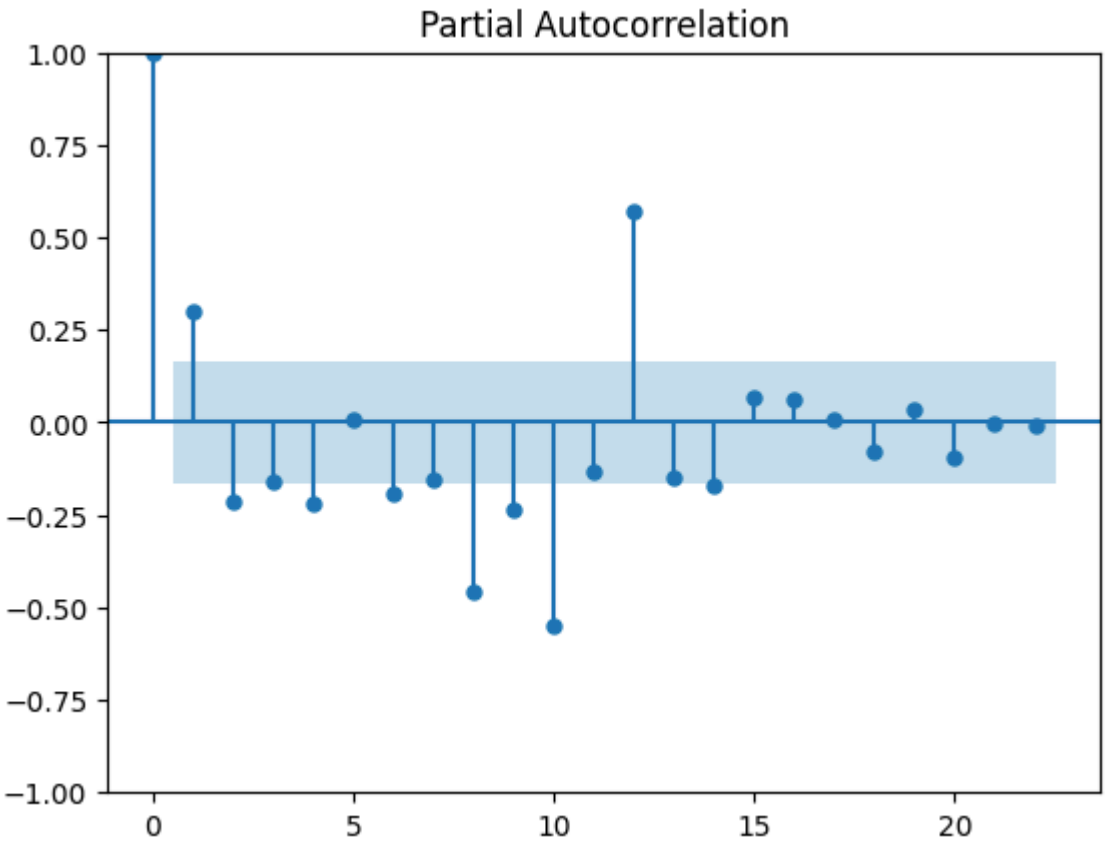
In [22]:
```
result_diff = adfuller(diff_data)
print("ADF Statistic After Differencing:", result_diff[0])
print("p-value After Differencing:", result_diff[1])
```

```
ADF Statistic After Differencing: -2.8292668241699994
p-value After Differencing: 0.0542132902838255
```

In [23]:
```
plot_acf(diff_data)
plt.show()
```



In [24]:
```
plot_pacf(diff_data)
plt.show()
```



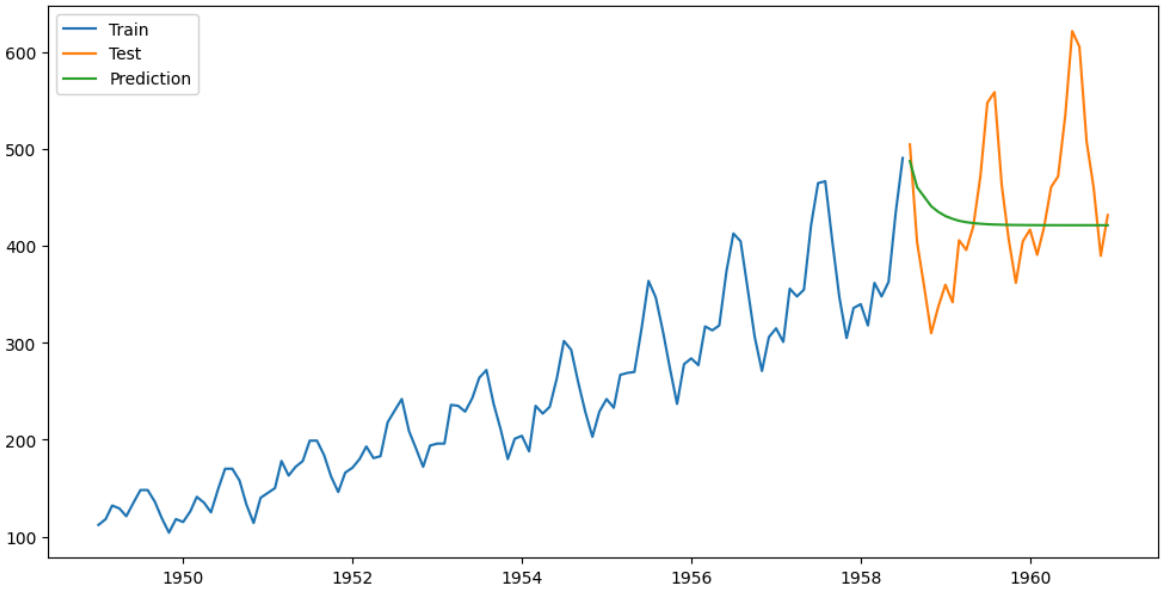
In [26]:
```
train_size = int(len(data) * 0.8)

train = data.iloc[:train_size]
test = data.iloc[train_size:]

model = ARIMA(train["Passengers"], order=(2,1,2))
model_fit = model.fit()
print(model_fit.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:             Passengers   No. Observations:                  115
Model:                 ARIMA(2, 1, 2)   Log Likelihood                -523.758
Date:                Fri, 06 Feb 2026   AIC                           1057.516
Time:                        13:40:25   BIC                           1071.197
Sample:                    01-01-1949   HQIC                          1063.069
                         - 07-01-1958
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.3280      0.145      2.268      0.023       0.045       0.611
ar.L2          0.2521      0.165      1.528      0.126      -0.071       0.575
ma.L1         -0.0125      0.109     -0.114      0.909      -0.227       0.202
ma.L2         -0.7544      0.130     -5.812      0.000      -1.009      -0.500
sigma2       568.4920    103.877      5.473      0.000     364.897     772.087
===================================================================================
==
Ljung-Box (L1) (Q):                   0.02   Jarque-Bera (JB):                 3.
39
Prob(Q):                              0.90   Prob(JB):                         0.
18
Heteroskedasticity (H):               5.24   Skew:                             0.
11
Prob(H) (two-sided):                  0.00   Kurtosis:                         2.
19
===================================================================================
==

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

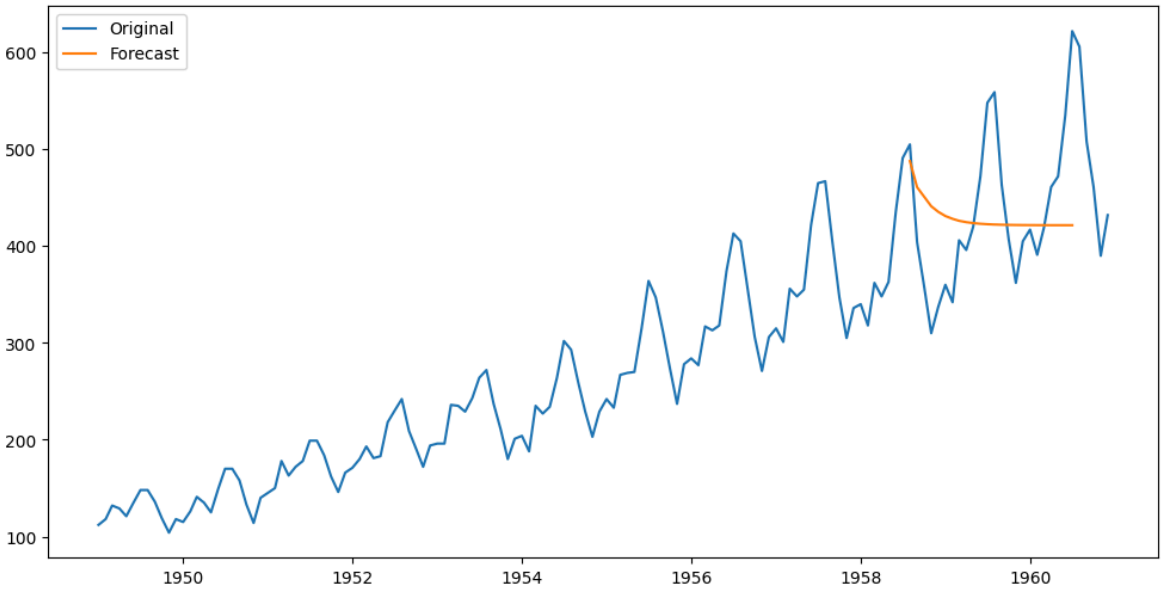
In [31]:
```
import warnings
warnings.filterwarnings("ignore")

start = len(train)
end = len(train) + len(test) - 1

pred = model_fit.predict(start=start, end=end, typ="levels")

mae = mean_absolute_error(test["Passengers"], pred)
rmse = np.sqrt(mean_squared_error(test["Passengers"], pred))

print("MAE:", mae)
print("RMSE:", rmse)
```

```
MAE: 63.54531129227126
RMSE: 82.51301128388961
```

In [32]:
```
plt.figure(figsize=(12,6))
plt.plot(train["Passengers"], label="Train")
plt.plot(test["Passengers"], label="Test")
plt.plot(pred, label="Prediction")
plt.legend()
plt.show()
```



In [29]:
```
future = model_fit.forecast(steps=24)

plt.figure(figsize=(12,6))
plt.plot(data["Passengers"], label="Original")
plt.plot(future, label="Forecast")
plt.legend()
plt.show()
```



In [30]:
```
print("Next 24 Months Forecast:")
print(future)
```

```
Next 24 Months Forecast:
1958-08-01    487.825560
1958-09-01    460.796800
1958-10-01    451.130922
1958-11-01    441.145635
1958-12-01    435.433346
1959-01-01    431.042077
1959-02-01    428.161471
1959-03-01    426.109440
1959-04-01    424.710071
1959-05-01    423.733689
1959-06-01    423.060606
1959-07-01    422.593654
1959-08-01    422.270786
1959-09-01    422.047151
1959-10-01    421.892392
1959-11-01    421.785245
1959-12-01    421.711080
1960-01-01    421.659739
1960-02-01    421.624199
1960-03-01    421.599597
1960-04-01    421.582567
1960-05-01    421.570778
1960-06-01    421.562618
1960-07-01    421.556968
Freq: MS, Name: predicted_mean, dtype: float64
```

In [ ]: