In [1]:
```python
import yfinance as yf
import pandas as pd
import matplotlib.pyplot as plt
import warnings

warnings.filterwarnings('ignore')
stock_data = yf.download('AAPL', start='2025-01-01')
stock_data.head(10)
```

```
[*********************100%***********************]  1 of 1 completed
```

Out[1]:

Price	Close	High	Low	Open	Volume
Ticker	AAPL	AAPL	AAPL	AAPL	AAPL
Date					
2025-01-02	243.263199	248.500565	241.238085	248.330961	55740700
2025-01-03	242.774368	243.592387	241.307905	242.774368	40244100
2025-01-06	244.410416	246.734810	242.614744	243.722074	45045600
2025-01-07	241.627136	244.959095	240.769205	242.395272	40856000
2025-01-08	242.115952	243.123531	239.472335	241.337830	37628900
2025-01-10	236.280045	239.582077	232.439303	239.432429	61710900
2025-01-13	233.835922	234.105277	229.167192	232.968021	49630700
2025-01-14	232.718613	235.551775	231.910564	234.185076	39435300
2025-01-15	237.297562	238.384950	233.865838	234.075339	39832000
2025-01-16	227.710693	237.437230	227.481251	236.778830	71759100

In [2]:
```python
# Reindex to include all calendar days
stock_data = stock_data.asfreq('D')
```

In [3]:
```python
# Optionally fill missing values
stock_data = stock_data.fillna(method='ffill')  # Forward-fill missing data
```

In [4]:
```python
stock_data.tail(10)
```

Out[4]:

Price	Close	High	Low	Open	Volume
Ticker	AAPL	AAPL	AAPL	AAPL	AAPL
Date					
2025-07-20	211.179993	211.789993	209.699997	210.869995	48974600.0
2025-07-21	212.479996	215.779999	211.630005	212.100006	51377400.0
2025-07-22	214.399994	214.949997	212.229996	213.139999	46404100.0
2025-07-23	214.149994	215.149994	212.410004	215.000000	46989300.0
2025-07-24	213.759995	215.690002	213.529999	213.899994	46022600.0
2025-07-25	213.880005	215.240005	213.399994	214.699997	40268800.0
2025-07-26	213.880005	215.240005	213.399994	214.699997	40268800.0
2025-07-27	213.880005	215.240005	213.399994	214.699997	40268800.0
2025-07-28	214.050003	214.850006	213.059998	214.029999	37858000.0
2025-07-29	211.270004	214.809998	210.820007	214.179993	49943600.0

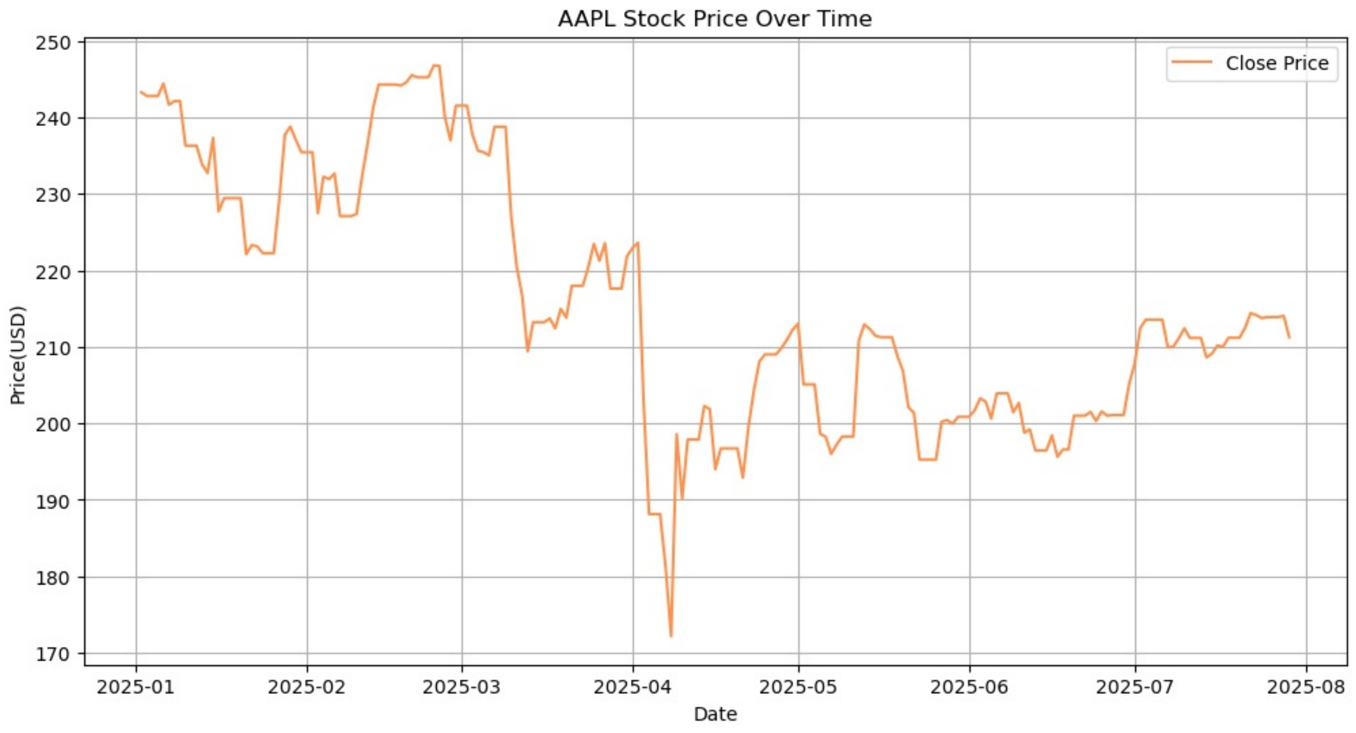
In [5]:
```python
stock_data= stock_data[['Close']]
stock_data
```

Out[5]:

Price	Close
Ticker	AAPL
Date	
2025-01-02	243.263199
2025-01-03	242.774368
2025-01-04	242.774368
2025-01-05	242.774368
2025-01-06	244.410416
...	...
2025-07-25	213.880005
2025-07-26	213.880005
2025-07-27	213.880005
2025-07-28	214.050003
2025-07-29	211.270004

209 rows × 1 columns

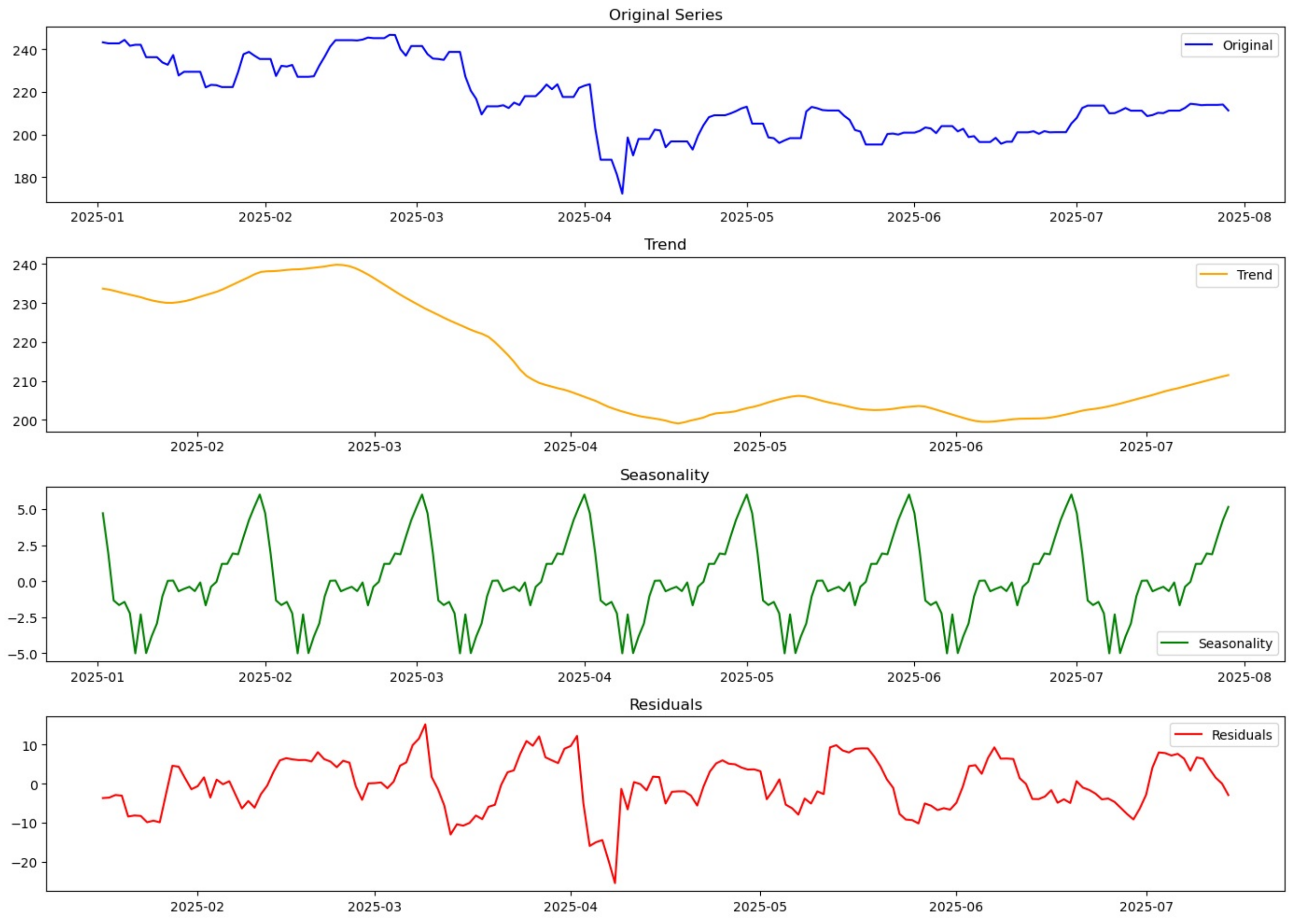
In [6]:
```
plt.figure(figsize=(12, 6))
plt.plot(stock_data.index, stock_data['Close'], label='Close Price',color='#FF914D')
plt.title('AAPL Stock Price Over Time')
plt.xlabel('Date')
plt.ylabel('Price(USD)')
plt.legend()
plt.grid(True)
plt.show()
```



In [7]:
```
from statsmodels.tsa.seasonal import seasonal_decompose

# Perform additive decomposition
decomposition = seasonal_decompose(stock_data['Close'], model='additive', period=30)

# Extract components
trend = decomposition.trend
seasonal = decomposition.seasonal
residuals = decomposition.resid
```

In [8]:
```
# Plot all components
plt.figure(figsize=(14, 10))

# Original Series
plt.subplot(4, 1, 1)
plt.plot(stock_data['Close'], label='Original', color='blue')
plt.title('Original Series')
plt.legend()
```

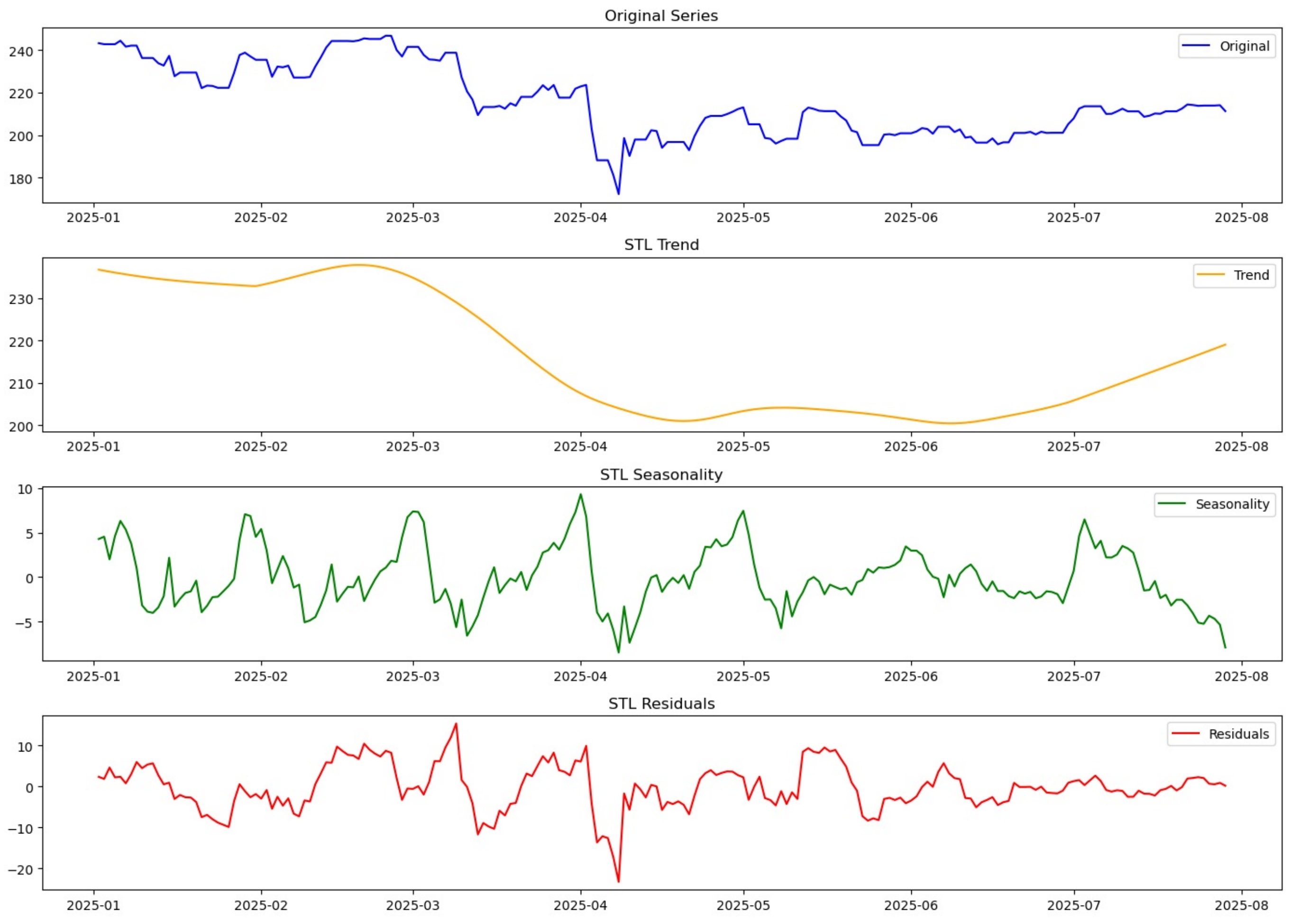
```
# Trend
plt.subplot(4, 1, 2)
plt.plot(trend, label='Trend', color='orange')
plt.title('Trend')
plt.legend()

# Seasonality
plt.subplot(4, 1, 3)
plt.plot(seasonal, label='Seasonality', color='green')
plt.title('Seasonality')
plt.legend()

# Residuals
plt.subplot(4, 1, 4)
plt.plot(residuals, label='Residuals', color='red')
plt.title('Residuals')
plt.legend()

plt.tight_layout()
plt.show()
```



In [9]:

```
from statsmodels.tsa.seasonal import STL

# STL decomposition (period=30 assumes monthly seasonality for daily data)
stl = STL(stock_data['Close'], period=30)
result = stl.fit()

# Extract components
trend = result.trend
seasonal = result.seasonal
residual = result.resid

# Plot all components
plt.figure(figsize=(14, 10))

# Original Series
plt.subplot(4, 1, 1)
plt.plot(stock_data['Close'], label='Original', color='blue')
plt.title('Original Series')
plt.legend()

# Trend
plt.subplot(4, 1, 2)
plt.plot(trend, label='Trend', color='orange')
```

```python
plt.title('STL Trend')
plt.legend()

# Seasonality
plt.subplot(4, 1, 3)
plt.plot(seasonal, label='Seasonality', color='green')
plt.title('STL Seasonality')
plt.legend()

# Residuals
plt.subplot(4, 1, 4)
plt.plot(residual, label='Residuals', color='red')
plt.title('STL Residuals')
plt.legend()

plt.tight_layout()
plt.show()
```



In [10]:
```python
from statsmodels.tsa.stattools import adfuller

# Perform ADF Test
result = adfuller(stock_data['Close'].dropna())  # dropna is important

# Print results
print("ADF Statistic:", result[0])
print("p-value:", result[1])
print("Critical Values:")
for key, value in result[4].items():
    print(f"   {key}: {value}")

# Interpretation
if result[1] < 0.05:
    print("✅ The series is stationary (reject H0)")
else:
    print("❌ The series is non-stationary (fail to reject H0)")
```

```
ADF Statistic: -2.154983572242222
p-value: 0.22298737804418461
Critical Values:
   1%: -3.4621857592784546
   5%: -2.875537986778846
   10%: -2.574231080806213
❌ The series is non-stationary (fail to reject H0)
```

In [11]:
```python
def adf_test(series):
    result=adfuller(series)
    print('ADf Statistic:',result[0])
    print('p-value:', result[1])
    for key, value in result[4].items():
        print('Critical Value(%s): %.3f' % (key, value))
```

In [12]:
```
prices = stock_data['Close']
```

In [13]:
```
adf_test(prices)
```

```
ADf Statistic: -2.154983572242222
p-value: 0.22298737804418461
Critical Value(1%): -3.462
Critical Value(5%): -2.876
Critical Value(10%): -2.574
```

In [14]:
```
prices
```

Out[14]:

Ticker	AAPL
Date	
2025-01-02	243.263199
2025-01-03	242.774368
2025-01-04	242.774368
2025-01-05	242.774368
2025-01-06	244.410416
...	...
2025-07-25	213.880005
2025-07-26	213.880005
2025-07-27	213.880005
2025-07-28	214.050003
2025-07-29	211.270004

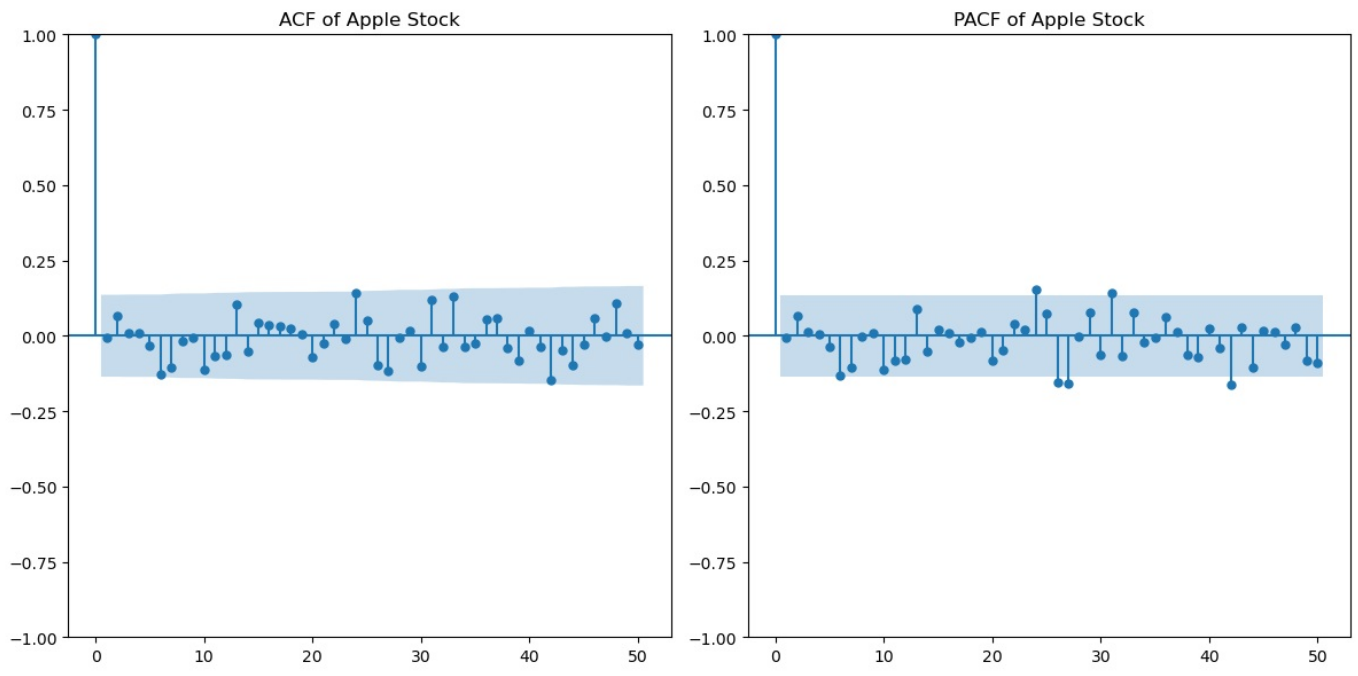
209 rows × 1 columns

In [15]:
```
#prices.diff().dropna()
```

In [16]:
```
##adf_test(prices.diff().dropna())
```

In [17]:
```
 ##9.253517446728197e-27<0.05
```

In [18]:
```
from statsmodels.graphics.tsaplots import plot_acf,plot_pacf
plt.figure(figsize=(12, 6))

# ACF plot
plt.subplot(1,2,1)
plot_acf(stock_data['Close'].diff().dropna(),ax=plt.gca(),lags=50)
plt.title('ACF of Apple Stock')

#PACF plot
plt.subplot(1,2,2)
plot_pacf(stock_data['Close'].diff().dropna(),ax=plt.gca(),lags=50,method='ywm')
plt.title('PACF of Apple Stock')

plt.tight_layout()
plt.show()
```



In [19]:
```
!pip install pmdarima
```

```
Requirement already satisfied: pmdarima in c:\programdata\anaconda3\lib\site-packages (2.0.4)
Requirement already satisfied: joblib>=0.11 in c:\programdata\anaconda3\lib\site-packages (from pmdarima) (1.4.2
)
Requirement already satisfied: Cython!=0.29.18,!=0.29.31,>=0.29 in c:\programdata\anaconda3\lib\site-packages (f
rom pmdarima) (3.1.2)
Requirement already satisfied: numpy>=1.21.2 in c:\programdata\anaconda3\lib\site-packages (from pmdarima) (1.26
.4)
Requirement already satisfied: pandas>=0.19 in c:\programdata\anaconda3\lib\site-packages (from pmdarima) (2.2.2
)
Requirement already satisfied: scikit-learn>=0.22 in c:\programdata\anaconda3\lib\site-packages (from pmdarima)
(1.5.1)
Requirement already satisfied: scipy>=1.3.2 in c:\programdata\anaconda3\lib\site-packages (from pmdarima) (1.13.
1)
Requirement already satisfied: statsmodels>=0.13.2 in c:\programdata\anaconda3\lib\site-packages (from pmdarima)
(0.14.2)
Requirement already satisfied: urllib3 in c:\programdata\anaconda3\lib\site-packages (from pmdarima) (2.2.3)
Requirement already satisfied: setuptools!=50.0.0,>=38.6.0 in c:\programdata\anaconda3\lib\site-packages (from p
mdarima) (75.1.0)
Requirement already satisfied: packaging>=17.1 in c:\programdata\anaconda3\lib\site-packages (from pmdarima) (24
.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\programdata\anaconda3\lib\site-packages (from pandas
>=0.19->pmdarima) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.19->pm
darima) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.19->
pmdarima) (2023.3)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\programdata\anaconda3\lib\site-packages (from scikit-l
earn>=0.22->pmdarima) (3.5.0)
Requirement already satisfied: patsy>=0.5.6 in c:\programdata\anaconda3\lib\site-packages (from statsmodels>=0.1
3.2->pmdarima) (0.5.6)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-packages (from patsy>=0.5.6->statsmodels
>=0.13.2->pmdarima) (1.16.0)
```

In [20]:
```
import pmdarima as pm

model = pm.auto_arima(
    stock_data['Close'],
    seasonal=False,
    stepwise=True,
    suppress_warnings=True,
    trace=True
)
```

```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=1175.427, Time=0.45 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=1172.363, Time=0.01 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=1174.354, Time=0.02 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=1174.355, Time=0.02 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=1170.669, Time=0.01 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=1176.140, Time=0.08 sec

Best model:  ARIMA(0,1,0)(0,0,0)[0]
Total fit time: 0.605 seconds
```

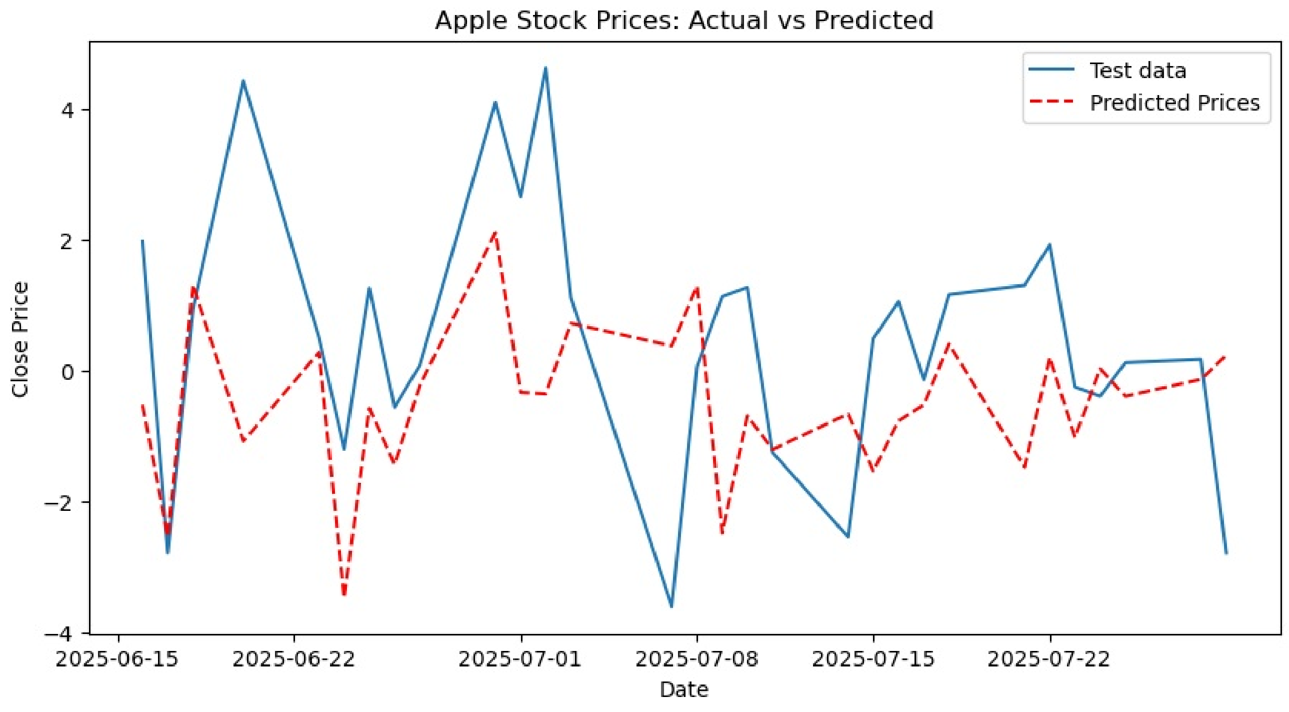
AR MODEL (AUTO REGRESSIVE)

In [21]:
```
stock_data =yf.download('AAPL',start='2025-01-01')
apple_price_stationary=stock_data['Close'].diff().dropna()

train_data,test_data = apple_price_stationary[:-30],apple_price_stationary[-30:]
```

```
[*********************100%***********************]  1 of 1 completed
```

In [22]:
```
from statsmodels.tsa.ar_model import AutoReg
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
import numpy as np

# Fit the AR model to the training data
model = AutoReg(train_data, lags=30)
model_fit = model.fit()

# Make predictions on the test data
predictions = model_fit.predict(
    start=len(train_data),
    end=len(train_data) + len(test_data) - 1,
    dynamic=False
)

# Plot the actual vs predicted values
plt.figure(figsize=(10, 5))
plt.plot(test_data.index, test_data, label='Test data')
plt.plot(test_data.index, predictions, color='red', linestyle='--', label='Predicted Prices')
plt.title('Apple Stock Prices: Actual vs Predicted')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.legend()
plt.show()

# Evaluating model using RMSE
rmse = np.sqrt(mean_squared_error(test_data, predictions))
print(f'RMSE: {rmse:.1f}')
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:836: ValueWarning: No supported ind
ex is available. Prediction results will be given with an integer index beginning at `start`.
  return get_prediction_index(
```



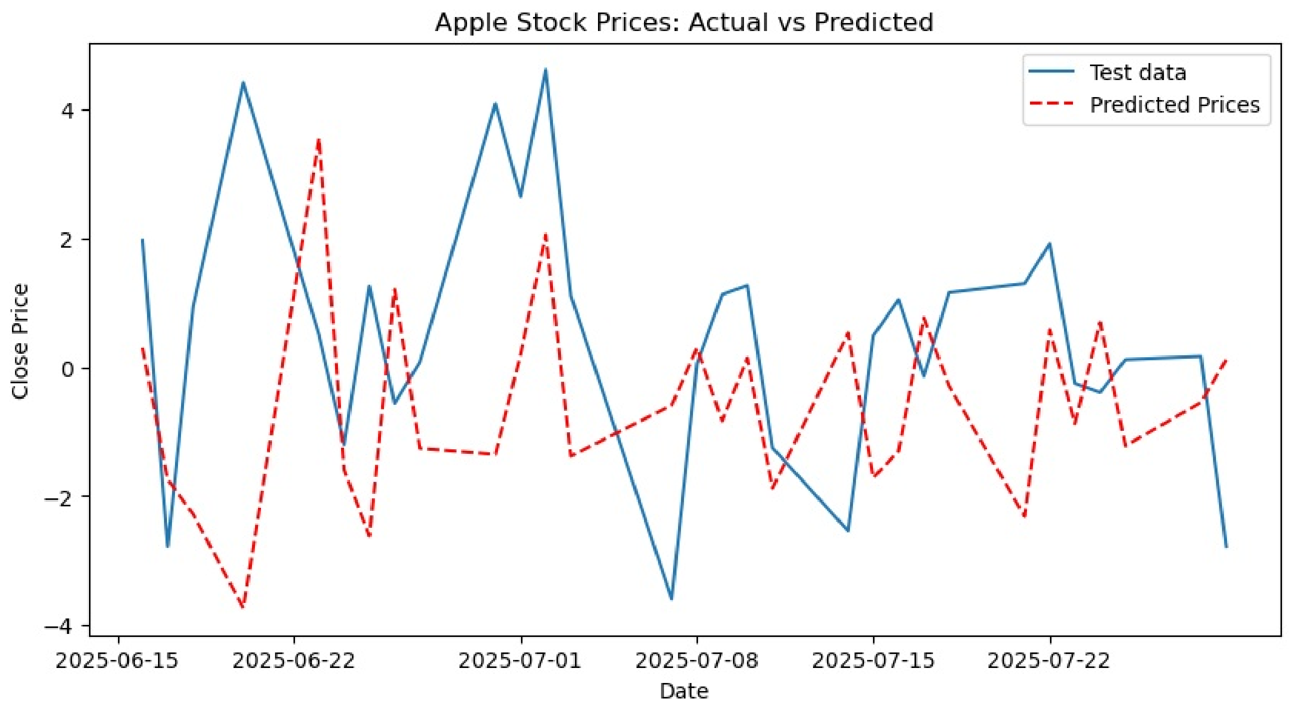
```
RMSE: 2.2
```

MA MODEL(MOVING AVERAGE MODEL)

In [34]:
```
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
import numpy as np
```

```
# Fit the ARIMA model to the training data
model = ARIMA(train_data, order=(0, 0, 30))
model_fit = model.fit()

# Make predictions on the test data
predictions = model_fit.predict(
    start=len(train_data),
    end=len(train_data) + len(test_data) - 1,
    dynamic=False
)

# Plot the actual vs predicted values
plt.figure(figsize=(10, 5))
plt.plot(test_data.index, test_data, label='Test data')
plt.plot(test_data.index, predictions, color='red', linestyle='--', label='Predicted Prices')
plt.title('Apple Stock Prices: Actual vs Predicted')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.legend()
plt.show()

# Evaluating model using RMSE
rmse = np.sqrt(mean_squared_error(test_data, predictions))
print(f'RMSE: {rmse:.1f}')
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\base\model.py:607: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:836: ValueWarning: No supported ind
ex is available. Prediction results will be given with an integer index beginning at `start`.
  return get_prediction_index(
```



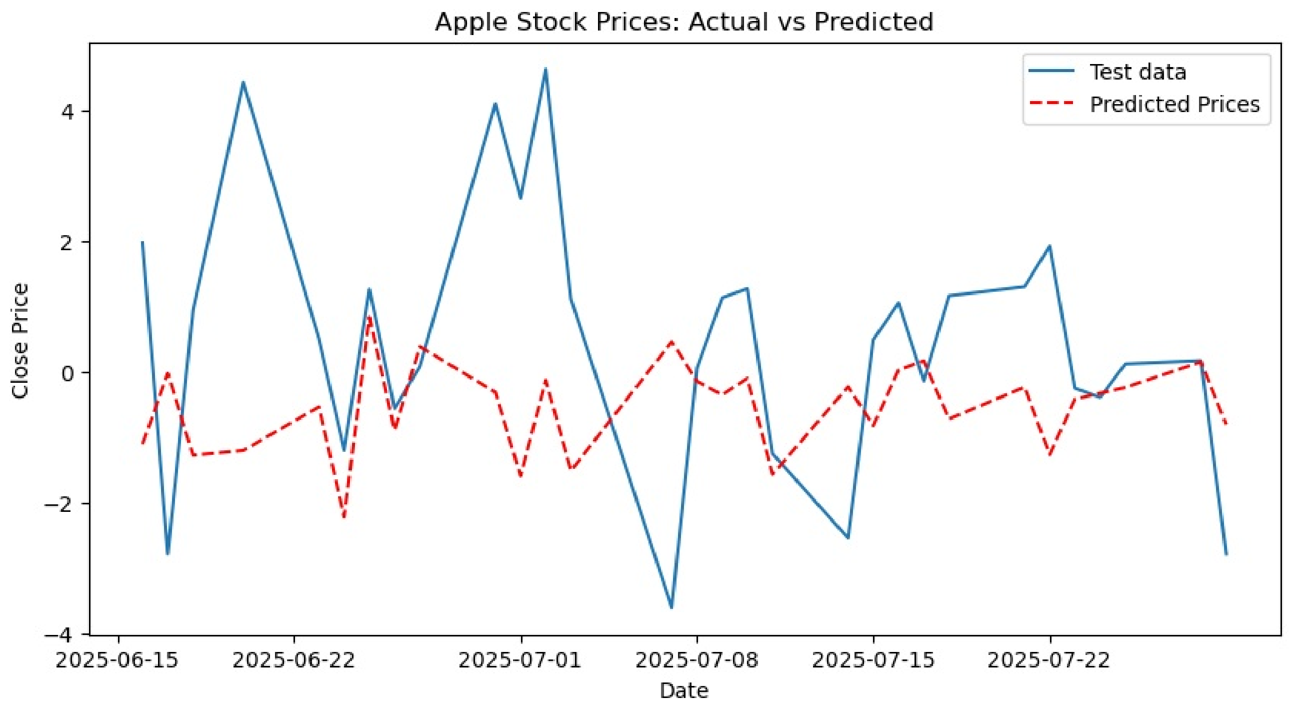
```
RMSE: 2.7
```

ARMA MODEL(AUTO REGRESSIVE MOVING AVERAGE MODEL)

In [24]:
```
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
import numpy as np

# Fit the ARIMA model to the training data
model = ARIMA(train_data, order=(5, 0, 7 ))
model_fit = model.fit()
```

```
# Make predictions on the test data
predictions = model_fit.predict(
    start=len(train_data),
    end=len(train_data) + len(test_data) - 1,
    dynamic=False
)

# Plot the actual vs predicted values
plt.figure(figsize=(10, 5))
plt.plot(test_data.index, test_data, label='Test data')
plt.plot(test_data.index, predictions, color='red', linestyle='--', label='Predicted Prices')
plt.title('Apple Stock Prices: Actual vs Predicted')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.legend()
plt.show()

# Evaluating model using RMSE
rmse = np.sqrt(mean_squared_error(test_data, predictions))
print(f'RMSE: {rmse:.1f}')
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\base\model.py:607: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:836: ValueWarning: No supported ind
ex is available. Prediction results will be given with an integer index beginning at `start`.
  return get_prediction_index(
```

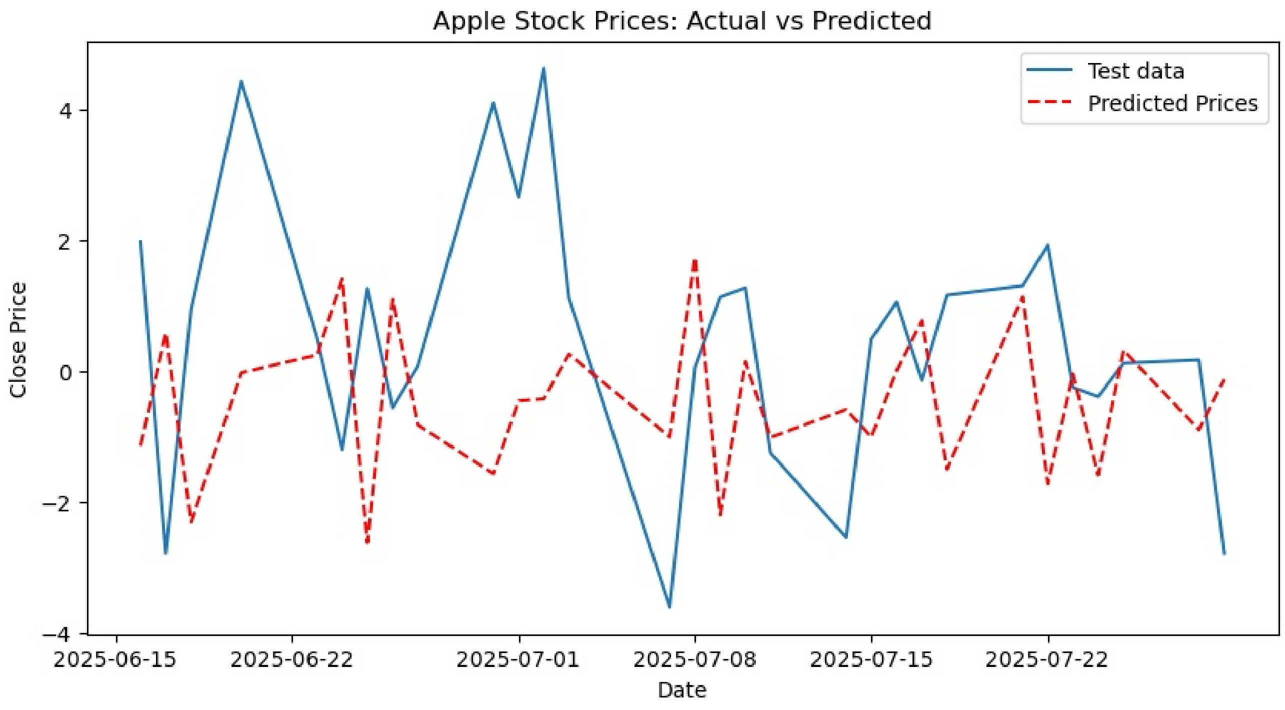


```
RMSE: 2.4
```

ARIMA MODEL

In [31]:
```
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
import numpy as np

# Fit the ARIMA model to the training data
model = ARIMA(train_data, order=(8,1,9))
model_fit = model.fit()

# Make predictions on the test data
predictions = model_fit.predict(
    start=len(train_data),
    end=len(train_data) + len(test_data) - 1,
    dynamic=False
)

# Plot the actual vs predicted values
plt.figure(figsize=(10, 5))
plt.plot(test_data.index, test_data, label='Test data')
plt.plot(test_data.index, predictions, color='red', linestyle='--', label='Predicted Prices')
plt.title('Apple Stock Prices: Actual vs Predicted')
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.legend()
plt.show()

# Evaluating model using RMSE
rmse = np.sqrt(mean_squared_error(test_data, predictions))
print(f'RMSE: {rmse:.1f}')
```

```
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\base\model.py:607: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
C:\ProgramData\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:836: ValueWarning: No supported ind
ex is available. Prediction results will be given with an integer index beginning at `start`.
  return get_prediction_index(
```



```
RMSE: 2.6
```

In [37]:
```
!pip install tensorflow
```

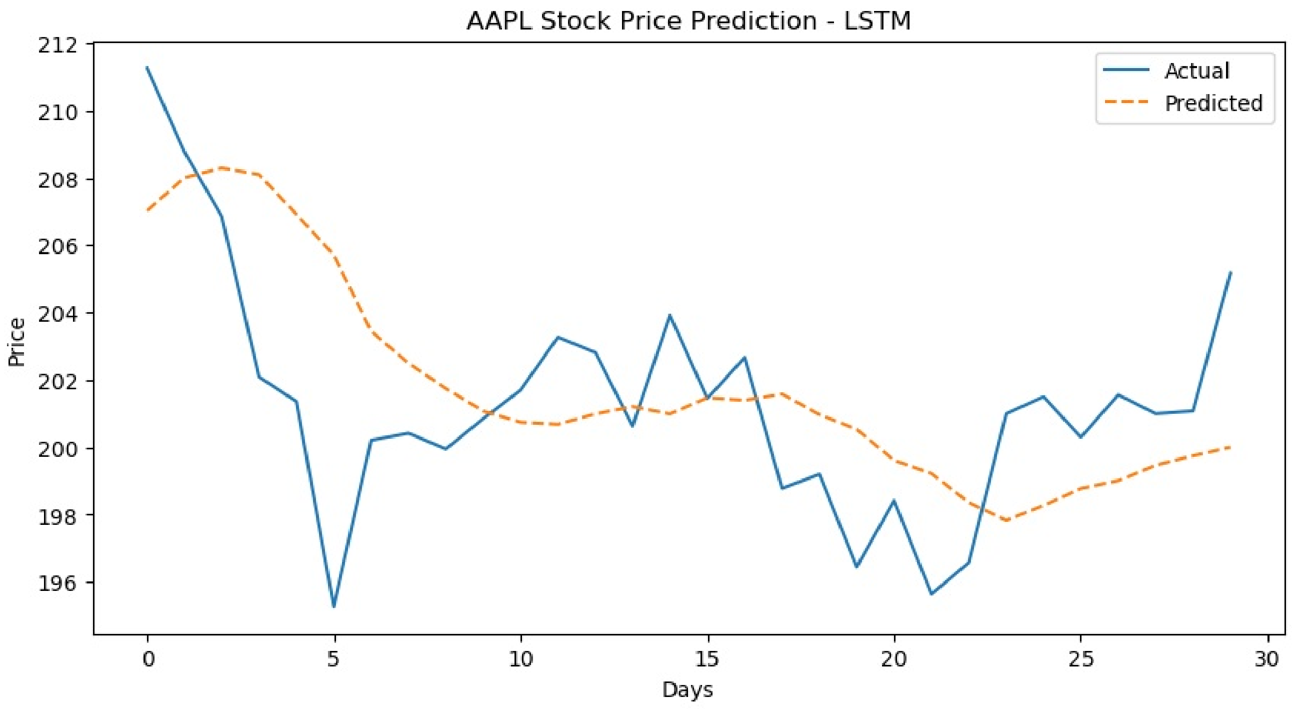
```
Requirement already satisfied: tensorflow in c:\programdata\anaconda3\lib\site-packages (2.19.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\programdata\anaconda3\lib\site-packages (from tensorflow) (2
.3.1)
Requirement already satisfied: astunparse>=1.6.0 in c:\programdata\anaconda3\lib\site-packages (from tensorflow)
(1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in c:\programdata\anaconda3\lib\site-packages (from tensorfl
ow) (25.2.10)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in c:\programdata\anaconda3\lib\site-packages
(from tensorflow) (0.6.0)
Requirement already satisfied: google-pasta>=0.1.1 in c:\programdata\anaconda3\lib\site-packages (from tensorflo
w) (0.2.0)
Requirement already satisfied: libclang>=13.0.0 in c:\programdata\anaconda3\lib\site-packages (from tensorflow)
(18.1.1)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\programdata\anaconda3\lib\site-packages (from tensorflow)
(3.4.0)
Requirement already satisfied: packaging in c:\programdata\anaconda3\lib\site-packages (from tensorflow) (24.1)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.3
in c:\programdata\anaconda3\lib\site-packages (from tensorflow) (4.25.3)
Requirement already satisfied: requests<3,>=2.21.0 in c:\programdata\anaconda3\lib\site-packages (from tensorflo
w) (2.32.3)
Requirement already satisfied: setuptools in c:\programdata\anaconda3\lib\site-packages (from tensorflow) (75.1.
0)
Requirement already satisfied: six>=1.12.0 in c:\programdata\anaconda3\lib\site-packages (from tensorflow) (1.16
.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\programdata\anaconda3\lib\site-packages (from tensorflow)
(3.1.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\programdata\anaconda3\lib\site-packages (from tens
orflow) (4.11.0)
Requirement already satisfied: wrapt>=1.11.0 in c:\programdata\anaconda3\lib\site-packages (from tensorflow) (1.
14.1)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\programdata\anaconda3\lib\site-packages (from tensorflo
w) (1.74.0)
Requirement already satisfied: tensorboard~=2.19.0 in c:\programdata\anaconda3\lib\site-packages (from tensorflo
w) (2.19.0)
Requirement already satisfied: keras>=3.5.0 in c:\programdata\anaconda3\lib\site-packages (from tensorflow) (3.1
1.0)
Requirement already satisfied: numpy<2.2.0,>=1.26.0 in c:\programdata\anaconda3\lib\site-packages (from tensorfl
ow) (1.26.4)
Requirement already satisfied: h5py>=3.11.0 in c:\programdata\anaconda3\lib\site-packages (from tensorflow) (3.1
1.0)
Requirement already satisfied: ml-dtypes<1.0.0,>=0.5.1 in c:\programdata\anaconda3\lib\site-packages (from tenso
rflow) (0.5.3)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\programdata\anaconda3\lib\site-packages (from astunparse
>=1.6.0->tensorflow) (0.44.0)
Requirement already satisfied: rich in c:\programdata\anaconda3\lib\site-packages (from keras>=3.5.0->tensorflow
) (13.7.1)
Requirement already satisfied: namex in c:\programdata\anaconda3\lib\site-packages (from keras>=3.5.0->tensorflo
w) (0.1.0)
Requirement already satisfied: optree in c:\programdata\anaconda3\lib\site-packages (from keras>=3.5.0->tensorfl
ow) (0.17.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\programdata\anaconda3\lib\site-packages (from requ
ests<3,>=2.21.0->tensorflow) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\lib\site-packages (from requests<3,>=2.2
1.0->tensorflow) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\programdata\anaconda3\lib\site-packages (from requests<3
,>=2.21.0->tensorflow) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-packages (from requests<3
,>=2.21.0->tensorflow) (2025.6.15)
Requirement already satisfied: markdown>=2.6.8 in c:\programdata\anaconda3\lib\site-packages (from tensorboard~=
2.19.0->tensorflow) (3.4.1)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in c:\programdata\anaconda3\lib\site-packag
es (from tensorboard~=2.19.0->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in c:\programdata\anaconda3\lib\site-packages (from tensorboard~=
2.19.0->tensorflow) (3.0.3)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\programdata\anaconda3\lib\site-packages (from werkzeug>=1
.0.1->tensorboard~=2.19.0->tensorflow) (2.1.3)
Requirement already satisfied: markdown-it-py>=2.2.0 in c:\programdata\anaconda3\lib\site-packages (from rich->k
eras>=3.5.0->tensorflow) (2.2.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in c:\programdata\anaconda3\lib\site-packages (from rich-
>keras>=3.5.0->tensorflow) (2.15.1)
Requirement already satisfied: mdurl~=0.1 in c:\programdata\anaconda3\lib\site-packages (from markdown-it-py>=2.
2.0->rich->keras>=3.5.0->tensorflow) (0.1.0)
```

In [43]:

```python
import numpy as np
import pandas as pd
import yfinance as yf
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import LSTM, Dense
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt

# 1. Load data
```

```python
df = yf.download('AAPL', start='2023-01-01', end='2025-07-01')
data = df[['Close']].dropna()

# 2. Scale data
scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(data)

# 3. Create sequences (60 days to predict next day)
def create_sequences(data, lookback=60):
    X, y = [], []
    for i in range(lookback, len(data)):
        X.append(data[i-lookback:i])
        y.append(data[i])
    return np.array(X), np.array(y)

lookback = 60
X, y = create_sequences(scaled_data, lookback)

# 4. Train/test split (last 30 days as test)
X_train, X_test = X[:-30], X[-30:]
y_train, y_test = y[:-30], y[-30:]

# 5. LSTM model
model = Sequential()
model.add(LSTM(50, activation='relu', return_sequences=False, input_shape=(lookback, 1)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
model.fit(X_train, y_train, epochs=20, batch_size=16, verbose=0)

# 6. Predict
y_pred = model.predict(X_test)

# 7. Inverse scale
y_test_inv = scaler.inverse_transform(y_test)
y_pred_inv = scaler.inverse_transform(y_pred)

# 8. RMSE
rmse = np.sqrt(mean_squared_error(y_test_inv, y_pred_inv))
print(f'✅ RMSE: {rmse:.2f}')

# 9. Plot
plt.figure(figsize=(10,5))
plt.plot(y_test_inv, label='Actual')
plt.plot(y_pred_inv, label='Predicted', linestyle='--')
plt.title('AAPL Stock Price Prediction - LSTM')
plt.xlabel('Days')
plt.ylabel('Price')
plt.legend()
plt.show()
```

```
[*********************100%***********************]  1 of 1 completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 156ms/step
✅ RMSE: 3.37
```



In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

```
[illegible]
```