EX:No.8 DATE:12/04/25	**Create an ARIMA model for time series forecasting.**

AIM:

To build and evaluate an ARIMA model for time series forecasting on stock market data.

ALGORITHM:

- Import the necessary libraries and load the time series dataset (e.g., `AAPL.csv`).
- Convert the 'Date' column to datetime format and set it as the index.
- Visualize the data and check for stationarity using ADF (Augmented Dickey-Fuller) test.
- Apply differencing if the data is non-stationary.
- Determine the optimal (p, d, q) parameters using ACF, PACF, or auto_arima.
- Fit the ARIMA model on the training data using the selected parameters.
- Forecast future values and evaluate the model using metrics like MAE, MSE, and RMSE.

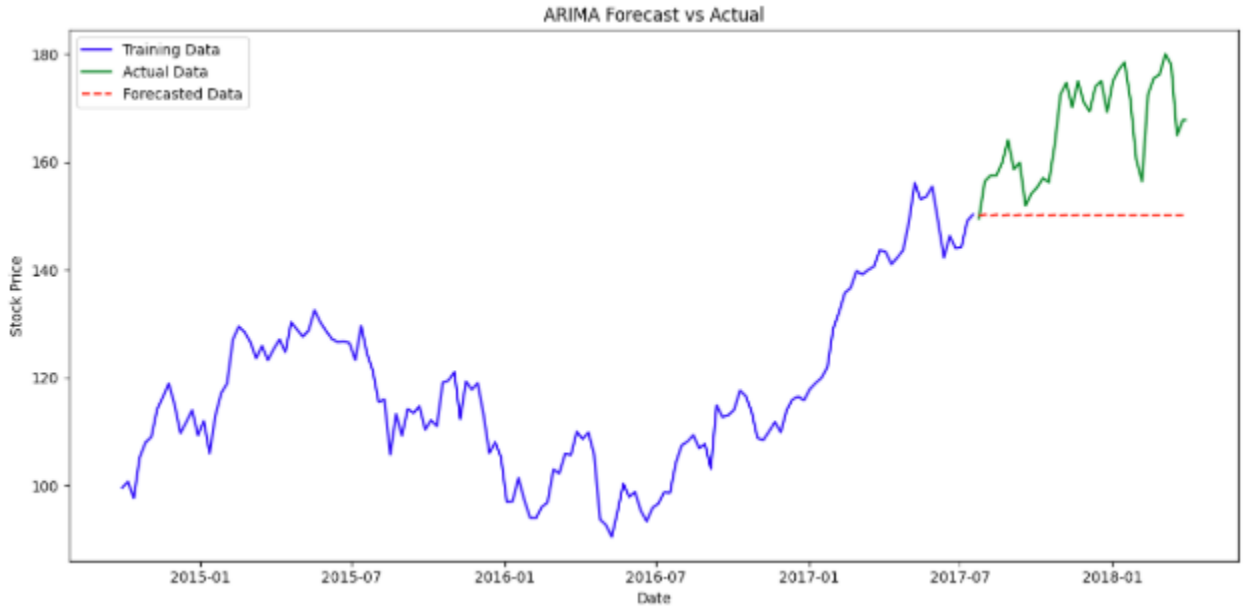
CODE:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
import pmdarima as pm


# Load dataset
data = pd.read_csv('AAPL.csv')
data['Date'] = pd.to_datetime(data['Date'])
data.set_index('Date', inplace=True)


# Use 'Close' column
ts = data['Close']
```

```
# Train-test split
train_size = int(len(ts) * 0.8)
train, test = ts[:train_size], ts[train_size:]

# Fit Auto ARIMA on training data
auto_model = pm.auto_arima(train, seasonal=False, stepwise=True, trace=True,
                           suppress_warnings=True, error_action="ignore")

# Summary of model
print(auto_model.summary())

# Forecast for test period
forecast = auto_model.predict(n_periods=len(test))

# Evaluate the model
mse = mean_squared_error(test, forecast)
print(f"\nMean Squared Error: {mse:.4f}")

# Plot results
plt.figure(figsize=(12, 6))
plt.plot(train.index, train, label='Training Data', color='blue')
plt.plot(test.index, test, label='Actual Data', color='green')
plt.plot(test.index, forecast, label='Forecasted Data', color='red', linestyle='--')
plt.title('ARIMA Forecast vs Actual')
plt.xlabel('Date')
plt.ylabel('Stock Price')
```

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

OUTPUT:



RESULT:

Thus the program has been completed and verified successfully.