```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense

# Load your stock price data into a Pandas DataFrame
# Replace 'your_data.csv' with your actual data file
data = pd.read_csv('your_data.csv')

# Use only the 'Close' prices for prediction
data = data[['Close']]

# Normalize the data
scaler = MinMaxScaler()
data = scaler.fit_transform(data)

# Split the data into training and testing sets
train_size = int(len(data) * 0.80)
train_data, test_data = data[0:train_size], data[train_size:]

# Create sequences for training and testing
def create_sequences(data, seq_length):
    sequences = []
    for i in range(len(data) - seq_length):
        sequences.append(data[i:i+seq_length])
    return np.array(sequences)

seq_length = 10  # You can adjust this parameter
X_train = create_sequences(train_data, seq_length)
y_train = train_data[seq_length:]
X_test = create_sequences(test_data, seq_length)
y_test = test_data[seq_length:]

# Build the LSTM model
model = Sequential()
model.add(LSTM(50, activation='relu', input_shape=(seq_length, 1)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mean_squared_error')

# Train the model
model.fit(X_train, y_train, epochs=100, batch_size=64)

# Make predictions
predicted = model.predict(X_test)

# Inverse transform the predictions to get actual stock prices
```

```
predicted = scaler.inverse_transform(predicted)

# Plot the results
plt.figure(figsize=(16, 8))
plt.title('Stock Price Prediction')
plt.xlabel('Date')
plt.ylabel('Price')
plt.plot(data[train_size + seq_length:], label='True Price', color='b')
plt.plot(range(train_size + seq_length, len(data)), predicted, label='Predicted Price', color='r')
plt.legend()
plt.show()
```