```python
# Cracking the Market Code with AI-Driven Stock Price Prediction Using Time Series Analysis

# ==============================
# 1. Importing Required Libraries
# ==============================
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
import yfinance as yf

# ==============================
# 2. Data Collection
# ==============================
data = yf.download('AAPL', start='2020-01-01', end='2023-12-31')
data.to_csv("apple_stock.csv")

# ==============================
# 3. Data Preprocessing
# ==============================
data.dropna(inplace=True)
data = data[['Open', 'High', 'Low', 'Close', 'Volume']]

scaler = MinMaxScaler()
data_scaled = scaler.fit_transform(data)

# ==============================
# 4. Feature Engineering
# ==============================
X = []
y = []
window = 60
for i in range(window, len(data_scaled)):
    X.append(data_scaled[i-window:i])
    y.append(data_scaled[i, 3])  # Predicting 'Close'

X, y = np.array(X), np.array(y)

# ==============================
# 5. Train-Test Split
# ==============================
split = int(0.8 * len(X))
X_train, X_test = X[:split], X[split:]
y_train, y_test = y[:split], y[split:]

# ==============================
# 6. Model Building (LSTM)
# ==============================
model = Sequential()
model.add(LSTM(units=50, return_sequences=True, input_shape=(X.shape[1], X.shape[2])))
model.add(Dropout(0.2))
model.add(LSTM(units=50))
model.add(Dropout(0.2))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mean_squared_error')

# ==============================
# 7. Model Training
# ==============================
model.fit(X_train, y_train, epochs=20, batch_size=32, validation_data=(X_test, y_test))

# ==============================
# 8. Evaluation
# ==============================
y_pred = model.predict(X_test)
y_test_inv = scaler.inverse_transform(np.concatenate((np.zeros((y_test.shape[0], 3)),
                                                      np.expand_dims(y_test, axis=1),
                                                      np.zeros((y_test.shape[0], 1))), axis=1))[:, 3]
y_pred_inv = scaler.inverse_transform(np.concatenate((np.zeros((y_pred.shape[0], 3)),
                                                      y_pred,
                                                      np.zeros((y_pred.shape[0], 1))), axis=1))[:, 3]

print("MAE:", mean_absolute_error(y_test_inv, y_pred_inv))
print("RMSE:", np.sqrt(mean_squared_error(y_test_inv, y_pred_inv)))
print("R2 Score:", r2_score(y_test_inv, y_pred_inv))

# ==============================
# 9. Visualization
# ==============================
plt.figure(figsize=(12, 6))
plt.plot(y_test_inv, label="Actual")
plt.plot(y_pred_inv, label="Predicted")
plt.title("Actual vs Predicted Stock Price")
plt.xlabel("Time")
plt.ylabel("Price")
plt.legend()
plt.tight_layout()
plt.show()

# ==============================
# 10. Save Model
# ==============================
model.save("lstm_stock_model.h5")

```