1.Source Code

```
 import numpy as np
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
def load_stock_data(ticker, start, end):
df = yf.download(ticker, start=start, end=end)
return df[['Close']]
def preprocess_data(df, sequence_length=60):
scaler = MinMaxScaler(feature_range=(0, 1))
scaled_data = scaler.fit_transform(df)
X, y = [], []
for i in range(sequence_length, len(scaled_data)):
X.append(scaled_data[i-sequence_length:i, 0])
y.append(scaled_data[i, 0])
X = np.array(X)
y = np.array(y)
return X, y, scaler
model = Sequential([
LSTM(50, return_sequences=True, input_shape=input_shape), Dropout(0.2),
LSTM(50, return_sequences=False),
Dropout(0.2),
Dense(25),
Dense(1)

])

model.compile(optimizer='adam', loss='mean_squared_error')
return model

def train_and_predict(ticker='AAPL', start='2010-01-01', end='2024-01-01'):
df = load_stock_data(ticker, start, end)

X, y, scaler = preprocess_data(df)
```

```
# Reshape for LSTM [samples, time steps, features]
X = X.reshape((X.shape[0], X.shape[1], 1))


model = build_model((X.shape[1], 1)) model.fit(X, y, epochs=10, batch_size=32)
# Predict future values
last_60_days = df[-60:].values

last_60_scaled = scaler.transform(last_60_days)
X_test = [last_60_scaled.flatten()]

X_test = np.array(X_test).reshape((1, 60, 1))
predicted_price = model.predict(X_test)

predicted_price = scaler.inverse_transform(predicted_price)

print(f"Predicted next closing price for {ticker}: {predicted_price[0][0]}")
return predicted_price[0][0]
# Run prediction
train_and_predict()
```