use google stock prices dataset and design a time series analysis and prediction system using RNN

```
import pandas as pd
import numpy as np

df1 = pd.read_csv("Google_stock_price_train.csv")
df2 = pd.read_csv("Google_stock_price_test.csv")

df1.info()

df1['Close'] = df1['Close'].astype(str).str.replace(",","").astype(float)
df2['Close'] = df2['Close'].astype(str).str.replace(",","").astype(float)

from sklearn.preprocessing import MinMaxScaler
train_scaler = MinMaxScaler()
df1['Normalized close'] =
train_scaler.fit_transform(df1['Close'].values.reshape(-1,1))

test_scaler = MinMaxScaler()
df2['Normalized close'] =
test_scaler.fit_transform(df2['Close'].values.reshape(-1,1))

x_train = df1['Normalized close'].values[:-1].reshape(-1,1,1)
y_train = df1['Normalized close'].values[1:].reshape(-1,1,1)
x_test = df2['Normalized close'].values[:-1].reshape(-1,1,1)
y_test = df2['Normalized close'].values[1:].reshape(-1,1,1)


from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
model = Sequential()
model.add(LSTM(4, input_shape = (1,1)))
model.add(Dense(1))
model.compile(optimizer = 'adam', loss = 'mse')
model.summary()

model.fit(x_train,y_train, validation_data= (x_test,y_test), epochs = 100,
batch_size= 1)

test_loss = model.evaluate(x_test,y_test)
print('Testing loss: ', test_loss)

pred = model.predict(x_test)

y_test_actual = test_scaler.inverse_transform(y_test.reshape(-1,1))
y_test_pred = test_scaler.inverse_transform(pred.reshape(-1,1))

index = 1
print("Actual: ", y_test_actual[index])
print("Predicted: ", y_test_pred[index])
```