STOCK PRICE PREDICTION REPORT

Introduction*:*

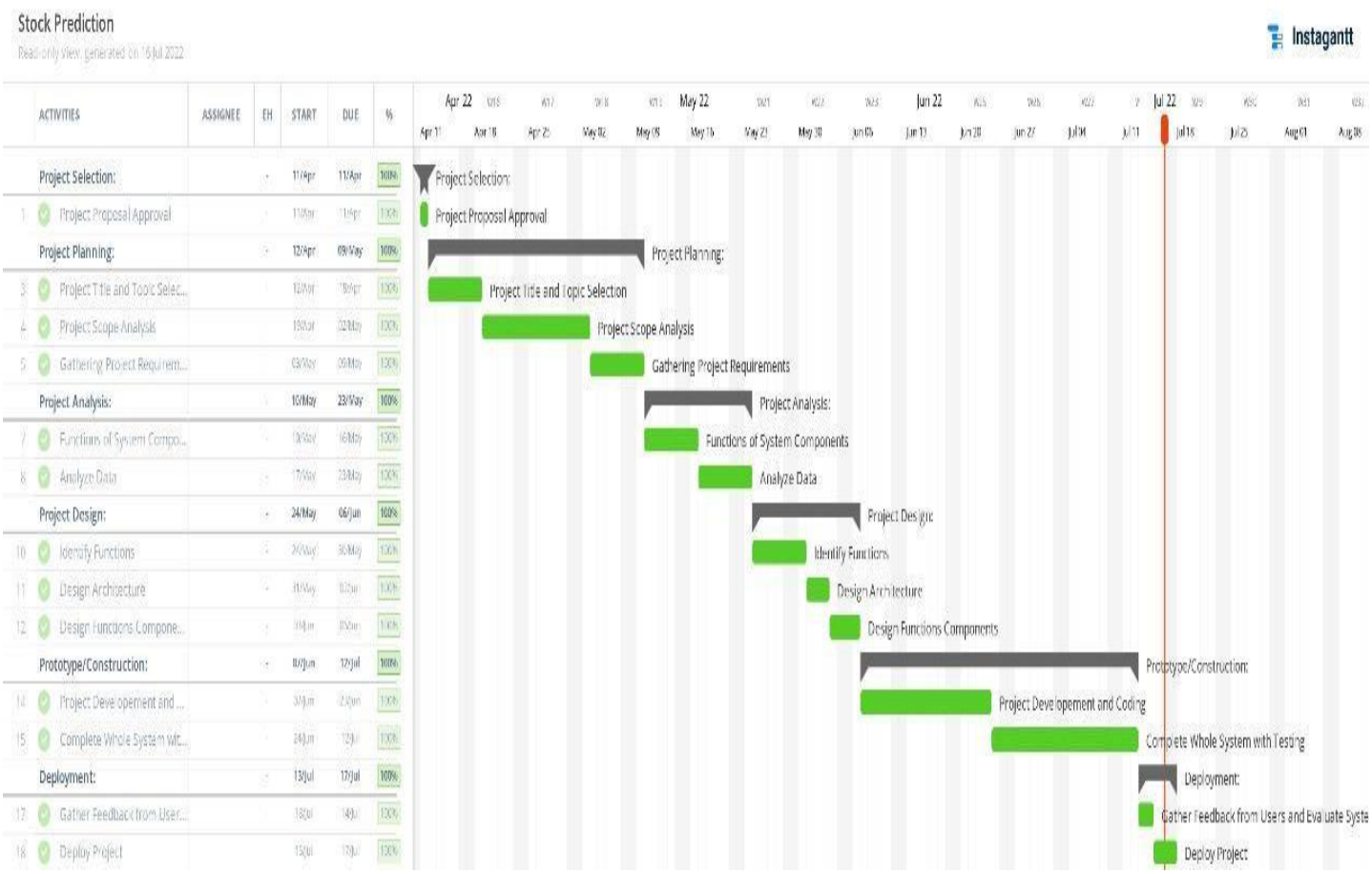
Stock market prices are highly unpredictable and volatile. This means that there are no consistent patterns in the data that allow you to model stock prices over time near-perfectly. This model considers the historical equity share price of a company price and applies RNN (Recurrent) technique called Long Short Term Memory (LSTM). The proposed approach considers available historic data of a share and it provides prediction on a particular feature. The features of shares are Opening price, day High, day Low, Close price and Volume. The prediction of stock value is a complex task which needs a robust algorithm background in order to compute the longer term share prices. Stock prices are correlated within the nature of market; hence it will be difficult to predict the costs. The proposed algorithm using the market data to predict the share price using machine learning techniques like recurrent neural network named as Long Short Term Memory, in that process weights are corrected for each data points using stochastic gradient descent. This system will provide accurate outcomes in comparison to currently available stock price predictor algorithms. The network is trained and evaluated with various sizes of input data to urge the graphical outcomes.

PROPOSED SYSTEM:

Getting Dataset from Yfinance stocks is First step. Then there is a need to extract the feature which is required for data analysis, then divide it as testing and training data,training the algorithm to predict the price and the final step is to visualize the data.The proposed system can get the output of prediction list of stock price and graph of prediction table like that user can view the final predicted result. The successful prediction of the stock will be a great asset for the stock market institutions and will provide real-life solutions to the problems that stock investors face.

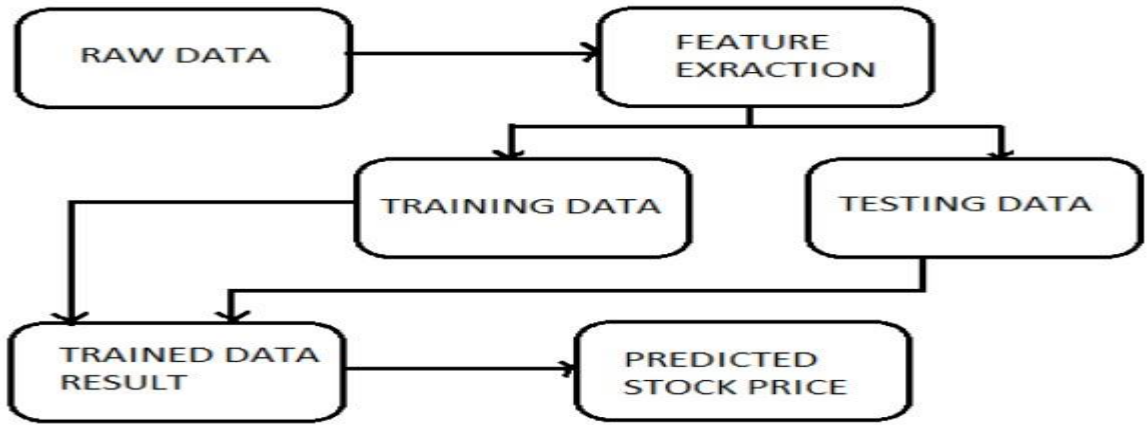
Planning:

STOCK PRICE PREDICTION REPORT



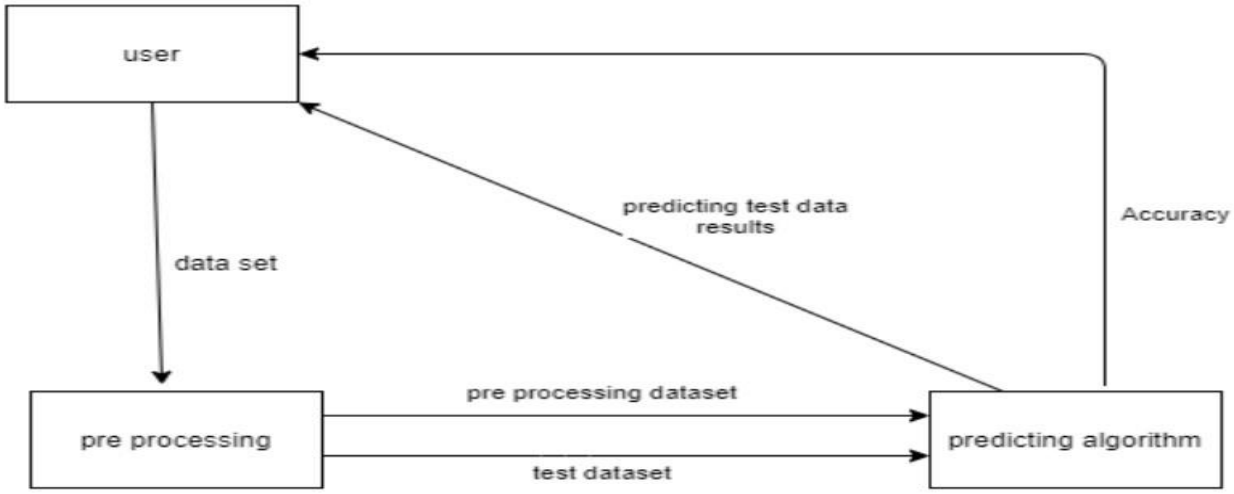
https://app.instagantt.com/shared/62d3176a78a33203f9f568eb **MODELLING:**

- **System Architecture:**



- **Collaboration Diagram:**

STOCK PRICE PREDICTION REPORT



- **Use-Case Diagram:**

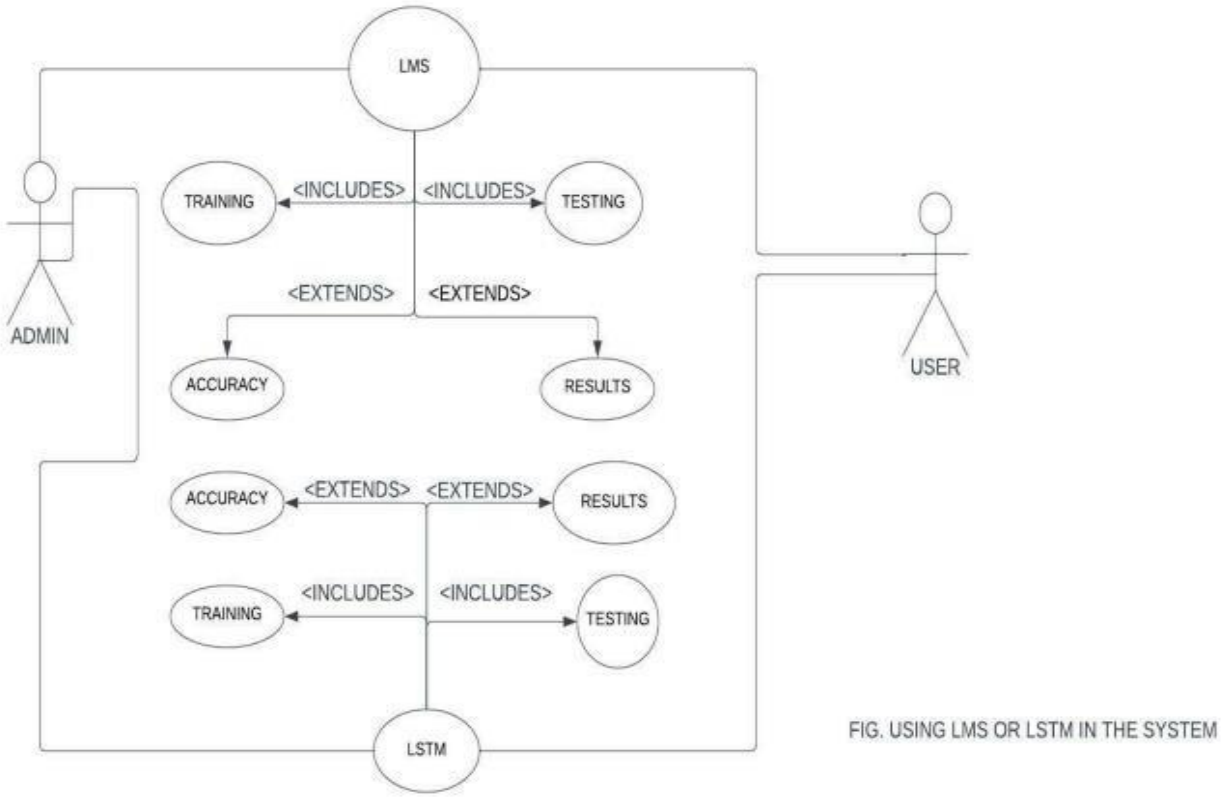
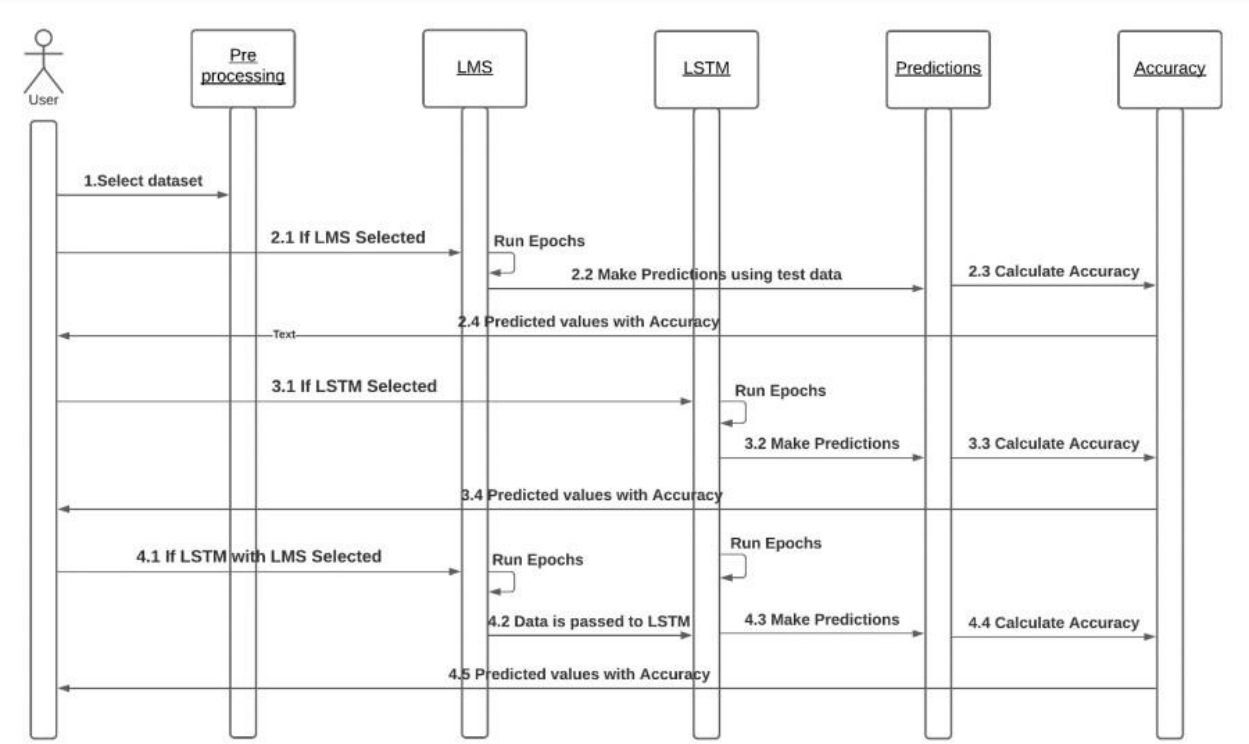


FIG. USING LMS OR LSTM IN THE SYSTEM

- **Sequence Diagram:**

STOCK PRICE PREDICTION REPORT



CONSTRUCTION:

TOOLS USED:

- Python (Numpy, Pandas, Matplotlib, Tensorflow, Keras, Sci-kitlearn) Streamlit

CODE:

import numpy as np import

pandas as pd import

matplotlib.pyplot as plt import

pandas_datareader as data

start="2010-01-01" end="2019-12-31"

#GETTING DATASET

df=data.DataReader('AAPL','yahoo',start,end) df.head()

df.tail()

STOCK PRICE PREDICTION REPORT

df=df.reset_index() df.head()

df=df.drop(["Date","Adj Close"],axis=1)

df.head() plt.plot(df.Close) df

ma100=df.Close.rolling(100).mean()

ma100

plt.figure(figsize=(12,6))

plt.plot(df.Close) plt.plot(ma100,"r")

ma200=df.Close.rolling(200).mean()

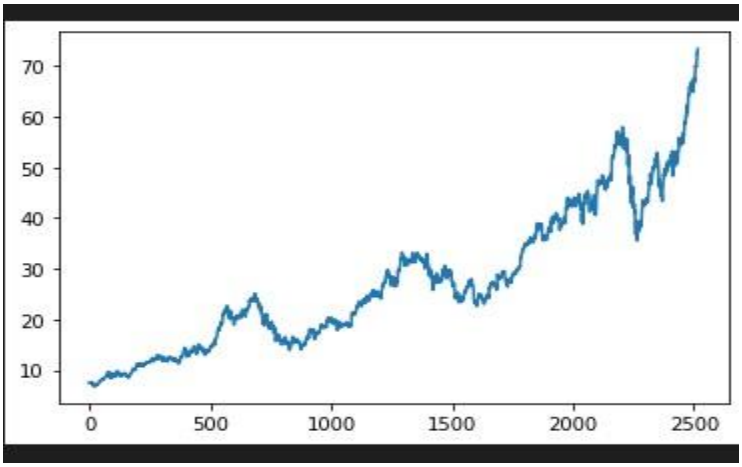
ma200

plt.figure(figsize=(12,6))

plt.plot(df.Close)

plt.plot(ma100,"r")

plt.plot(ma200,"g") df.shape



SPLITTING DATA INTO TRAINING AND TESTING

data_training=pd.DataFrame(df["Close"][0:int(len(df)*0.70)])

data_testing=pd.DataFrame(df["Close"][int(len(df)*0.70):int(len(df))])

print(data_training.shape) print(data_testing.shape)

data_training.head() data_testing.head() from sklearn.preprocessing

import MinMaxScaler scaler=MinMaxScaler(feature_range=(0,1))

STOCK PRICE PREDICTION REPORT

```
data_training_array=scaler.fit_transform(data_training)
data_training_array


data_training_array.shape


x_train=[] y_train=[]


for i in range(100,data_training_array.shape[0]):     x_train.append(data_training_array[i-
100:i])     y_train.append(data_training_array[i,0])
x_train,y_train=np.array(x_train),np.array(y_train) x_train.shape
```

#ML LSTM MODEL

```
from keras.layers import Dense,Dropout, LSTM from
keras.models import Sequential model=Sequential()
model.add(LSTM(units=50,activation="relu",return_sequences=
True,input_shape=(x_train.shape[1],1))) model.add(Dropout(0.2))


model.add(LSTM(units=60,activation="relu",return_sequences=True)) model.add(Dropout(0.3))


model.add(LSTM(units=80,activation="relu",return_sequences=True)) model.add(Dropout(0.4))


model.add(LSTM(units=50,activation="relu")) model.add(Dropout(0.5))


model.add(Dense(units=1)) model.summary()
model.compile(optimizer="adam",loss="mean_squared_error")
model.fit(x_train,y_train,epochs=50)
model.save("keras_model.h5") data_testing.head()
data_training.tail()
```

STOCK PRICE PREDICTION REPORT

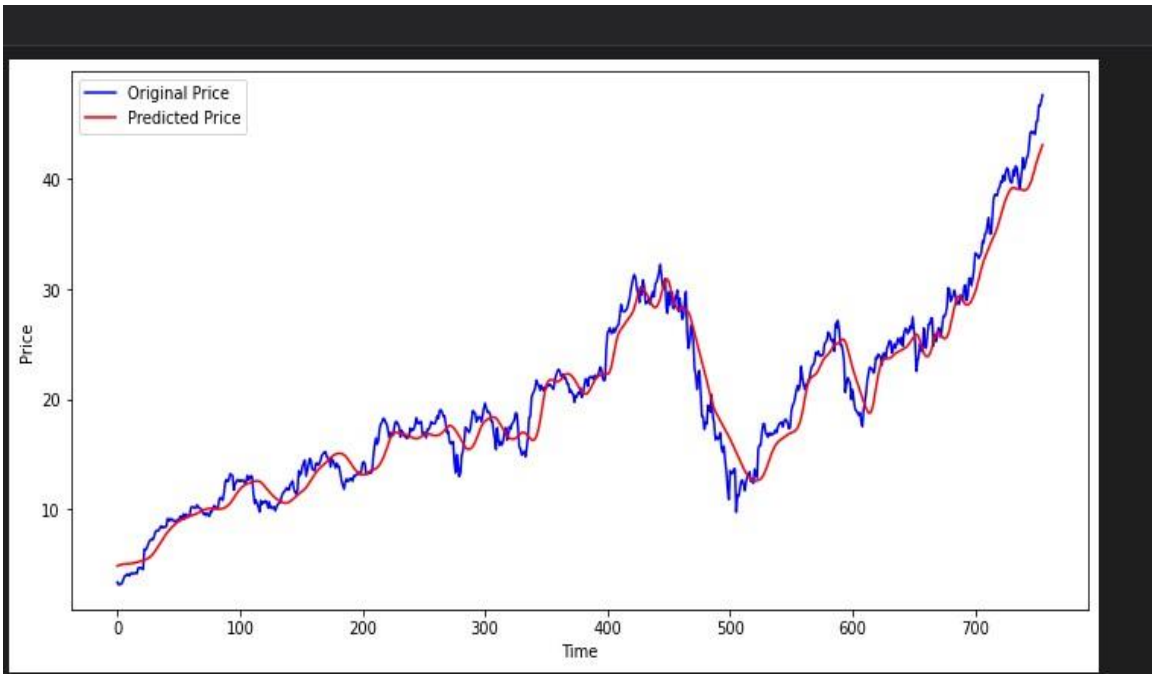
```
past_100_days=data_training.tail(100)
final_df=past_100_days.append(data_testing,ignore_index=True)
final_df.head() input_data=scaler.fit_transform(final_df)
input_data input_data.shape
x_test=[] y_test=[]

for i in range(100,input_data.shape[0]):
x_test.append(input_data[i-100:i])
y_test.append(input_data[i,0])
x_test,y_test=np.array(x_test),np.array(y_test)
print(x_test.shape) print(y_test.shape)
```

#Making Stock Prediction Model Now

```
y_predicted=model.predict(x_test)
y_predicted.shape y_test
y_predicted scaler.scale_
scale_factor=1/0.02099517
y_predicted=y_predicted*scale_factor y_test=y_test*scale_factor
plt.figure(figsize=(12,6)) plt.plot(y_test,"b",label="Original
Price") plt.plot(y_predicted,"r",label="Predicted Price")
plt.xlabel("Time") plt.ylabel("Price") plt.legend()
plt.show()
```

STOCK PRICE PREDICTION REPORT



STOCK PRICE PREDICTION ON WEBPAGE:

```
import numpy as np import pandas as
pd import matplotlib.pyplot as plt
import pandas_datareader as data
from keras.models import load_model
import streamlit as st


start="2010-01-01" end="2019-12-31"
 st.title("Stock Price Prediction")
user_input=st.text_input("Enter Stock","AAPL")
df=data.DataReader(user_input,'yahoo',start,end)
```

STOCK PRICE PREDICTION REPORT

```
#DESCRIBING OUR DATA NOW!!
 st.subheader("Data from 2010-
2020") st.write(df.describe())

#VISUALZING OUR DATA
 st.subheader("Closing Price vs Time
Chart") fig=plt.figure(figsize=(12,6))
plt.plot(df.Close) st.pyplot(fig)


st.subheader("Closing Price vs Time Chart with 100MA and 200MA")
ma100=df.Close.rolling(100).mean()
ma200=df.Close.rolling(200).mean()
fig=plt.figure(figsize=(12,6)) plt.plot(df.Close,"b")
plt.plot(ma100,"r") plt.plot(ma200,"g") st.pyplot(fig)



#split data into training and test data
data_training=pd.DataFrame(df["Close"][0:int(len(df)*0.70)])
data_testing=pd.DataFrame(df["Close"][int(len(df)*0.70):int(len(df))])
print(data_training.shape) print(data_testing.shape)
 from sklearn.preprocessing import
MinMaxScaler
scaler=MinMaxScaler(feature_range=(0,1))
data_training_array=scaler.fit_transform(data_training)


model=load_model("Keras_model.h5")


past_100_days=data_training.tail(100)
final_df=past_100_days.append(data_testing,ignore_index=True)
input_data=scaler.fit_transform(final_df)
 x_test=[]
y_test=[]
```

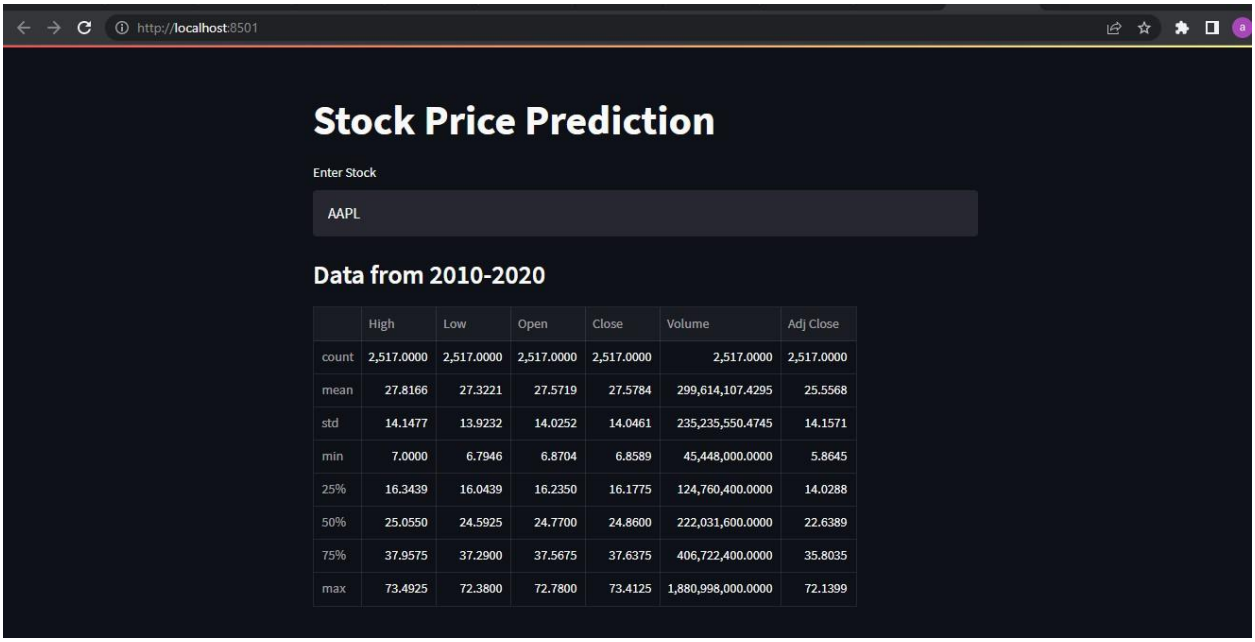
STOCK PRICE PREDICTION REPORT

```
for i in range(100,input_data.shape[0]):
x_test.append(input_data[i-100:i])     y_test.append(input_data[i,0])

x_test,y_test=np.array(x_test),np.array(y_test)
y_predicted=model.predict(x_test) scaler.scale_
scale_factor=1/0.02099517
y_predicted=y_predicted*scale_factor
y_test=y_test*scale_factor


st.subheader("Prediction VS Original")
fig2=plt.figure(figsize=(12,6))
plt.plot(y_test,"b",label="Original Price")
plt.plot(y_predicted,"r",label="Predicted Price")
plt.xlabel("Time") plt.ylabel("Price")
plt.legend() st.pyplot(fig2)
```

RESULTS:



	High	Low	Open	Close	Volume	Adj Close
count	2,517.0000	2,517.0000	2,517.0000	2,517.0000	2,517.0000	2,517.0000
mean	27.8166	27.3221	27.5719	27.5784	299,614,107.4295	25.5568
std	14.1477	13.9232	14.0252	14.0461	235,235,550.4745	14.1571
min	7.0000	6.7946	6.8704	6.8589	45,448,000.0000	5.8645
25%	16.3439	16.0439	16.2350	16.1775	124,760,400.0000	14.0288
50%	25.0550	24.5925	24.7700	24.8600	222,031,600.0000	22.6389
75%	37.9575	37.2900	37.5675	37.6375	406,722,400.0000	35.8035
max	73.4925	72.3800	72.7800	73.4125	1,880,998,000.0000	72.1399

STOCK PRICE PREDICTION REPORT

STOCK PRICE PREDICTION REPORT

TASKS:

- Stage 1: (Raw Data) In this stage, the historical stock data is collected from https://finance.yahoo.com/ and this historical data is used for the prediction of future stock prices.
- Stage 2: (Data Preprocessing) The pre-processing stage involves
  a) Data discretization: Part of data reduction but with particular importance, especially for numerical data. b) Data transformation: Normalization.

c) Data cleaning: Fill in missing values.

d) Data integration: Integration of data files.

After the dataset is transformed into a clean dataset, the dataset is divided into training and testing sets so as to evaluate. Here, the training values are taken as the more recent values. Testing data is kept as 5-10 percent of the total dataset.

- Stage 3: (Feature Extraction) In this layer, only the features which are to be fed to the LSTM Neural network. We will choose the feature from Date, open, high, low, close, and volume.
- Stage 4: (Training Neural Network) In this stage, the data is fed to the neural network and trained for prediction assigning random biases and weights. Our LSTM model is composed of a sequential input layer followed

by 2 LSTM layers and dense layer with ReLU activation and then finally a dense output layer with linear activation function.

- Stage 5: (Output Generation) In this layer, the output value generated by the output layer of the RNN is compared with the Closing value. The predicted value is compared with original value via Graph.

SUMMARY/CONCLUSION:

The popularity of stock market trading is growing rapidly, which is encouraging researchers to find out new methods for the prediction using new techniques. The forecasting technique is not only helping the researchers but it also helps investors and any person dealing with the stock market. In order to help predict the stock indices, a forecasting model with good accuracy is required. In this work, we have used one of the most precise forecasting technology using Recurrent Neural Network and Long Short-Term Memory unit which helps investors, analysts or any person interested in investing in the stock market by providing them a good knowledge of the future situation of the stock market.

Parameters used:

- Date (Date of stock price)
- Open (Open price of a share)
- Close (Closing price of a share)
- Volume/ trade
- High (Highest share value for the day)
- Low (Lowest share value for the day)

LIMITATIONS OF PROJECT:

- LSTM is prone to overfitting
- Requires lot of Time and Resources to get trained and be ready for real life application.
- It is unable to solve problem of vanishing Gradient.

- Need High Memory bandwidth to train model. Prefers Small Weight Initializations **REFERENCES:**

- Stock Price Prediction Using LSTM on Indian Share Market by Achyut Ghosh,Soumik Bose1, Giridhar Maji, Narayan C. Debnath, Soumya Sen
- Xiongwen Pang, Yanqiang Zhou, Pan Wang, Weiwei Lin, “An innovative neural network approach for stock market prediction”, 2018
- Ishita Parmar, Navanshu Agarwal, Sheirsh Saxena, Ridam Arora, Shikhin Gupta,Himanshu Dhiman, Lokesh Chouhan Department of Computer Science and Engineering National Institute of Technology, Hamirpur – 177005, INDIA – Stock Market Prediction Using Machine Learning.
- Kai Chen, Yi Zhou and FangyanDai —A LSTM-based method for stock returns prediction: A case study of China stock market,‖ IEEE International Conference on Big Data,2015.
- A.U.S.S Pradeep, SorenGoyal, J. A. Bloom, I. J. Cox and M. Miller, —Detection of statistical arbitrage using machine learning techniques in Indian Stock market, IIT Kanpur, April 15, 2013.
- Mariam Moukalled Wassim El-Hajj Mohamad Jaber Computer Science Department American University of Beirut - Automated Stock Price Prediction Using Machine Learning.
- Hyeong Kyu Choi, B.A Student Dept. of Business Administration Korea University Seoul, Korea = Stock Price Correlation Coefficient Prediction with ARIMA-LSTM Hybrid Model.
- Lavanya Ra SRM Institute of Science and Technology | SRM · Department of Computer Science - Stock Market Prediction.