A Synopsis Report On

**STOCK MARKET PREDICTION**

 (INFORMATION TECHNOLOGY)

SEMESTER 6th

SUBMITTED BY

Miss. Antima Yadav

Mr. Sakhare Sohan

Mr. Rahul Reddy

UNDER THE GUIDANCE OF

Miss. Rupali Sathe



DEPARTMENT OF INFORMATION TECHNOLOGY

**PILLAI HOC COLLEGE OF ENGINEERING & TECHNOLOGY, RASAYANI**

UNIVERSITY OF MUMBAI

AY 2022-23

**Pillai HOC College of Engineering & Technology, Rasayani**

Year: 2022-2023

**INFORMATION TECHNOLOGY**

**Certificate**

This is to certify that the project entitled **“STOCK MARKET PREDICTION”** is successfully completed by following students:

|  |  |
| --- | --- |
| **Student Name** | **Roll No.** |
| Miss. Antima Yadav | 68 |
| Mr. Sakhare Sohan | 59 |
| Mr. Rahul Reddy | 58 |

As per the syllabus & in partial fulfilment for the completion Bachelor’s degree in Information Technology from University of Mumbai, it is also to certify that this is the original work of the candidate done during the academic year 2022-23.

\_\_\_\_\_\_\_\_\_\_\_\_                                                                         \_\_\_\_\_\_\_\_\_\_\_\_

Project Guide                                                                         Head of Department

\_\_\_\_\_\_\_\_\_\_\_\_

Principal

\_\_\_\_\_\_\_\_\_\_\_\_                                                                            \_\_\_\_\_\_\_\_\_\_\_\_

Internal Examiner                                                                     External Examiner

**ACKNOWLEDGEMENT**

A project is never complete without the guidance of experts who have already gone through this in the past before and hence become master of it and as a result, our guides. So, we would like to take this opportunity to thank all those individuals who helped us in visualizing our project.

We express our deep gratitude to our project guide **Miss. Rupali Sathe** for providing timely assistance to our query and guidance that she gave owing to her experience in this field for the past many years. She had indeed been a lighthouse for us on this journey. We would also like to thank our guide for providing us with her expert opinion and valuable suggestions at every stage of the project.

We would also take this opportunity to thank our project coordinator **Miss. Rupali Sathe**. For her guidance in selecting this project and also for providing us all the details on the proper presentation of this project.

We would like to take this opportunity to thank, **Head of Department of Information Technology, Dr. Divya Chirayil**. for her motivation and valuable support. Also, acknowledgment is complete without thanking the teaching and non-teaching staff of the department for their kind support.

We extend our sincere appreciation to all our Professors and Principal **Dr.J. W. Bakal, Principal of Pillai HOC College of Engineering and Technology, Rasayani** for providing the infrastructure and resources required for the project.

Thanking You,

042002

**Abstract:**

It has never been easy to invest in a set of assets, the abnormally of financial market does not allow simple models to predict future asset values with higher accuracy. Machine learning, which consist of making computers perform tasks that normally requiring human intelligence is currently the dominant trend in scientific research. This article aims to build a model using Recurrent Neural Networks (RNN) and especially Long-Short Term Memory model (LSTM) to predict future stock market values. The main objective of this paper is to see in which precision a Machine learning algorithm can predict and how much the epochs can improve our model.

**Index**

|  |  |
| --- | --- |
| **Sr. No.** | **Contents** |
| **1.**  1.1  1.2  1.3  1.4  1.5 | **Introduction**  Objectives  Scope  Problem Statement  Problem Solution  Literature Survey |
| **2.**  2.1  2.2  2.3 | **Methodology & Implementation**  Methodology  Workflow Diagram  LSTM Model |
| **3.**  3.1  3.2  3.3 | **Requirement gathering & Analysis**  Yfinance API  Colab  Flask |
| **4.**  4.1  4.2  4.3  4.4 | **Result**  Screenshot  Colab Code  Flask Code  HTML Code |
| **5.** | **Conclusion** |
| **6.** | **References** |

**1. Introduction**

Several studies have been the subject of using machine learning in the quantitative financial, predicting prices of managing and constricting entire portfolio of assets, as well as, investment process, and many other operations can be covered by machine learning algorithms. In general machine learning is a term used for all algorithm’s methods using computers to reveal patterns based only on data and not using any programming instructions.

**1.1 Objective**

The aims to build a model using Recurrent Neural Networks (RNN) and especially Long-Short Term Memory model (LSTM) to predict future stock market values. The main objective of this project is to create a Machine learning algorithm can predict and how much the epochs can improve our model. In general, to create a machine learning algorithm using computers to reveal patterns based only on data and not using any programming instructions.

**1.2 Scope**

* Building a type of models offers a mechanism that combine weak sources of information and make it a strange tool that can be used efficiently.
* Building algorithms can reveal complex patterns characterized by non-linearity as well as some relations that are difficult to detect with linear algorithms.
* The models do depend one long term memory (passed sequences of data), in this regard a class of machine learning algorithms based on Recurrent Neural Network prove to be very useful in financial market price prediction and forecasting.
* To use ML algorithm based on LSTM RNN to forecast the adjusted closing prices for a portfolio of assets, the main objective here is to obtain the most accurate trained algorithm, to predict future values for our portfolio.

**1.3 Problem Statement**

* The accuracy of any ML model is highly dependent on the quantity and quality of data used for training. The stock market is a highly complex and dynamic system, and it can be difficult to gather enough relevant data to accurately predict future prices.
* Models don't depend one long term memory (passed sequences of data).
* The stock market can be subject to manipulation by large institutional investors or other market actors, which can make it difficult for ML models to accurately predict future prices.
* ML models can sometimes become too complex and end up overfitting to the training data. This means that they perform well on the training data, but poorly on new, unseen data. In the context of stock market prediction, this can lead to inaccurate predictions and financial losses.

**1.4 Problem Solution**

* LSTM models are well-suited to handling non-stationary data, as they are capable of capturing long-term dependencies and can adapt to changes in the statistical properties of the data over time.
* mitigating the impact of black swan events is to use a combination of LSTM models with other types of models that are better suited to handling sudden changes in the market, such as decision trees or support vector machines.
* LSTM models can be trained on a variety of different input features, including news articles, social media sentiment, and other data sources that may provide additional insight into market manipulation or other types of market behaviour.

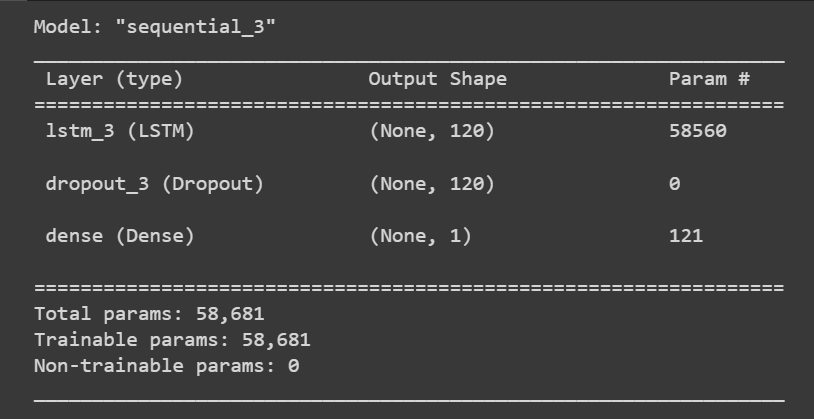
**1.5 Literature Survey**

* "Stock Price Prediction Using LSTM, RNN and CNN-Sliding Window Model," by Rajat Agarwal and Ayush Kumar Jain, published in the International Journal of Computer Science and Information Technology Research. This paper compares the performance of LSTM, recurrent neural network (RNN), and convolutional neural network (CNN) models for stock market prediction, finding that the LSTM model outperformed the other models.
* "A Deep Learning Approach to Stock Price Forecasting," by Jin Zhang and Yi Zhang, published in the IEEE Access journal. This paper proposes a novel LSTM model for stock market prediction that takes into account both technical and fundamental indicators, and finds that the LSTM model outperforms traditional statistical models.
* "Stock Price Prediction Using Deep Learning on News Articles," by Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang, published in the Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. This paper uses an LSTM model to predict stock prices based on news articles, and finds that the model outperforms a baseline model that does not incorporate textual data.

**2. Methodology and Implementation**

**2.1** **Methodology**

* The data in this paper consist of the daily opening prices of two stocks in the New York Stock Exchange NYSE (AAPL and TSLA) extracted from yahoo finance, for AAPL our data series cover the period going from 3/20/2013 to 3/20/2023, over a duration of 10 years. To build our model we are going to use the LSTM RNN, our model uses 70% of data for training and the other 30% of data for testing. For training we use mean squared error to optimize our model. Also, we used different Epochs for training data (12 epochs, 25 epochs and 50 epochs) our model will be structured as follow



**2.2** **WorkFlow Diagram**

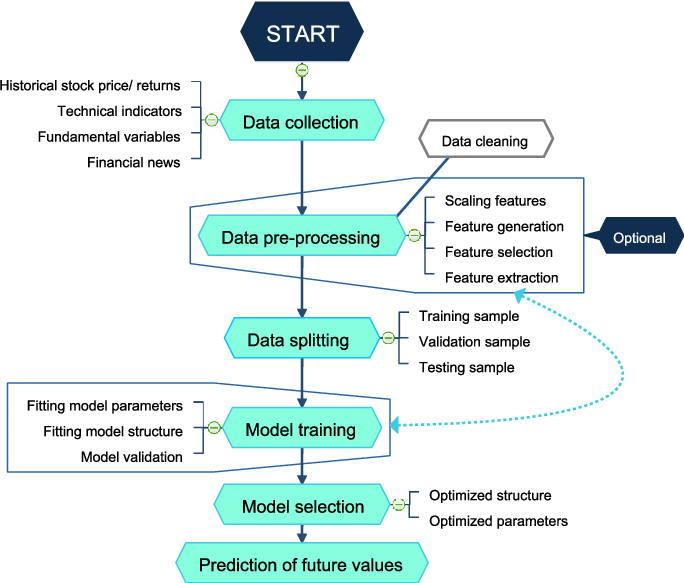
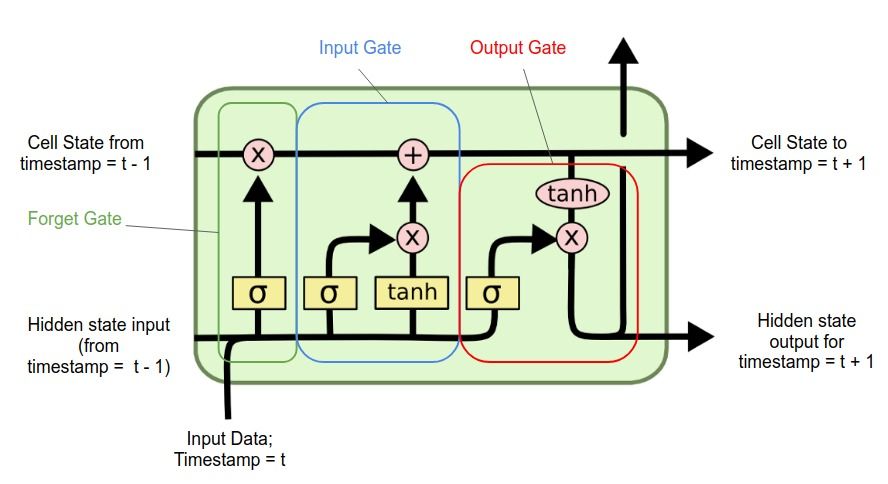


Figure 1: Block diagram of system

**2.3** **LSTM Model**



**3.** **REQUIREMENT GATHERING AND ANALYSIS**

**3.1 yfinance**

The yfinance is one of the famous modules in Python, which is used to collect online data, and with it, we can collect the financial data of Yahoo. With the help of the yfinance module, we retrieve and collect the company's financial information (such as financial ratios, etc.) as well as the histories of marketing data by using its functions. But, before we start learning more about this module and its implementation as well as applications, we have to install the yfinance module in our system (as it is not a built-in module in Python).



Figure 2: yfinance API

**3.2 Google Colab**

* Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing access free of charge to computing resources including GPUs.



Figure 3: Google Colab

**3.3 Flask**

**Flask** is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

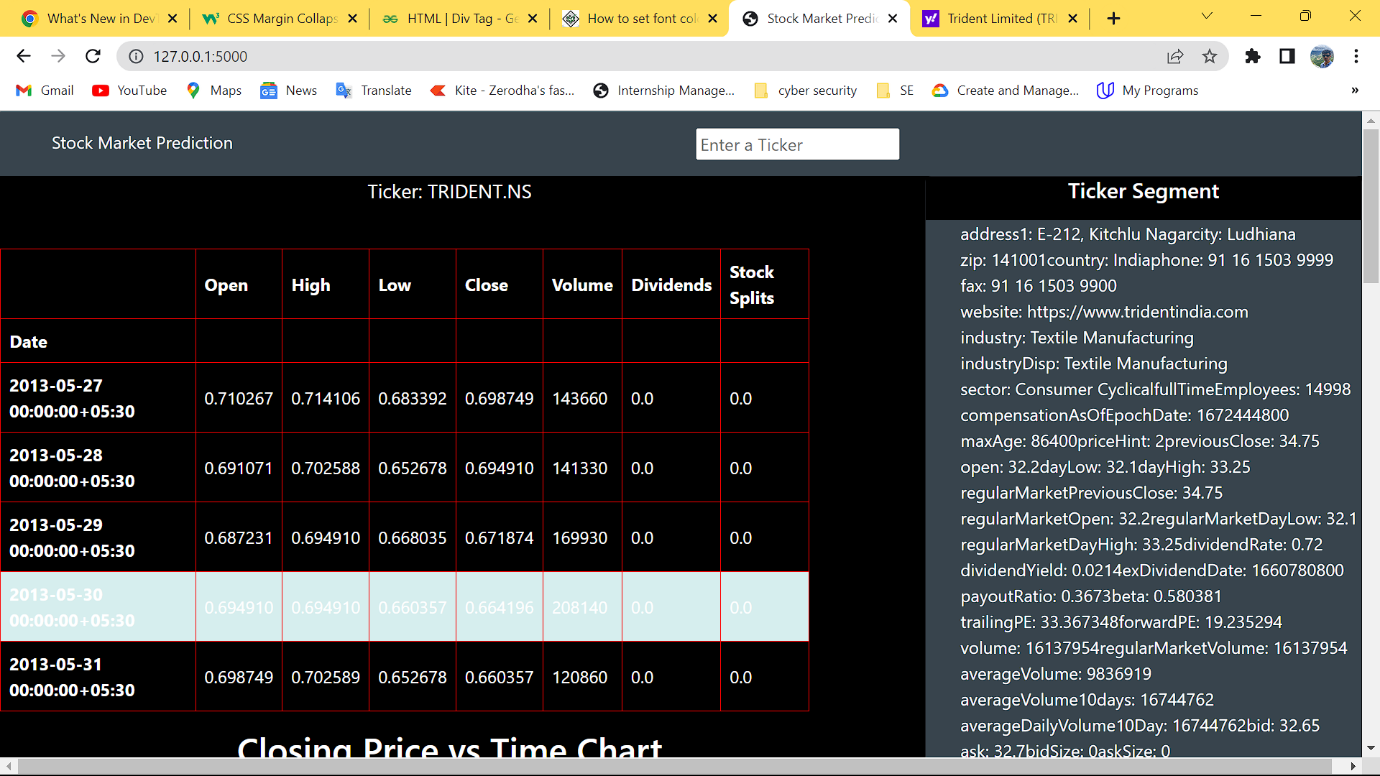


Figure 4: Flask

**4. Results**

* For different data set we can observe that training with less data and more epochs can improve our testing result and at the same time allow us to have beater forecasting and prediction values. The following table shows the precision of our training and testing for all the epochs for both AAPL and TSLA asset price

**4.1 Screenshot**



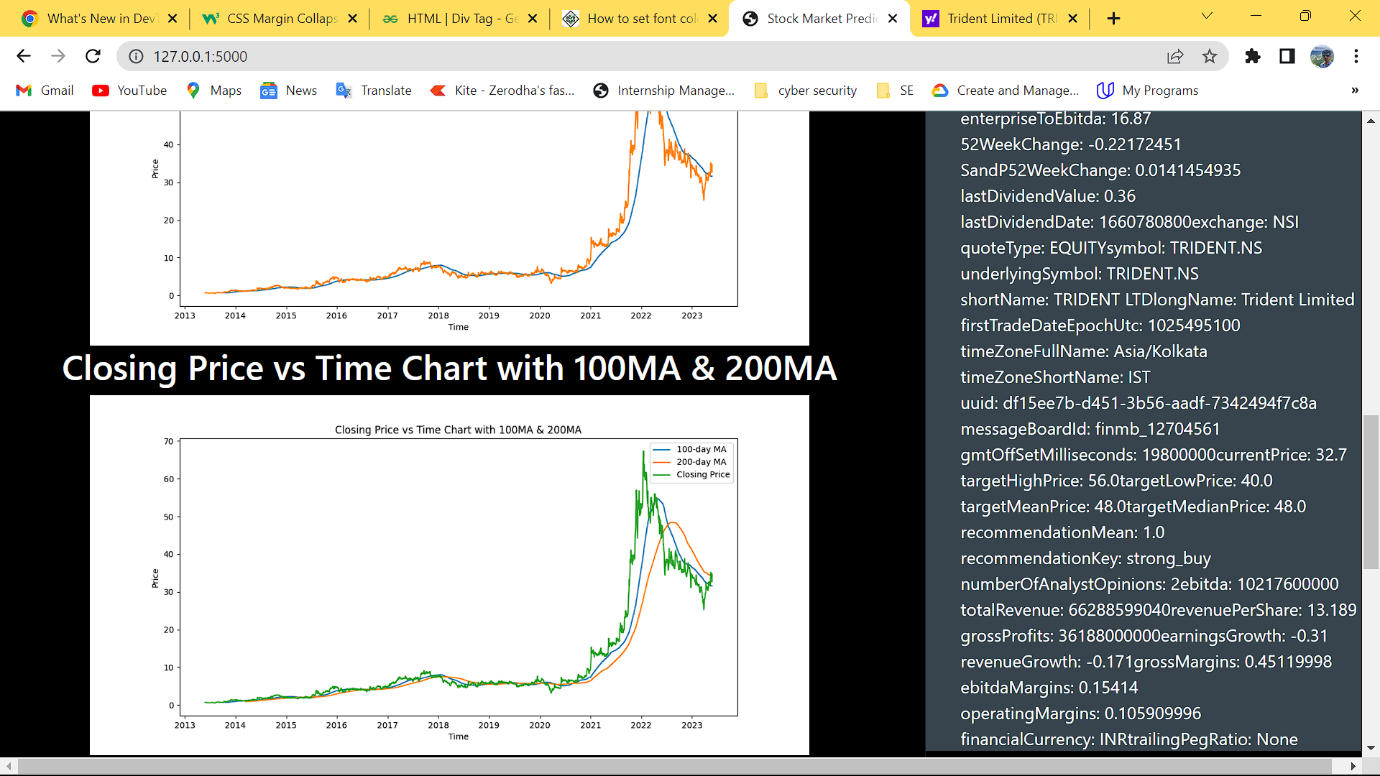


Figure 6: UI of Web Application

**4.2 Colab code:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import pandas\_datareader as data

!pip install yfinance

import yfinance as yf

aapl = yf.Ticker("AAPL")

hist = aapl.history(period="10y")

hist.head()

hist = hist.reset\_index()

hist.head()

hist = hist.drop(['Date', 'Dividends',  'Stock Splits'], axis = 1)

hist.head()

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

ma100

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

plt.plot(hist.Close)

plt.plot(ma100, 'r')

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

ma200

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

plt.plot(hist.Close)

plt.plot(ma100, 'r')

plt.plot(ma200, 'g')

hist.shape

data\_training = pd.DataFrame(hist['Close'][0:int(len(hist)\*0.70)])

data\_testing = pd.DataFrame(hist['Close'][int(len(hist)\*0.70):int(len(hist))])

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)

data\_training\_array

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

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 = Sequential()

model.add(LSTM(units = 60, activation = 'relu', return\_sequences = True))

model.add(Dropout(0.3))

model = Sequential()

model.add(LSTM(units = 80, activation = 'relu', return\_sequences = True))

model.add(Dropout(0.4))

model = Sequential()

model.add(LSTM(units = 120, activation = 'relu'))

model.add(Dropout(0.5))

model.add(Dense(units = 1))

model.compile(optimizer='adam', loss = 'mean\_squared\_error')

model.fit(x\_train, y\_train, epochs = 50)

model.save('keras\_model.h5')

//Model Created

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)

y\_predicted = model.predict(x\_test)

y\_predicted.shape

scaler.scale\_

scale\_factor = 1/0.00795641

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()

**4.3 Flask App**

from flask import Flask, render\_template, request

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from keras.models import load\_model

import streamlit as st

import yfinance as yf

import base64

from io import BytesIO

from sklearn.preprocessing import MinMaxScaler

app = Flask(\_\_name\_\_)

user\_input = 'AAPL'

model = load\_model('D:/FlaskApp/venv/keras\_model.h5')

@app.route('/', methods=['GET'])

def home():

    aapl = yf.Ticker('AAPL')

    df = aapl.history(period="10y")

    table\_html = df.head().to\_html(border=None)

    describe = df.describe()

            # Fetch stock data and calculate moving averages

    # Assume that you have already fetched the stock data and stored it in a DataFrame named `df`

    ma100 = df['Close'].rolling(window=100).mean()

    ma200 = df['Close'].rolling(window=200).mean()

    # Create chart 1: Closing Price vs Time Chart

    fig1 = plt.figure(figsize=(12, 6))

    plt.plot(df['Close'])

    plt.xlabel('Time')

    plt.ylabel('Price')

    plt.title('Closing Price vs Time Chart')

    chart\_image1 = get\_base64\_chart\_image(fig1)

    # Create chart 2: Closing Price vs Time Chart with 100MA

    fig2 = plt.figure(figsize=(12, 6))

    plt.plot(ma100, label='100-day MA')

    plt.plot(df['Close'], label='Closing Price')

    plt.xlabel('Time')

    plt.ylabel('Price')

    plt.title('Closing Price vs Time Chart with 100MA')

    plt.legend()

    chart\_image2 = get\_base64\_chart\_image(fig2)

    # Create chart 3: Closing Price vs Time Chart with 100MA & 200MA

    fig3 = plt.figure(figsize=(12, 6))

    plt.plot(ma100, label='100-day MA')

    plt.plot(ma200, label='200-day MA')

    plt.plot(df['Close'], label='Closing Price')

    plt.xlabel('Time')

    plt.ylabel('Price')

    plt.title('Closing Price vs Time Chart with 100MA & 200MA')

    plt.legend()

    chart\_image3 = get\_base64\_chart\_image(fig3)

    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))])

    scaler = MinMaxScaler(feature\_range=(0,1))

    data\_training\_array = scaler.fit\_transform(data\_training)

    # Prepare input data for prediction

    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 = []

    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)

    # Make predictions

    y\_predicted = model.predict(x\_test)

    scaler = scaler.scale\_

    scale\_factor = 1/scaler[0]

    y\_predicted = y\_predicted \*scale\_factor

    y\_test = y\_test \*scale\_factor

    # Plot the chart

    fig4 = 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()

    chart\_image4 = get\_base64\_chart\_image(fig4)

    return render\_template("home.html", user\_input = user\_input, table = table\_html, describe = describe, chart\_image1=chart\_image1, chart\_image2=chart\_image2, chart\_image3=chart\_image3, chart\_image4=chart\_image4)

@app.route('/', methods=(['POST']))

def predict():

    if request.method == 'POST':

        user\_input = request.form.get('ticker')

        aapl = yf.Ticker(user\_input)

        df = aapl.history(period="10y")

        table\_html = df.head().to\_html(border=None)

        describe = df.describe()

            # Fetch stock data and calculate moving averages

    # Assume that you have already fetched the stock data and stored it in a DataFrame named `df`

        ma100 = df['Close'].rolling(window=100).mean()

        ma200 = df['Close'].rolling(window=200).mean()

    # Create chart 1: Closing Price vs Time Chart

        fig1 = plt.figure(figsize=(12, 6))

        plt.plot(df['Close'])

        plt.xlabel('Time')

        plt.ylabel('Price')

        plt.title('Closing Price vs Time Chart')

        chart\_image1 = get\_base64\_chart\_image(fig1)

    # Create chart 2: Closing Price vs Time Chart with 100MA

        fig2 = plt.figure(figsize=(12, 6))

        plt.plot(ma100, label='100-day MA')

        plt.plot(df['Close'], label='Closing Price')

        plt.xlabel('Time')

        plt.ylabel('Price')

        plt.title('Closing Price vs Time Chart with 100MA')

        plt.legend()

        chart\_image2 = get\_base64\_chart\_image(fig2)

    # Create chart 3: Closing Price vs Time Chart with 100MA & 200MA

        fig3 = plt.figure(figsize=(12, 6))

        plt.plot(ma100, label='100-day MA')

        plt.plot(ma200, label='200-day MA')

        plt.plot(df['Close'], label='Closing Price')

        plt.xlabel('Time')

        plt.ylabel('Price')

        plt.title('Closing Price vs Time Chart with 100MA & 200MA')

        plt.legend()

        chart\_image3 = get\_base64\_chart\_image(fig3)

        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))])

        scaler = MinMaxScaler(feature\_range=(0,1))

        data\_training\_array = scaler.fit\_transform(data\_training)

    # Prepare input data for prediction

        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 = []

        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)

    # Make predictions

        y\_predicted = model.predict(x\_test)

        scaler = scaler.scale\_

        scale\_factor = 1/scaler[0]

        y\_predicted = y\_predicted \*scale\_factor

        y\_test = y\_test \*scale\_factor

    # Plot the chart

        fig4 = 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()

        chart\_image4 = get\_base64\_chart\_image(fig4)

        return render\_template("home.html", user\_input = user\_input, table = table\_html, chart\_image1=chart\_image1, chart\_image2=chart\_image2, chart\_image3=chart\_image3, chart\_image4=chart\_image4)

def get\_base64\_chart\_image(fig):

    # Convert chart to base64-encoded image

    buffer = BytesIO()

    plt.savefig(buffer, format='png')

    buffer.seek(0)

    chart\_image = base64.b64encode(buffer.getvalue()).decode('utf-8')

    return chart\_image

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(host='0.0.0.0', port=5000, debug=True)

**4.4 HTML Code (Template)**

<!DOCTYPE html>

<html lang="en">

<head>

    <style>

    .my-table {

      border-collapse: collapse;

      width: 90%;

  padding: 8px;

  color: white;

}

    .my-table th {

      padding: 8px;

  text-align: left;

  border: 1px solid black;

  border-color:red;

  color: white;

}

    .my-table td {

      padding: 8px;

  text-align: left;

  border: 1px solid black;

  border-color:red;

  color: white;

}

.my-table tr:hover {background-color: #D6EEEE;}

.footer {

  display: flex;

  flex-flow: row wrap;

  padding: 30px 30px 20px 30px;

  color: white;

  background-color: black;

  border-top: 1px solid #e5e5e5;

}

.footer > \* {

  flex:  1 100%;

}

.footer\_\_addr {

  margin-right: 1.25em;

  margin-bottom: 2em;

}

.footer\_\_logo {

  font-family: 'Pacifico', cursive;

  font-weight: 400;

  text-transform: lowercase;

  font-size: 1.5rem;

}

.footer\_\_addr h2 {

  margin-top: 1.3em;

  font-size: 15px;

  font-weight: 400;

}

.nav\_\_title {

  font-weight: 400;

  font-size: 15px;

}

.footer address {

  font-style: normal;

  color: #999;

}

.footer\_\_btn {

  display: flex;

  align-items: center;

  justify-content: center;

  height: 36px;

  max-width: max-content;

  background-color: rgb(33, 33, 33, 0.07);

  border-radius: 100px;

  color: #2f2f2f;

  line-height: 0;

  margin: 0.6em 0;

  font-size: 1rem;

  padding: 0 1.3em;

}

.footer ul {

  list-style: none;

  padding-left: 0;

}

.footer li {

  line-height: 2em;

}

.footer a {

  text-decoration: none;

}

.footer\_\_nav {

  display: flex;

  flex-flow: row wrap;

}

.footer\_\_nav > \* {

  flex: 1 50%;

  margin-right: 1.25em;

}

.nav\_\_ul a {

  color: #999;

}

.nav\_\_ul--extra {

  column-count: 2;

  column-gap: 1.25em;

}

.legal {

  display: flex;

  flex-wrap: wrap;

  color: #999;

}

.legal\_\_links {

  display: flex;

  align-items: center;

}

.heart {

  color: #2f2f2f;

}

@media screen and (min-width: 24.375em) {

  .legal .legal\_\_links {

    margin-left: auto;

  }

}

@media screen and (min-width: 40.375em) {

  .footer\_\_nav > \* {

    flex: 1;

  }

  .nav\_\_item--extra {

    flex-grow: 2;

  }

  .footer\_\_addr {

    flex: 1 0px;

  }

  .footer\_\_nav {

    flex: 2 0px;

  }

}

body {

          background-color:#38444d;

        }

        dl, ol, ul {

          list-style-type: none;

          overflow: hidden;

          background-color: #38444d;

          text-align: center;

          margin-top: 0;

          margin-bottom: 0;

        }

        div {

            text-align: center;

            padding: auto;

        }

        li {

          float: left;

        }

        .heading1{

            background-color: #38444d;

            color: white;

            margin: 16px;

            padding: 0;

            float: left;

        }

        .menu {

            margin-left: auto;

            margin-right: auto;

            background-color: #38444d;

        }

        form {

          padding: 15px;

        }

        p {

          margin-top: auto;

          margin-bottom: auto;

        }

        h2{

          color: white;

        }

        img{

          width: 100%;

        }

    </style>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ENjdO4Dr2bkBIFxQpeoTz1HIcje39Wm4jDKdf19U8gI4ddQ3GYNS7NTKfAdVQSZe" crossorigin="anonymous"></script>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-KK94CHFLLe+nY2dmCWGMq91rCGa5gtU4mk92HdvYe+M/SXH301p5ILy+dN9+nJOZ" crossorigin="anonymous">

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Stock Market Prediction</title>

</head>

<body>

  <nav>

    <ul style="margin-bottom: 0;">

    <div class="heading1">

    <!--Navbar-->

        Stock Market Prediction

    </div>

        <div class="menu">

          <form action="/" method="post">

            <input type="text" name="ticker" placeholder="Enter a Ticker"></input>

        </form>

        </div>

    </ul>

    </nav>

    <center >

    <div class="row" style="background-color:black">

        <div class="col-8">

    <p style="font-size: large; color: white; margin-bottom: 0; margin-top: 15;padding-top: 15;">Ticker:  {{user\_input}}</p>

    <br>

    <p style="padding-left: 5;">{{ table | safe}}</p>

    <h2 style="color: white;">Closing Price vs Time Chart</h2>

    <img src="data:image/png;base64,{{ chart\_image1 }}" alt="Chart 1" style="width: 80%;">

    <h2 style="color: white;">Closing Price vs Time Chart with 100MA</h2>

    <img src="data:image/png;base64,{{ chart\_image2 }}" alt="Chart 2" style="width: 80%;">

    <h2 style="color: white;">Closing Price vs Time Chart with 100MA & 200MA</h2>

    <img src="data:image/png;base64,{{ chart\_image3 }}" alt="Chart 3" style="width: 80%;">

    <h2 style="color: white;">Prediction</h2>

    <img src="data:image/png;base64,{{ chart\_image4 }}" alt="Chart 4" style="width: 80%;">

        </div>

    <div class="col-4">

        <div class="card text-bg-dark" style="background-color:black">

          <div  style="background-color:black">

            <h5 class="card-title">Ticker Segment</h5>

            <p class="card-text" style="background-color:black">

              <ul>

              {% for key, value in news.items() %}

              <li>{{ key }}: {{ value }}</li>

              {% endfor %}

              </ul>

               </p>

            <p class="card-text"><small>Last updated 3 mins ago</small></p>

          </div>

        </div>

    </div>

    </div>

  </center>

<footer class="footer">

    <div class="footer\_\_addr">

      <h1 class="footer\_\_logo">Stock Prediction</h1>

      <h2>Contact</h2>

      <address>

        5534 Somewhere In. The World 22193-10212<br>

        <a class="footer\_\_btn" href="mailto:example@gmail.com">Email Us</a>

      </address>

    </div>

    <ul class="footer\_\_nav">

      <li class="nav\_\_item">

        <h2 class="nav\_\_title">Media</h2>

        <ul class="nav\_\_ul">

          <li>

            <a href="#">Online</a>

          </li>

          <li>

            <a href="#">Print</a>

          </li>

          <li>

            <a href="#">Alternative Ads</a>

          </li>

        </ul>

      </li>

      <li class="nav\_\_item nav\_\_item--extra">

        <h2 class="nav\_\_title">Technology</h2>

        <ul class="nav\_\_ul nav\_\_ul--extra">

          <li>

            <a href="#">Hardware Design</a>

          </li>

          <li>

            <a href="#">Software Design</a>

          </li>

          <li>

            <a href="#">Digital Signage</a>

          </li>

          <li>

            <a href="#">Automation</a>

          </li>

          <li>

            <a href="#">Artificial Intelligence</a>

          </li>

          <li>

            <a href="#">IoT</a>

          </li>

        </ul>

      </li>

      <li class="nav\_\_item">

        <h2 class="nav\_\_title">Legal</h2>

        <ul class="nav\_\_ul">

          <li>

            <a href="#">Privacy Policy</a>

          </li>

          <li>

            <a href="#">Terms of Use</a>

          </li>

          <li>

            <a href="#">Sitemap</a>

          </li>

        </ul>

      </li>

    </ul>

    <div class="legal">

      <p>&copy; PHCET TEIT. All rights reserved.</p>

      <div class="legal\_\_links">

        <span>Made with <span class="heart">♥</span> remotely from Anywhere</span>

      </div>

    </div>

  </footer>

</body>

</html>

**5. Conclusion**

This paper proposes RNN based on LSTM built to forecast future values for both AAPL and TSLA assets, the result of our model has shown some promising result. The testing result conform that our model is capable of tracing the evolution of opening prices for both assets. For our future work we will try to find the best sets for bout data length and number of training epochs that beater suit our assets and maximize our predictions accuracy.

**6. References**

* [1] Batres-Estrada, B. (2015). Deep learning for multivariate financial time series.
* [2] Emerson, S., Kennedy, R., O'Shea, L., & O'Brien, J. (2019, May). Trends and Applications of Machine Learning in Quantitative Finance. In 8th International Conference on Economics and Finance Research (ICEFR 2019).
* [3] Heaton, J. B., Polson, N. G., & Witte, J. H. (2017). Deep learning for finance: deep portfolios. Applied Stochastic Models in Business and Industry, 33(1), 3-12.
* [4] Moritz, B., & Zimmermann, T. (2016). Tree-based conditional portfolio sorts: The relation between past and future stock returns. Available at SSRN 2740751.
* [5] Olah, C. (2015). Understanding lstm networks–colah’s blog. Colah. github. io.
* [6] Paiva, F. D., Cardoso, R. T. N., Hanaoka, G. P., & Duarte, W. M. (2018). Decision-Making for Financial Trading: A Fusion Approach of Machine Learning and Portfolio Selection. Expert Systems with Applications.
* [7] Patterson J., 2017. Deep Learning: A Practitioner’s Approach, O’Reilly Media.
* [8] Siami-Namini, S., & Namin, A. S. (2018). Forecasting economics and financial time series: Arima vs. lstm. arXiv preprint arXiv:1803.06386.
* [9] Takeuchi, L., & Lee, Y. Y. A. (2013). Applying deep learning to enhance momentum trading strategies in stocks. In Technical Report. Stanford University.