TEAM ID: PNT2022TMID52922

Crude Oil Price Prediction

TEAM MEMBERS:

Karthik S

Anish Panicker

Akash P

Kirthivasan

1. INTRODUCTION

1.1 PROJECT OVERREVIEW

Crude oil is one of the most crucial resources in today's world since it is the main fuel and because its price directly affects oil exploration, exploitation, and other activities as well as the environment and our economy. It has become imperative to predict oil prices since it benefits so many big and small businesses, people, and the government. Due to the evaporative nature of crude oil, it is very challenging to estimate its price with any degree of accuracy. The primary benefit of this crude oil price prediction using artificial intelligence is that it continually captures the volatile pattern of the crude oil prices that have been included by determining the ideal lag and number of the delay effect that regulates the prices of crude oil.

1.2 PURPOSE

Since there is no elasticity in the oil demand, producers will benefit from the price increase since it will result in higher profits. However, oil importers will pay more for their oil purchases. As the most traded commodity, oil, the repercussions are fairly substantial. Rising oil prices may even cause oil exporters to gain economic and political clout at the expense of oil importers. The price of crude oil is affected by a variety of variables.

The major goal of this project is to employ neural networks to forecast the price of crude oil. This decision enables us to purchase crude oil at the optimal moment. The greatest option for this type of prediction is time series analysis since we are utilizing past data on crude oil prices to forecast future crude oil prices. Therefore, to complete the assignment, we would construct an RNN (Recurrent Neural Network) utilizing LSTM.

2. LITERATURE SURVEY

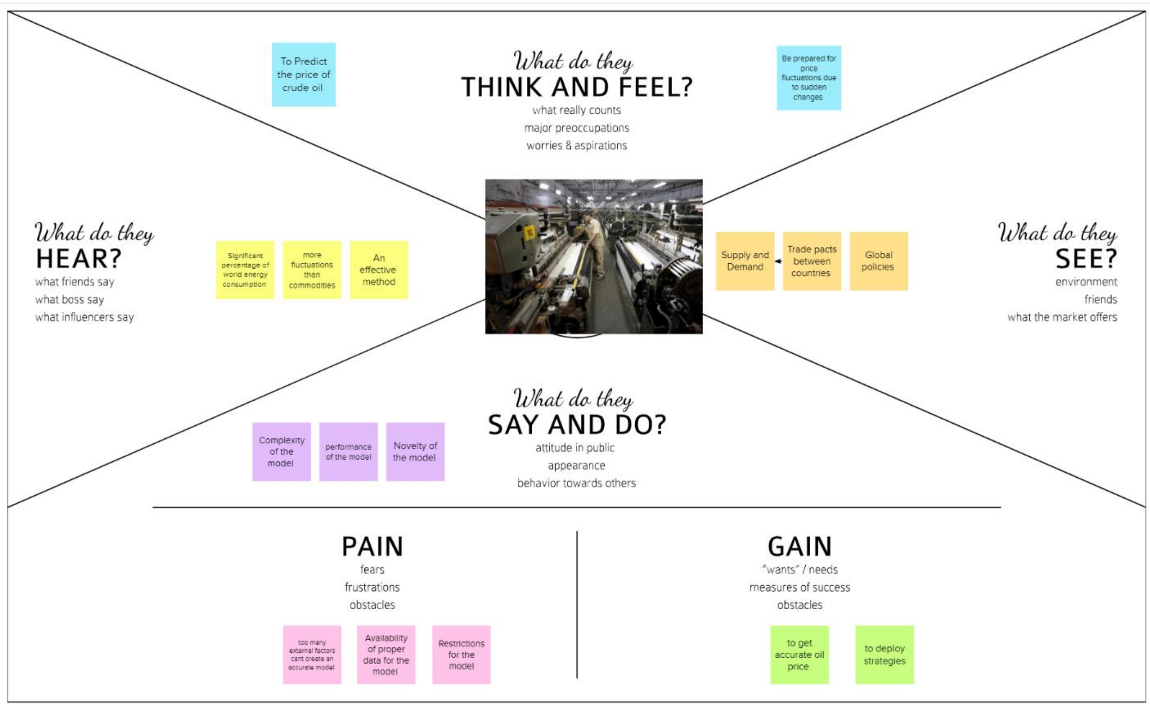
S. NO	**Title**	**Authors**	**Publication Date**	**Methodology**	**Merits**	**Demerits**
1	Forecasting Model for Crude Oil Price Using Artificial Neural Networks	Siddhivinayak Kulkarni Imad Haidar	2009	Artificial Neural Network, Deep Learning	The LSTM layers result in more accurate results.	Crude oil price signals exhibit highly nonlinear and complex behavior.

2.	Crude oil prices and volatility prediction by a hybrid model based on kernel extreme learning machine	Hongli Niu and Yazhi Zhao	17 September 2021	VMD-KELM	The VMD-KELM model shows a more powerful ability than other models in improving the precision of forecasting crude oil volatility.	-
3.	Crude oil price prediction using ANN	Nalini Gupta and Shobhit Nigam	January 2020	Artificial Neural Network	ANN model is effective. This capture the changing pattern of prices. Prediction is accurate.	Market trends have to be planned, then the ANN model will perform.
4.	Crude oil price prediction using complex network and deep learning algorithms	Makumbonori Bristone, Rajesh Prasad, Adamu Ali Abubakar	19 June 2019	Artificial Neural Network, Deep Learning	The appropriate number of LSTM layers can effectively improve the model.	The other factors that affect the crude oil price volatilities such as economic growth, exchange rate demand are not considered.

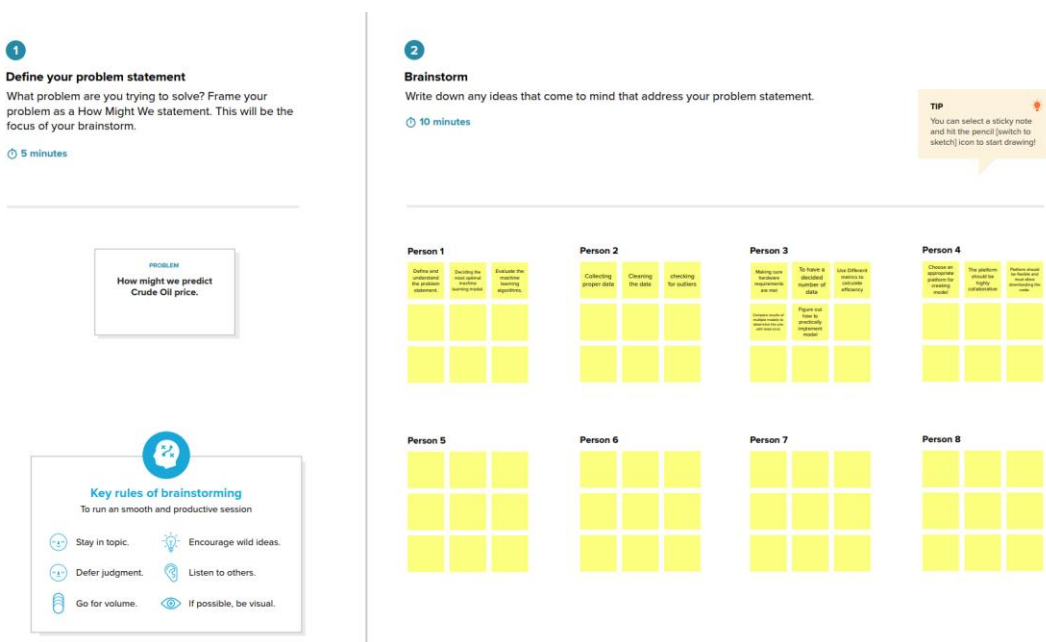
5.	Daily crude oil price forecasting using Hybridizing wavelet and Artificial Neural Network Model	Ani Shabri and Ruhaidah Samsudin	16 July 2014	Artificial Neural Network	The hybrid model showed a great improvement in crude oil price modeling and produced better forecasts than ANN model alone.	-
6.	Understanding crude oil prices	James D. Hamilton NBER	-	Analysis	Topics discussed include the role of commodity speculation, OPEC, and resource depletion	-
7.	A novel look back N feature approach towards prediction of crude oil price	Rudra Kalyan Nayak	-	ARIMA, LBNF Algorithm	Attained better training and accuracy by shifting the dataset into n class problem and more scope to classifier.	-

3. IDEATION AND PROPOSED SOLUTION-

3.1 EMPATHY MAP AND CANVAS



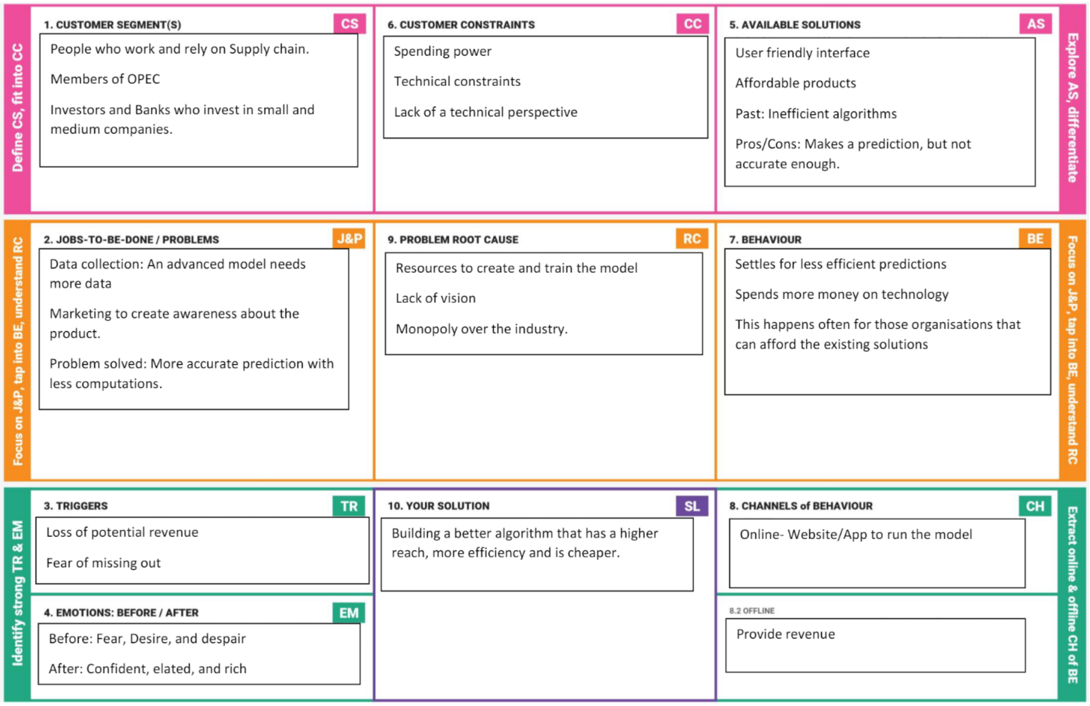
3.2 IDEATION AND BRAINSTORMING



3.2 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	This Guided Project mainly focuses on applying Neural Networks to predict the Crude Oil Price. This decision helps us to buy crude oil at the proper time. Time series analysis is the best option for this kind of prediction because we are using the Previous history of crude oil prices to predict future crude oil.
2.	Idea / Solution description	A data driven approach to predict crude oil prices.
3.	Novelty / Uniqueness	Considering outside variables which will affect the prices like natural disasters . Building a application for easier use
4.	Social Impact / Customer Satisfaction	Stabilizes the economy. Will help businesses predict the fuel prices and be prepared for uncertainties.
5.	Business Model (Revenue Model)	Any and every business with a supply chain currently relies on crude oil for transportation. A proper prediction of the crude oil prices can bring profits and order to a lot of businesses. Hence they will be our target consumers.
6.	Scalability of the Solution	The final solution will be a web application, Hence can be easily adapted to the organisation that uses this software.

3.4 PROBLEM SOLUTION FIT



4 REQUIREMENT ANALYSIS:

4.1 FUNCTIONAL REQUIREMENTS

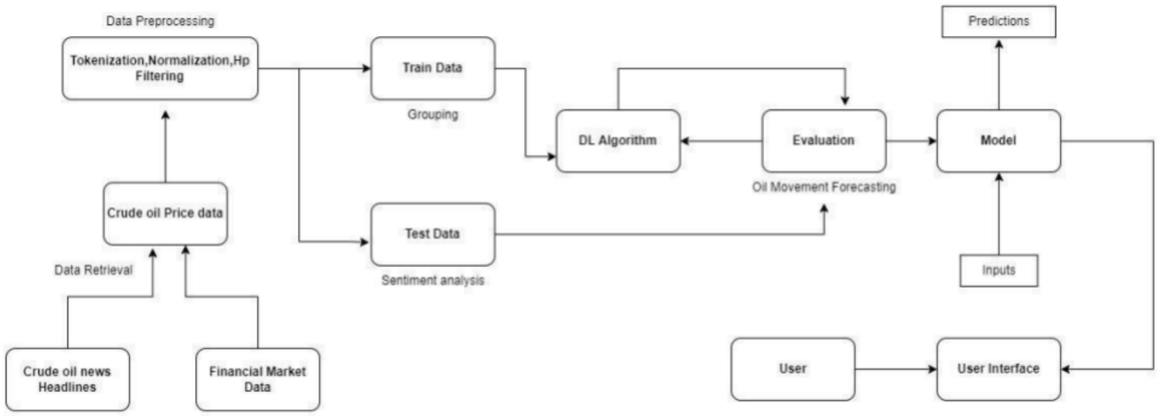
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Support	Provide answers to user’s queries.
FR-4	News	Current news related to crude oil will be shared with users.
FR-5	Notification	Notification will be sent for price alert to users.
FR-6	Database	User’s information will be stored in database.

4.2 NON -FUNCTIONAL REQUIREMENTS

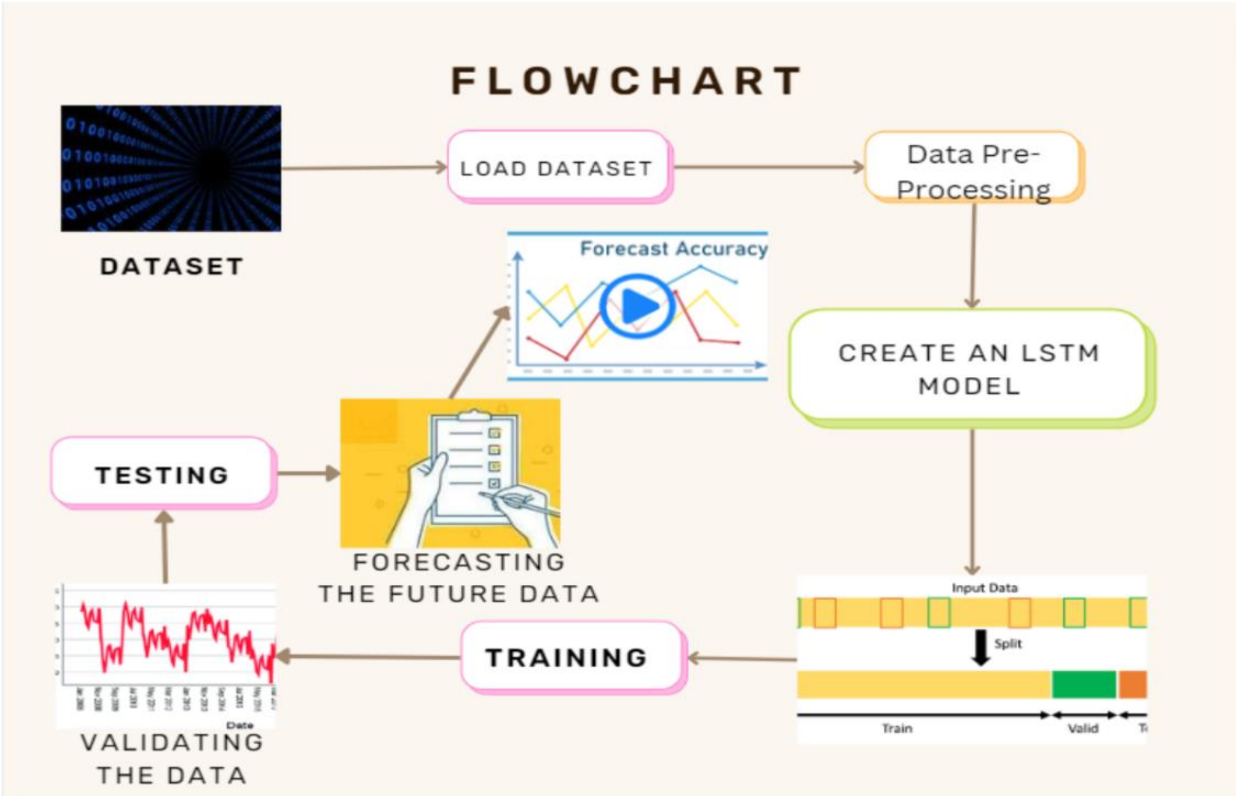
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	It can use by wide variety of client as it is very simple to learn and not complex to proceed.
NFR-2	**Security**	We are using login for the user thus; it will be very secure to use.
NFR-3	**Reliability**	It will be reliable that it can update with very time period so that the accuracy will be good.
NFR-4	**Performance**	It can perform fast even at a lower bandwidth.
NFR-5	**Availability**	Prediction will be available for every user but only for premium users some additional information alerts will be sent.
NFR-6	**Scalability**	It is scalable that we are going to use data in kb so that the quite amount of storage is satisfied.

5.PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS



5.2 SOLUTION AND TECHNICAL ARCHITECTURE



5.3 COMPONENTS AND TECHNOLOGIES

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application; Web UI, Mobile App.	HTML, CSS, JavaScript / Angular JS
2.	Application Logic-1	Logic for a process in the application	Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson Assistant
4.	Web Application	For web app	Python (Flask), Streamlit
5.	Database	Data Type, Configurations etc.	MySQL,
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Local Filesystem
8.	External API-1	Purpose of External API used in the application	Firebase
9.	Machine Learning Model	Purpose of Machine Learning Model	RNN, LSTM
10.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local, Firebase, Kubernetes

S.No	Characteristics	Description	Technology
	Open-Source Frameworks	Flask	Web Application
	Security Implementations	OAuth 2.0 Authentication	Authentication is provided by Google orFacebook or any available providers
	Scalable Architecture	Microservices	AWS Lambda
	Availability	Distributed servers	CDN
	Performance	Handle more than1000 users at a time(in server)	Flask

5.4 APPLICATION AND CHARACTERISTICS

5.5 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-3
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
Customer (Web user)			I can view the prices of crude oil .	I will be able to see the prices.	High	Sprint-2
Customer Care Executive			Will provide solutions and guidance to queries raised by users.	Helps to solve issues raised by users.	medium	Sprint-4
Administrator			It will display ,control access of results and store the results .	Displays and stores the results .	high	Sprint-4

6. PROJECT PLANNING & SCHEDULING

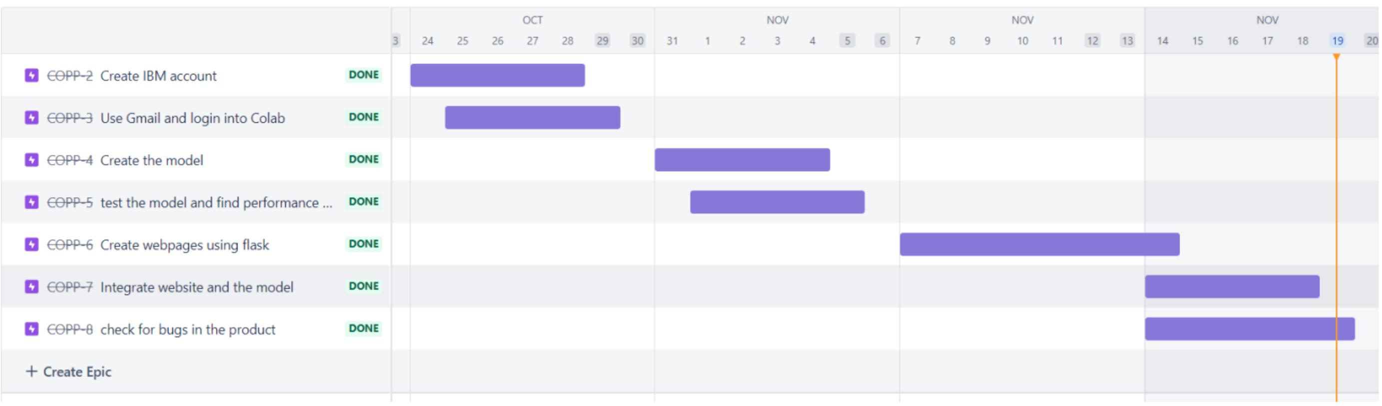
6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint 1	Google account	USN 1	A google account is used to log into google drive. This will be used to store datasets	10	High	1
Sprint 1	Google Colab	USN 2	The same google account is used to log into colab.	10	High	1
Sprint 2	ML modules	USN 3	Create the model using the train dataset	20	High	2
Sprint 2	ML modules	USN 4	Calculate the performance metrics and accuracy	10	Medium	2
Sprint 3		USN 5	Code the first webpage using flask	10	High	2
Sprint 3		USN 6	Code the second webpage using flask	10	High	2
Sprint 4		USN 7	Integrate the websites and the model	20	High	3
Sprint 4		USN 8	Check the final product for bugs	10	Low	1

6.2. Sprint Delivery Schedule

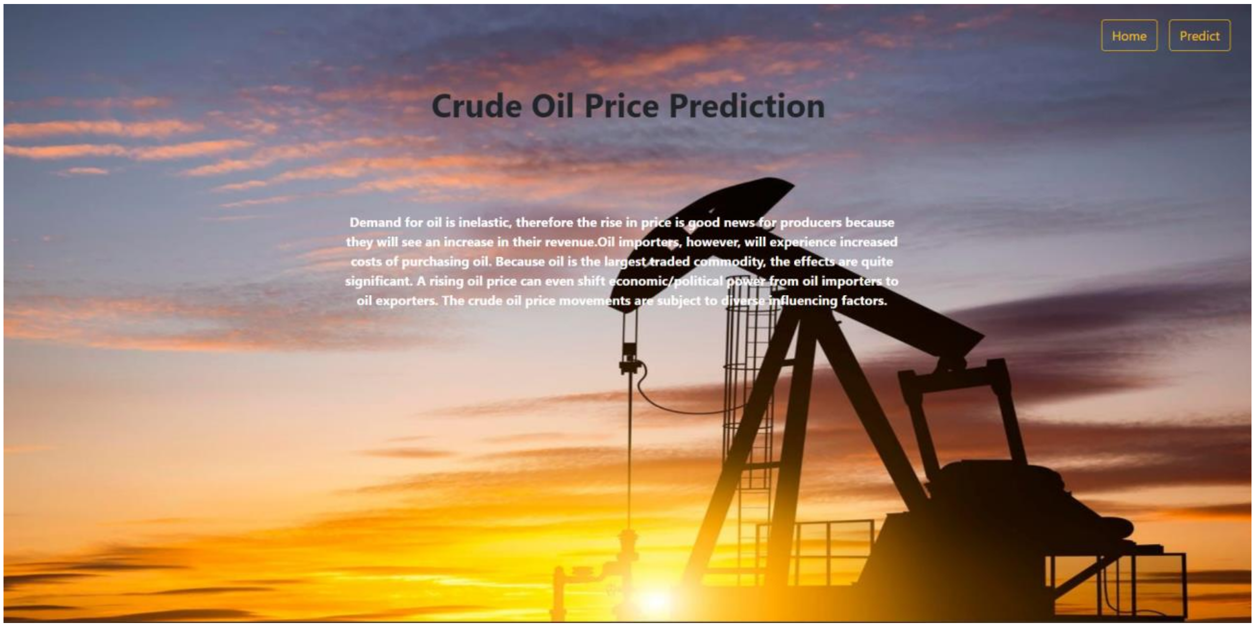
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	30	6 Days	31 Oct 2022	05 Nov 2022	30	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	14 Nov 2022
Sprint-4	30	6 Days	14 Nov 2022	19 Nov 2022	30	19 Nov 2022

6.3 Reports from JIRA

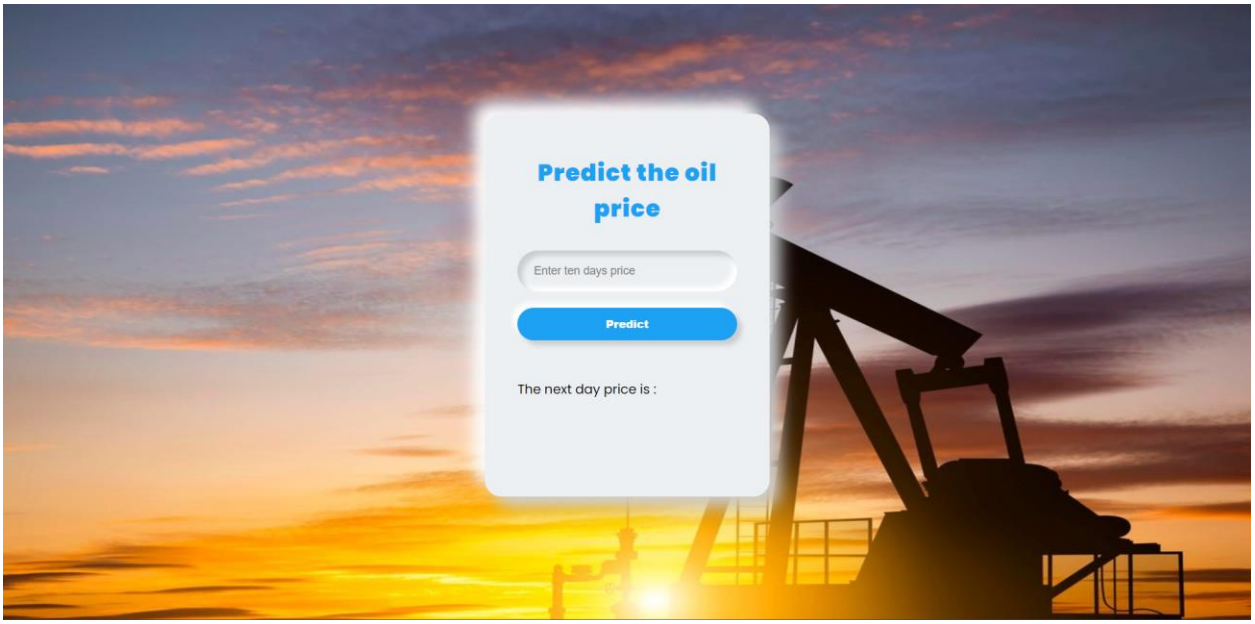


7. CODING & SOLUTIONING

7.1 Feature 1



7.2 Feature 2



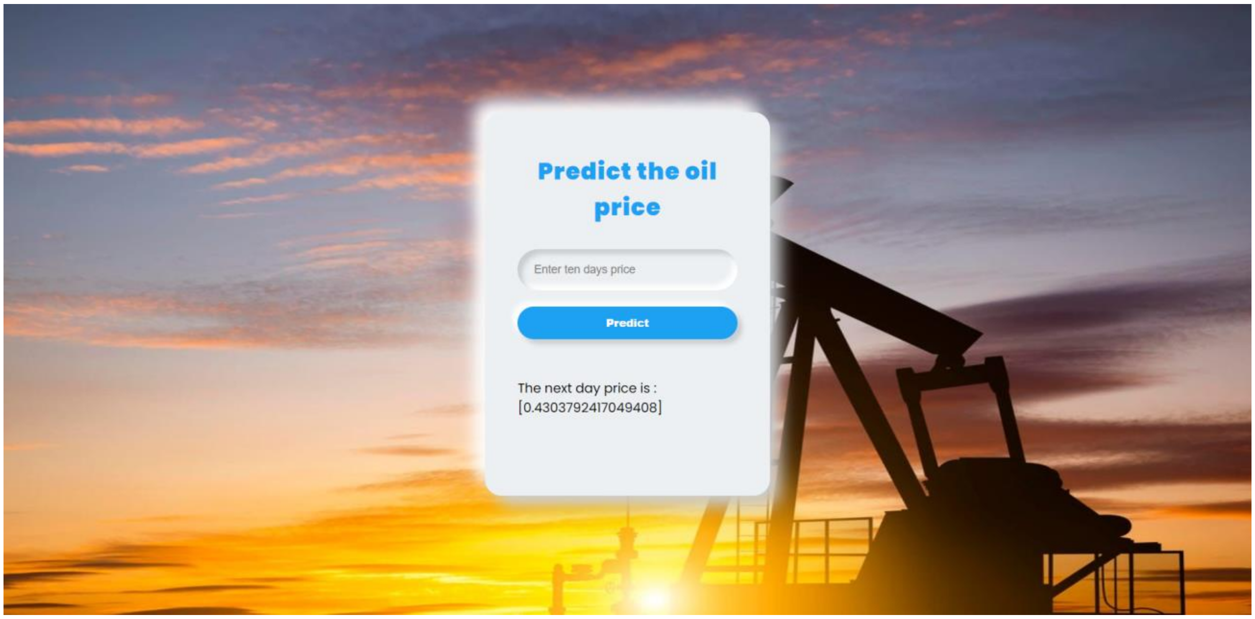
8. Testing

8.1 Test Cases

Test Case 1:

Input 1: 0.212,0.45,0.4657,0.236,0.344,0.9876,0.222,0.774,0.456,0.753

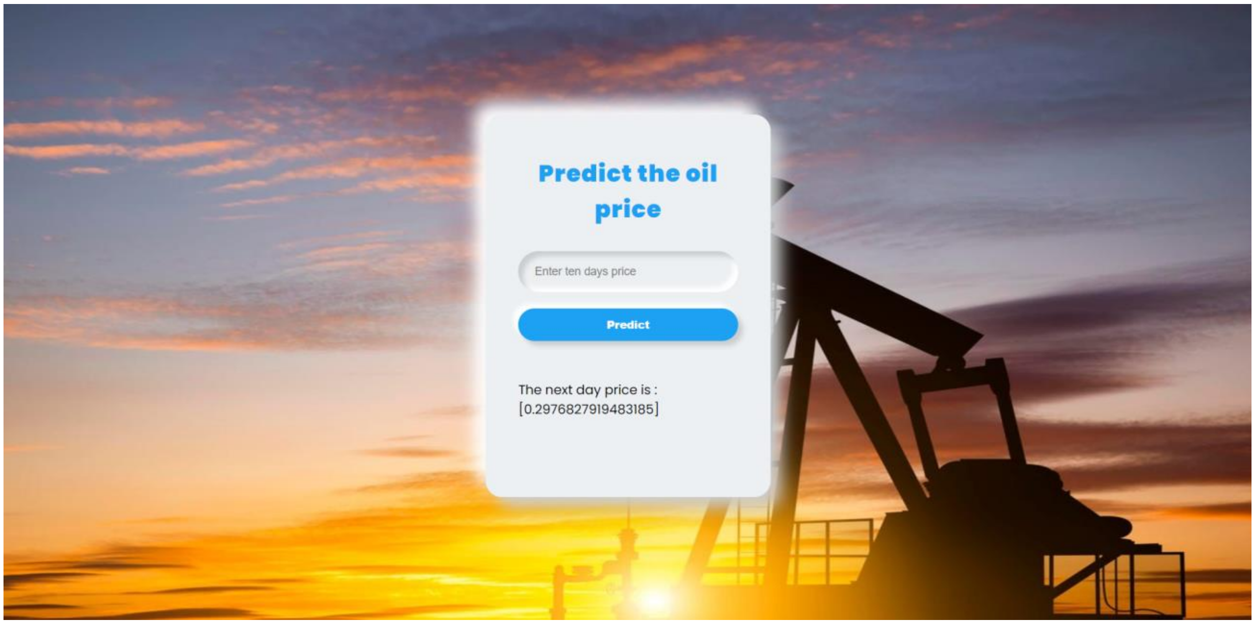
Output:



Test Case 2:

Input 2: 0.4,0.5,0.4,0.4,0.4,0.4,0.3,0.4,0.2,0.5

Output:



8.2 User Acceptance Testing

Acceptance Testing
UAT Execution & Report Submission

Date	03 November 2022
Team ID	PNT2022TMID52922
Project Name	Project – crude oil price prediction
Maximum Marks	4 Marks

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	0	0	0	`1	1
Duplicate	0	0	0	0	0
External	0	0	0	1	1
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	0	0	1	1
Totals	11	2	6	24	43

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

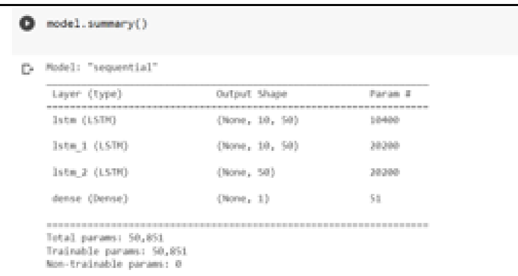
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

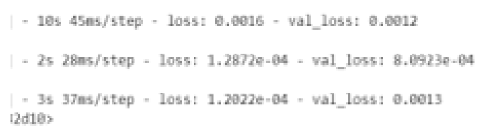
9. RESULT

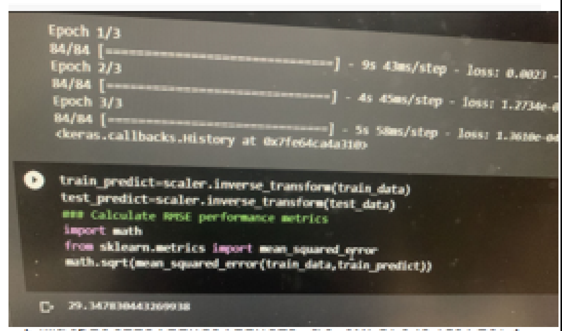
9.1 Performance Metrics:

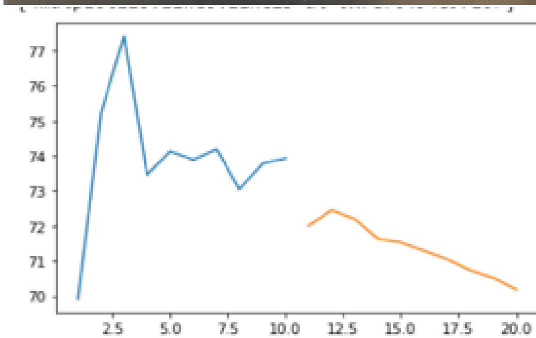
Date	17 November 2022
Team ID	PNT2022TMID52922
Project Name	Crude Oil Price Prediction

Model Performance Testing:

<table>
<tr><th>S.No.</th><th>Parameter</th><th>Values</th><th>Screenshots</th></tr>
<tr><td>1.</td><td>Model Summary</td><td></td><td>
</td></tr>
<tr><td>2.</td><td>Accuracy</td><td>Training Accuracy - 0.99374382493

Validation Accuracy – 2.201959455277266</td><td>




</td></tr>
</table>

10. ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

- not complicated (web app works to the point)
- quicker reaction (less latency)
- readily scalable
- easy to use interface
- Low in weight

DISADVANTAGES:

- Flask is limited to handling smaller applications.
- Even if the user may obtain the result quickly, the requirement to enter the last 10 days' worth of crude oil price values may be uncomfortable for the end users.
- Cost of maintenance

CONCLUSION:

The online app will provide the finest service to end customers (mostly investors) who are experts in investing and just seeking the crude oil price for their advantage, as the web app will not keep them waiting and will respond quickly. Our Web app demonstrates that our model obtains the greatest accuracy in terms of mean squared prediction error and directional accuracy ratio over a wide range of forecast time horizons. The LSTM model is employed in our web app (Long Short Term Memory). In comparison to other algorithms such as RTRN, BPTT, and RCC, LSTM produces more successful runs and learns significantly faster. LSTM can also tackle difficult problems. With LSTM, the difficulty of updating each weight is decreased to O (1), which is an extra benefit. The LSTM cell improves long-term memory performance by allowing for the learning of additional parameters. This makes it the most powerful RNN (Recurrent Neural Network) for predicting, particularly when your data has a longer-term trend. IBM Watson provides a platform for collaborative work and making data and model training easier. With Machine Learning, IBM Watson provides one-click deployment.

Immediate response:

End customers do not have to wait any longer than necessary. End consumers will find what they are seeking in a matter of seconds. There is no delay.

Scalability:

Because our web software is hosted on the IBM cloud, the code can be simply managed and deployed. Our website can be scaled quickly if the number of users grows.

12. FUTURE SCOPE

Long-term oil price predictions are nevertheless critical to the oil investment market since the commodity, albeit volatile, is frequently traded over longer periods. Oil is a commodity that is still in high demand and is scarce in supply, thus it is expected to grow in demand over time. Moreover, the long-term projection of oil prices is crucial to several groups in the economy.

The average prediction accuracy of the LSTM model was 66.67% (33.33%) and 439587 (673.8) times greater than that of the ANN and ARIMA models, respectively. As a result, we may infer that the LSTM model can increase short-term predicting accuracy for both types of pricing.

Because crude oil is still in great demand, the number of investors in it continues to grow. When children are interested in investing but are at an early level, our web software provides a better service by offering daily estimations to help them understand the pattern.

Because the price of crude oil is a complicated dynamic pattern, and new prices appear regularly, it is critical to update our model. Because we employ an LSTM model, the complexity is decreased to O(1), and it takes less time to retrain the model with fresh data, allowing us to serve clients more accurately.

13. Appendix

Source Code:

ML MODEL

Type *Markdown* and LaTeX: α^2

```
In [ ]: # import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt

In [2]: data = pd.read_excel("C:/Users/Administrator/Downloads/Crude Oil Prices Daily.xlsx")
        data.head()

Out[2]:
```

	Date	Closing Value
0	1986-01-02	25.56
1	1986-01-03	26.00
2	1986-01-06	26.53
3	1986-01-07	25.85
4	1986-01-08	25.87

```
In [3]: data.isnull().any()

Out[3]: Date             False
        Closing Value     True
        dtype: bool

In [4]: data.dropna(axis=0,inplace=True)

In [5]: data.isnull().sum()
```

In [6]:
```
data_oil=data.reset_index()['Closing Value']
data_oil
```

Out[6]:
```
0       25.56
1       26.00
2       26.53
3       25.85
4       25.87
        ...
8211    73.89
8212    74.19
8213    73.05
8214    73.78
8215    73.93
Name: Closing Value, Length: 8216, dtype: float64
```

In [7]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler(feature_range=(0,1))
data_oil=scaler.fit_transform(np.array(data_oil).reshape(-1,1))
```

In [8]:
```
data_oil
```

Out[8]:
```
array([[0.11335703],
       [0.11661484],
       [0.12053902],
       ...,
       [0.46497853],
       [0.47038353],
       [0.47149415]])
```

In [9]:
```
plt.plot(data_oil)
```

In [10]:
```
training_size=int(len(data_oil)*0.65)
test_size=len(data_oil)-training_size
train_data,test_data=data_oil[0:training_size,:],data_oil[training_size:len(data_oil),:1]
```

In [11]:
```
training_size,test_size
```

Out[11]:
```
(5340, 2876)
```

In [12]:
```
train_data.shape
```

Out[12]:
```
(5340, 1)
```

In [13]:
```
def create_dataset(dataset,time_step=1):
  dataX,dataY=[],[]
  for i in range(len(dataset)-time_step-1):
    a=dataset[i:(i+time_step),0]
    dataX.append(a)
    dataY.append(dataset[i+time_step,0])
  return np.array(dataX),np.array(dataY)
```

In [14]:
```
time_step=10
x_train,y_train=create_dataset(train_data,time_step)
x_test,y_test=create_dataset(test_data,time_step)
```

In [15]:
```
print(x_train.shape),print(y_train.shape)
```

```
(5329, 10)
(5329,)
```

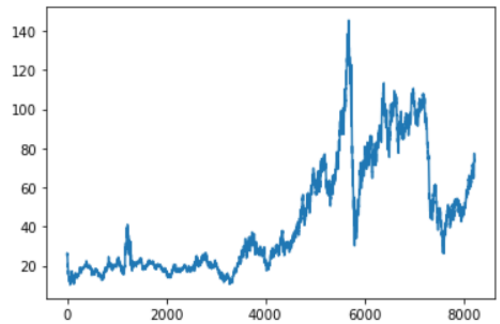
Out[15]:
```
(None, None)
```

In [16]:
```
print(x_test.shape),print(y_test.shape)
```
```
(2865, 10)
(2865,)
```
Out[16]:
```
(None, None)
```
In [17]:
```
x_train
```
Out[17]:
```
array([[0.11335703, 0.11661484, 0.12053902, ..., 0.10980305, 0.1089886 ,
        0.11054346],
       [0.11661484, 0.12053902, 0.11550422, ..., 0.1089886 , 0.11054346,
        0.10165852],
       [0.12053902, 0.11550422, 0.1156523 , ..., 0.11054346, 0.10165852,
        0.09906708],
       ...,
       [0.36731823, 0.35176958, 0.36080261, ..., 0.36391234, 0.37042796,
        0.37042796],
       [0.35176958, 0.36080261, 0.35354657, ..., 0.37042796, 0.37042796,
        0.37879461],
       [0.36080261, 0.35354657, 0.35295424, ..., 0.37042796, 0.37879461,
        0.37916482]])
```
In [18]:
```
x_train=x_train.reshape(x_train.shape[0],x_train.shape[1],1)
x_test=x_test.reshape(x_test.shape[0],x_test.shape[1],1)
```
In [19]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
```
In [20]:
```
model=Sequential()
```
In [21]:
```
model.add(LSTM(50,return_sequences=True,input_shape=(10,1)))
model.add(LSTM(50,return_sequences=True))
model.add(LSTM(50))
```
In [22]:
```
model.add(Dense(1))
```
In [23]:
```
model.summary()
```
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 10, 50)            10400

 lstm_1 (LSTM)               (None, 10, 50)            20200

 lstm_2 (LSTM)               (None, 50)                20200

 dense (Dense)               (None, 1)                 51

=================================================================
Total params: 50,851
Trainable params: 50,851
Non-trainable params: 0
_________________________________________________________________
```
In [24]:
```
model.compile(loss='mean_squared_error',optimizer='adam')
```
In [25]:
```
model.fit(x_train,y_train,validation_data=(x_test,y_test),epochs=3,batch_size=64,verbose=1)
```
```
Epoch 1/3
84/84 [==============================] - 20s 96ms/step - loss: 0.0022 - val_loss: 0.0016
Epoch 2/3
84/84 [==============================] - 6s 68ms/step - loss: 1.2947e-04 - val_loss: 8.2107e-04
Epoch 3/3
84/84 [==============================] - 5s 65ms/step - loss: 1.2985e-04 - val_loss: 0.0014
```
Out[25]:
```
<keras.callbacks.History at 0x270f4dad0d0>
```

```python
from tensorflow.keras.models import load_model
```

```python
model.save("crude_oil.h5")
```

```python
look_back=10
trainpredictPlot = np.empty_like(data_oil)
trainpredictPlot[:, :]= np.nan
trainpredictPlot[look_back:len(train_predict)+look_back, :] = train_predict
# shift test predictions for plotting
testPredictplot = np.empty_like(data_oil)
testPredictplot[:,: ] = np.nan
testPredictplot[look_back:len(test_predict)+look_back, :] = test_predict
# plot baseline and predictions
plt.plot(scaler.inverse_transform(data_oil))
plt.show()
```



n [32]:
```
temp_input=list(x_input)
temp_input=temp_input[0].tolist()
```

n [33]:
```
temp_input
```

ut[33]:
```
[0.44172960165852215,
 0.48111950244335855,
 0.49726047682511476,
 0.4679401747371539,
 0.4729749740855915,
 0.47119798608026064,
 0.47341922108692425,
 0.4649785280616022,
 0.4703835332444839,
 0.47149415074781587]
```

n [34]:
```
lst_output=[]
n_steps=10
i=0
while(i<10):
    if(len(temp_input)>10):
#print(temp_input)
       x_input=np.array(temp_input[1:])
       print("{} day input {}".format(i,x_input))
       x_input=x_input.reshape(1,-1)
       x_input = x_input.reshape((1, n_steps, 1)) #print(x_input)
       yhat = model.predict(x_input, verbose=0)
       print("{} day output {}".format(i,yhat))
       temp_input.extend(yhat[0].tolist())
       temp_input=temp_input[1:] #print(temp_input)
       lst_output.extend(yhat.tolist())
       i=i+1
    else:
       x_input = x_input.reshape((1, n_steps,1))
```

```
        i=i+1
    else:
        x_input = x_input.reshape((1, n_steps,1))
        yhat = model.predict(x_input, verbose=0)
        print(yhat[0])
        temp_input.extend(yhat[0].tolist())
        print(len(temp_input))
        lst_output.extend(yhat.tolist())
        i=i+1
```

```
[0.45455453]
11
1 day input [0.4811195  0.49726048 0.46794017 0.47297497 0.47119799 0.47341922
 0.46497853 0.47038353 0.47149415 0.45455453]
1 day output [[0.4580783]]
2 day input [0.49726048 0.46794017 0.47297497 0.47119799 0.47341922 0.46497853
 0.47038353 0.47149415 0.45455453 0.45807829]
2 day output [[0.45606512]]
3 day input [0.46794017 0.47297497 0.47119799 0.47341922 0.46497853 0.47038353
 0.47149415 0.45455453 0.45807829 0.45606512]
3 day output [[0.4516586]]
4 day input [0.47297497 0.47119799 0.47341922 0.46497853 0.47038353 0.47149415
 0.45455453 0.45807829 0.45606512 0.45165861]
4 day output [[0.45071504]]
5 day input [0.47119799 0.47341922 0.46497853 0.47038353 0.47149415 0.45455453
 0.45807829 0.45606512 0.45165861 0.45071504]
5 day output [[0.4487366]]
6 day input [0.47341922 0.46497853 0.47038353 0.47149415 0.45455453 0.45807829
 0.45606512 0.45165861 0.45071504 0.44873661]
6 day output [[0.4466326]]
7 day input [0.46497853 0.47038353 0.47149415 0.45455453 0.45807829 0.45606512
 0.45165861 0.45071504 0.44873661 0.44663259]
7 day output [[0.4439124]]
8 day input [0.47038353 0.47149415 0.45455453 0.45807829 0.45606512 0.45165861
 0.45071504 0.44873661 0.44663259 0.44391239]
8 day output [[0.4419663]]
9 day input [0.47149415 0.45455453 0.45807829 0.45606512 0.45165861 0.45071504
```

HTMl - home page:

```
<!DOCTYPE html>

<html lang="en">

<head>

    <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>Document</title>

    <link
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.cs
s"                    rel="stylesheet"                    integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
```

```
    <style>

.home_form{
    position: absolute;
    top: 20px;
    right: 20px;
}

.dashboard{
    margin: 0 10px 0 10px;
}

h1{
    position: relative;
    top: 100px;
    align-items: center;
    align-self: center;
    text-align: center;
    font-weight: 700;
}

div{
    position: relative;
    left: 410px;
```

```
    width: 700px;

    position: relative;

    top: 200px;

    text-align: center;

    color: white;

    font-weight:700;

}



*{

    margin: 0;

    padding: 0;

    background-image: url("static/css/image.jpeg");

    background-repeat: no-repeat;

    background-attachment: fixed;

    background-size: 100% 100%;

}



    </style>

</head>

<body>

    <form class="home_form">

        <a href="#"><button type="button" class="btn btn-outline-warning">Home</button></a>

        <a href="predict.html"><button type="button" class="btn btn-outline-warning dashboard">Predict</button></a>
```

```
    </form>

    <h1>Crude Oil Price Prediction</h1>

    <div>

        <p>

            Demand for oil is inelastic, therefore the rise in price is good news for producers because they will see an increase in their revenue.Oil importers, however, will experience increased costs of purchasing oil. Because oil is the largest traded commodity, the effects are quite significant. A rising oil price can even shift economic/political power from oil importers to oil exporters. The crude oil price movements are subject to diverse influencing factors.

        </p>

    </div>

</body>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-datepicker/1.7.1/js/bootstrap-datepicker.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>

</html>
```

Predict page:

```
<html>

    <head>
```

```
        <link rel="stylesheet" href="static/css/style.css">

        <style>

            body {

              background-image: url('static/css/image.jpeg');

              background-repeat: no-repeat;

              background-attachment: fixed;

              background-size: 100% 100%;

            }

            </style>


    </head>

    <script>

        document.getElementByID("demo").innerHTML                    =
document.getElementById("ten");

    </script>

<body>

<form action="/method" method="POST" enctype = "multipart/form-data">

<div class="container">

    <!--<div class="brand-logo"></div>-->

    <div class="brand-title">Predict the oil price</div>

    <div class="inputs">

      <!-- <label>Enter Price</label> -->

      <input  type="text"  placeholder="Enter  ten  days  price"  id="ten"
name="val"/>

      <button type="submit">Predict</button><br><br>
```

```
        The next day price is :  {{prediction}}

    </div>

  </div>

</form>

</body>

</html>
```

CSS:

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;900&display
=swap');


input {

  caret-color: red;

}


body {

  margin: 0;

  width: 100vw;

  height: 100vh;

  background: #ecf0f3;

  display: flex;

  align-items: center;

  text-align: center;

  justify-content: center;

  place-items: center;
```

```
  overflow: hidden;

  font-family: poppins;

}



.container {

  position: relative;

  width: 350px;

  height: 470px;

  border-radius: 20px;

  padding: 40px;

  box-sizing: border-box;

  background: #ecf0f3;

  box-shadow: 14px 14px 20px #cbced1, -14px -14px 20px white;

}



.brand-logo {

  height: 100px;

  width: 100px;

  background:       url("https://img.icons8.com/color/100/000000/twitter--v2.png");

  margin: auto;

  border-radius: 50%;

  box-sizing: border-box;

  box-shadow: 7px 7px 10px #cbced1, -7px -7px 10px white;

}
```

```
.brand-title {
  margin-top: 10px;
  font-weight: 900;
  font-size: 1.8rem;
  color: #1DA1F2;
  letter-spacing: 1px;
}

.inputs {
  text-align: left;
  margin-top: 30px;
}

label, input, button {
  display: block;
  width: 100%;
  padding: 0;
  border: none;
  outline: none;
  box-sizing: border-box;
}

label {
```

```css
  margin-bottom: 4px;
}

label:nth-of-type(2) {
  margin-top: 12px;
}

input::placeholder {
  color: gray;
}

input {
  background: #ecf0f3;
  padding: 10px;
  padding-left: 20px;
  height: 50px;
  font-size: 14px;
  border-radius: 50px;
  box-shadow: inset 6px 6px 6px #cbced1, inset -6px -6px 6px white;
}

button {
  color: white;
  margin-top: 20px;
```

```
  background: #1DA1F2;

  height: 40px;

  border-radius: 20px;

  cursor: pointer;

  font-weight: 900;

  box-shadow: 6px 6px 6px #cbced1, -6px -6px 6px white;

  transition: 0.5s;

}


button:hover {

  box-shadow: none;

}


a {

  position: absolute;

  font-size: 8px;

  bottom: 4px;

  right: 4px;

  text-decoration: none;

  color: black;

  background: yellow;

  border-radius: 10px;

  padding: 2px;

}
```

```
h1 {
  position: absolute;
  top: 0;
  left: 0;
}
```

Python file(app.py):

```
from flask import Flask, render_template, request, redirect
#
import numpy as np


#
# from tensorflow.k
from keras.saving.save import load_model


app = Flask(__name__,template_folder='template')




@app.route('/', methods=["GET"])
def index():
    return render_template('index.html')




@app.route('/predict.html', methods=["POST", "GET"])
```

```python
@app.route('/method', methods=["POST", "GET"])

def method():

    if request.method == "POST":

        string = request.form['val']

        string = string.split(',')

        temp_input = [eval(i) for i in string]


        x_input = np.zeros(shape=(1, 10))

        x_input.shape


        lst_output = []

        n_steps = 10

        i = 0

        while (i < 10):

            if (len(temp_input) > 10):

                x_input = np.array(temp_input[1:])

                x_input = x_input.reshape(1, -1)

                x_input = x_input.reshape((1, n_steps, 1))

                yhat = model.predict(x_input, verbose=0)

                temp_input.extend(yhat[0].tolist())

                temp_input = temp_input[1:]

                lst_output.extend(yhat.tolist())

                i = i + 1
```

```
            else:

                x_input = x_input.reshape((1, n_steps, 1))

                yhat = model.predict(x_input, verbose=0)

                temp_input.extend(yhat[0].tolist())

                lst_output.extend(yhat.tolist())

                i = i + 1

        val = lst_output[9]

        return render_template('predict.html', prediction=val)

    if request.method == "GET":

        return render_template('predict.html')




if __name__ == "__main__":

    model = load_model(r'C:\Users\akash\Desktop\IBM-python app\IBM-Project-8197-1664354492\Project Development Phase\Sprint 3\crude_oil.h5')

    app.run(debug=True)
```

Github repository link:

https://github.com/IBM-EPBL/IBM-Project-19445-1659698005

Demo video link:

https://github.com/IBM-EPBL/IBM-Project-19445-1659698005/blob/main/Final%20Deliverables/project_demo_video.mp4