IBM-Project-38682-1660384421

PROJECT DOCUMENTATION

REPORT

CRUDE-OIL PRICE PREDICTION

TECHNOLOGY: ARTIFICIAL INTELLIGENCE

Team ID: PNT2022TMID41737

TEAM MEMBERS

Team Size: 4

Team Leader	: MOHAMED ASHRAF ALI M
Team member	: AKASH K
Team member	: KAMALESH P
Team member	: KALIDHAS S

INDEX

1. INTRODUCTION:

1.1 PROJECT OVERVIEW

This document is provided as a report for the project **Crude Oil Price Prediction.** Crude oil is amongst the most important resources in today's world, it is the chief fuel and its cost has a direct effect on the global habitat, our economy and oil exploration, exploitation and other activities. Prediction of oil prices has become the need of the hour, it is a boon to many large and small industries, individuals, the government. The evaporative nature of crude oil, its price prediction becomes extremely difficult and it is hard to be precise with the same. Several different factors that affect crude oil prices.

1.2 PURPOSE

Crude oil price fluctuations have a far-reaching impact on global economies and thus price forecasting can assist in minimizing the risks associated with volatility in oil prices. Price forecasts are very important to various stakeholders: governments, public and private enterprises, policymakers, and investors. According to economic theory, the price of crude oil should be easily predictable from the equilibrium between demand and supply, wherein demand forecasts are usually made from GDP, exchange rates and domestic prices, and supply is predicted from past production data and reserve data. Predicting demand for oil is usually straightforward, however supply is heavily affected by political activity such as cartelisation by OPEC to regulate prices, technological advances leading to the extraction of higher amounts of oil, and wars and other conflicts which can affect supply unpredictably.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

One of the most significant commodities in the world, crude oil is responsible for one-third of the world's energy use. It serves as the foundation for the majority of the items we use on a daily basis, ranging from plastics to transportation fuels. Since changes in the price of crude oil have a significant impact on national economies around the world, price forecasting can help reduce the risks brought on by oil price volatility. For a variety of stakeholders, including governments, public and private organizations, policymakers, and investors, price projections are crucial.

2.2 REFERENCES

- https://drive.google.com/drive/folders/1yq9UqoGpyAQFKR6ARNFwpV MoφΨτOHδXμ?υσπ=σηαρινγ

• Abdullah, S. N. and Zeng, X. (2010) ""Machine learning approach for crude oil price prediction with Artificial Neural Networks-Quantitative (ANN-Q) model"" Proceedings of the International Joint Conference on Neural Networks (IJCNN'2010).

• Agnolucci, P. (2009) ""Volatility in crude oil futures: A comparison of the predictive ability of GARCH and implied volatility models"" Energy Economics 31, 316-321.

• Alizadeh, A. and Mafinezhad, K (2010)""Monthly Brent Oil Price Forecasting Using Artificial Neural Networks and A Crisis Index"" Proceedings of the International Conference on Electronics and Information Engineering (ICEIE'2010), 2, 465-468.

• Aloui, C. Hamdi, M., Mensi, W. and Nguyen, D. Y. (2012) ""Further evidence on the time varying efficiency of crude oil markets"" Energy Studies Review 19(2), 38-51. Amano A. (1987) ""A Small Forecasting Model of the World Oil Market"" Journal of Policy Modeling 9(4), 615-635.

• Amin-Naseri, M. R. and Gharacheh, E. A. (2007) ""A hybrid artificial intelligence approach to monthly forecasting of crude oil price time series"" The Proceedings of the 10th International Conference on Engineering Applications of Neural Networks (CEANN'2007), 160-167.

• Barone-Adesi, G., Bourgoin, F. and Giannopoulos, K. (1998) ""Don"t Look Back"" Risk 11, 100-104. Beidas-Strom, S. and Pescatori, A. (2014) ""Oil price volatility and the role of speculation"" IMF working paper (WP/14/218).

• Bernabe, A., Martina, E., Alvarez-Ramirez, J. and Ibarra-Valdez, C. (2004) ""A multi-model approach for describing crude oil price dynamics"" Physica A: Statistical Mechanics and its Applications 338(3), 567-584.

• Blanchard, O. J. and Gali, J. (2007) ""The macroeconomic effects of oil shocks: Why are the 2000s so different from 1970s?"" NBER working paper number 13368.

• Cheong, C.W. (2009) ""Modelling and forecasting crude oil markets using ARCH-type models"" Energy Policy 37, 2346-2355.

• Dees, S., Karadeloglou, P., Kaufmann, R. K. and Sanchez, M. (2007) ""Modelling the world oil market: assessment of a quarterly econometric model"" Energy Policy 35, 178-191.

• Fattouh, B. (2012) ""Speculation and oil price formation"", Review of Environment, Energy and Economics (Re3), 1-5.

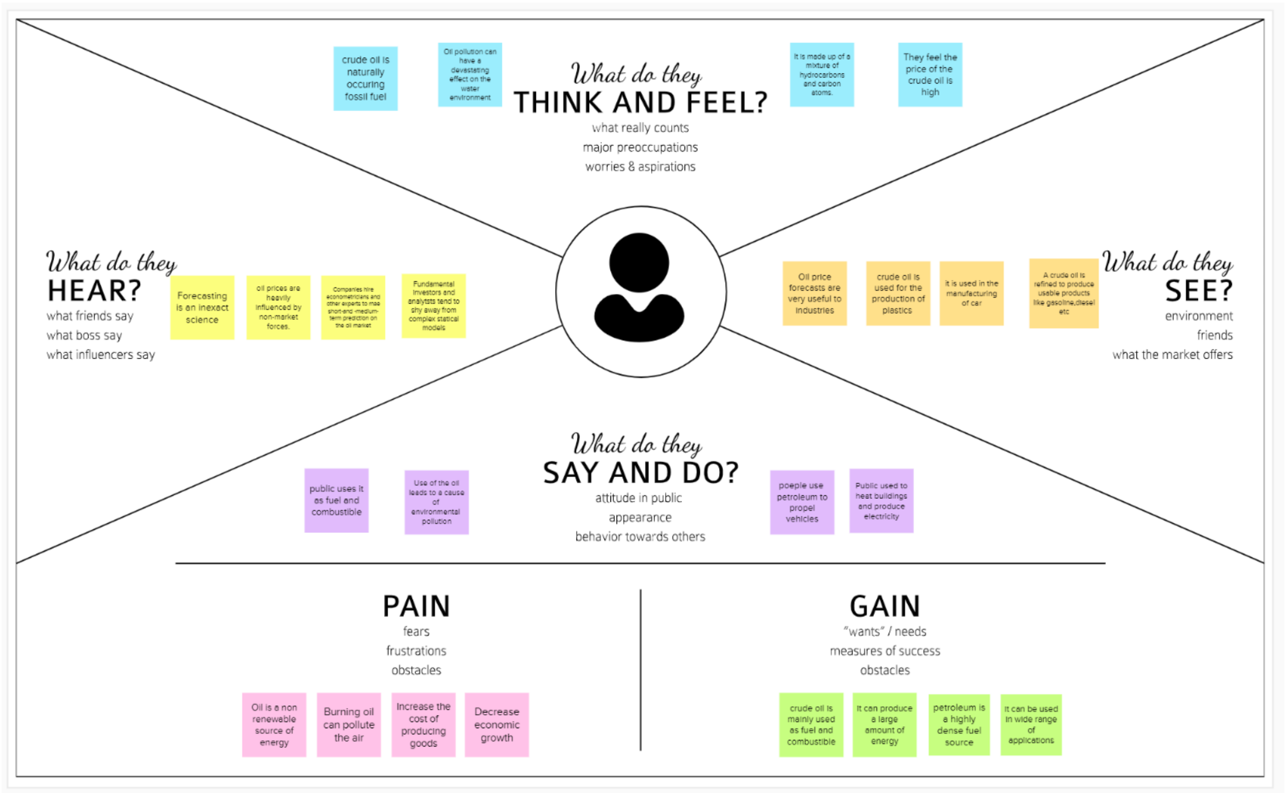
• Ghaffari, A. and Zare, S. (2009) ""A novel algorithm for prediction of crude oil price variation based on soft computing"" Energy Economics 31, 531-536.

2.3 PROBELM STATEMENT DEFINITION

It is required to forecast CRUDE OIL PRICE in international market. The input and output should also be shown as charts and/or dashboards in various formats (like day, week, work-week, month, quarter, year, etc.). The models should be built with comprehensive explanation of data (using EDA), trend analysis, assumptions, data cleaning and validation, data augmentation (if required). Performance of various models need to be clearly evaluated and best model needs to be recommended based on some robust evaluation criteria e.g., AIC(Akaike information criterion), Accuracy, RMSE, MSE etc.

3. IDEATION & PROPOSED SOLUTION:

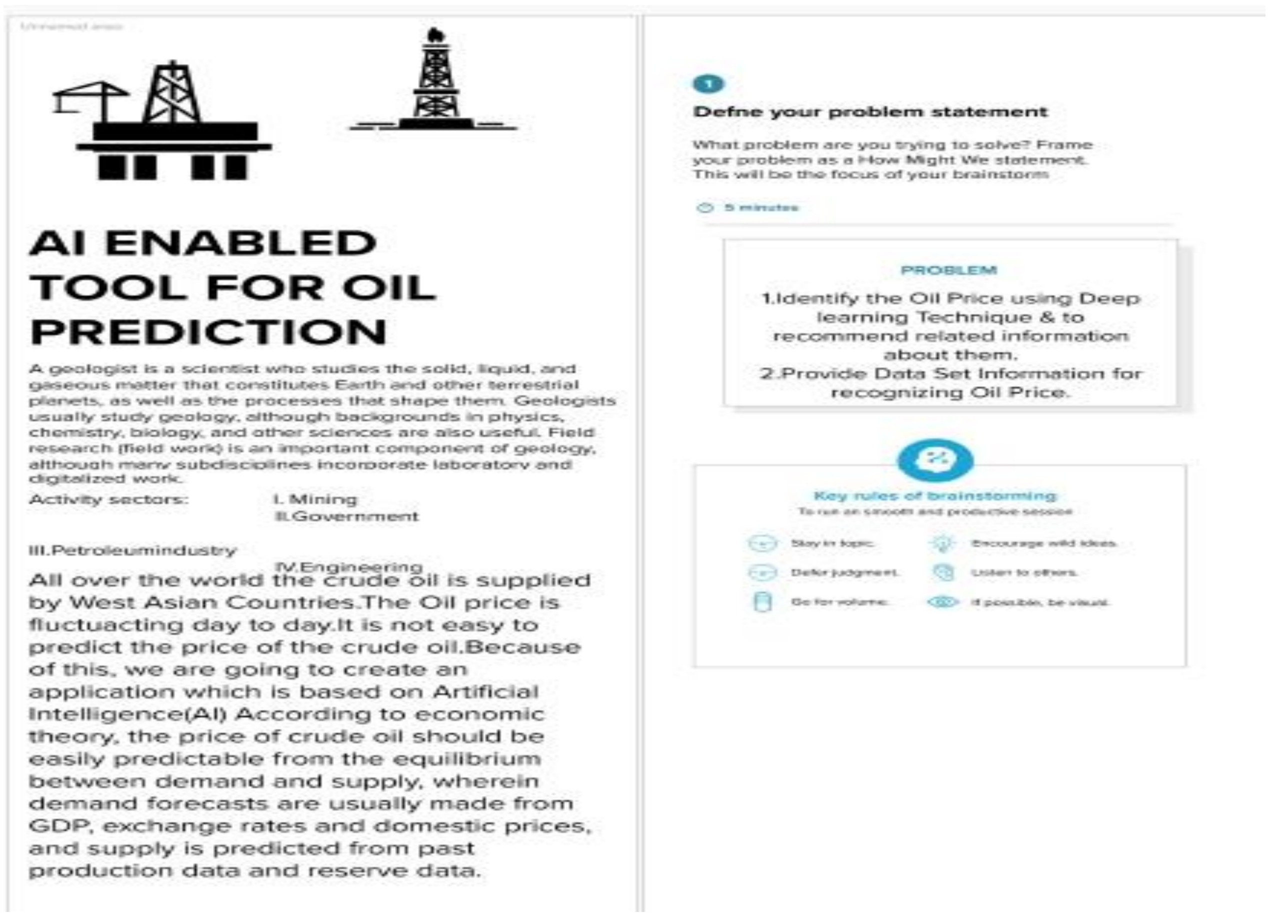
3.1 EMPATHY MAPS CANVAS


What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
crude oil is naturally occuring fossil fuel
Oil pollution can have a devastating effect on the water environment
It is made up of a mixture of hydrocarbons and carbon atoms.
They feel the price of the crude oil is high
What do they HEAR?
what friends say
what boss say
what influencers say
Forecasting is an inexact science
oil prices are heavily influenced by non-market forces.
Companies hire econometricians and other experts to mae short-and -medium-term prediction on the oil market
Fundamental investors and analytsts tend to shy away from complex statical models
What do they SEE?
environment
friends
what the market offers
Oil price forecasts are very useful to industries
crude oil is used for the production of plastics
it is used in the manufacturing of car
A crude oil is refined to produce usable products like gasoline,diesel etc
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
public uses it as fuel and combustible
Use of the oil leads to a cause of environmental pollution
poeple use petroleum to propel vehicles
Public used to heat buildings and produce electricity
PAIN
fears
frustrations
obstacles
Oil is a non renewable source of energy
Burning oil can pollute the air
Increase the cost of producing goods
Decrease economic growth
GAIN
"wants" / needs
measures of success
obstacles
crude oil is mainly used as fuel and combustible
It can produce a large amount of energy
petroleum is a highly dense fuel source
It can be used in wide range of applications

3.2 IDEATION AND BRAINSTORMING

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

Step-1: Team Gathering, Collaboration and Select the Problem Statement



AI ENABLED TOOL FOR OIL PREDICTION

A geologist is a scientist who studies the solid, liquid, and gaseous matter that constitutes Earth and other terrestrial planets, as well as the processes that shape them. Geologists usually study geology, although backgrounds in physics, chemistry, biology, and other sciences are also useful. Field research (field work) is an important component of geology, although many subdisciplines incorporate laboratory and digitalized work.

Activity sectors: I. Mining
II.Government
III.Petroleumindustry
IV.Engineering

All over the world the crude oil is supplied by West Asian Countries.The Oil price is fluctuacting day to day.It is not easy to predict the price of the crude oil.Because of this, we are going to create an application which is based on Artificial Intelligence(AI) According to economic theory, the price of crude oil should be easily predictable from the equilibrium between demand and supply, wherein demand forecasts are usually made from GDP, exchange rates and domestic prices, and supply is predicted from past production data and reserve data.

1

Defne your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm

5 minutes

PROBLEM

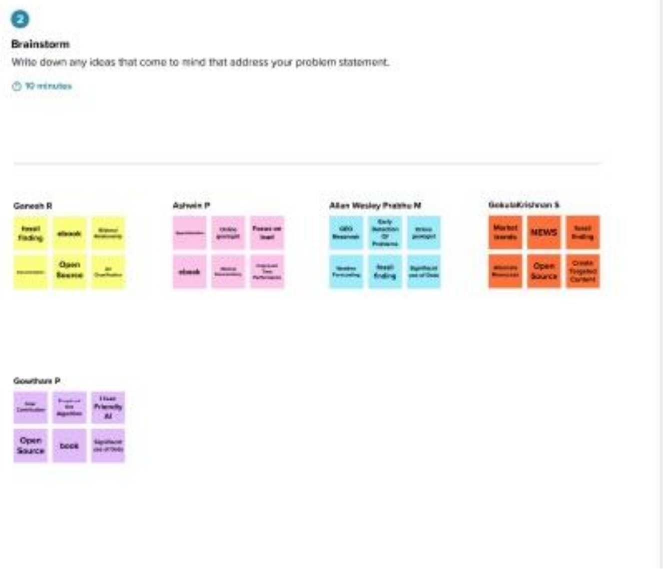
1.Identify the Oil Price using Deep learning Technique & to recommend related information about them.
2.Provide Data Set Information for recognizing Oil Price.

Key rules of brainstorming

To run an smooth and productive session

Stay in topic. Encourage wild ideas.
Defer judgment. Listen to others.
Go for volume. If possible, be visual.

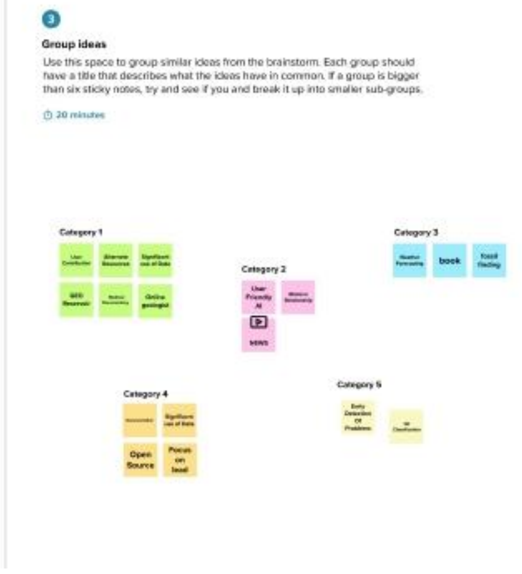
Step-2: Brainstorm, Idea Listing and Grouping



Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

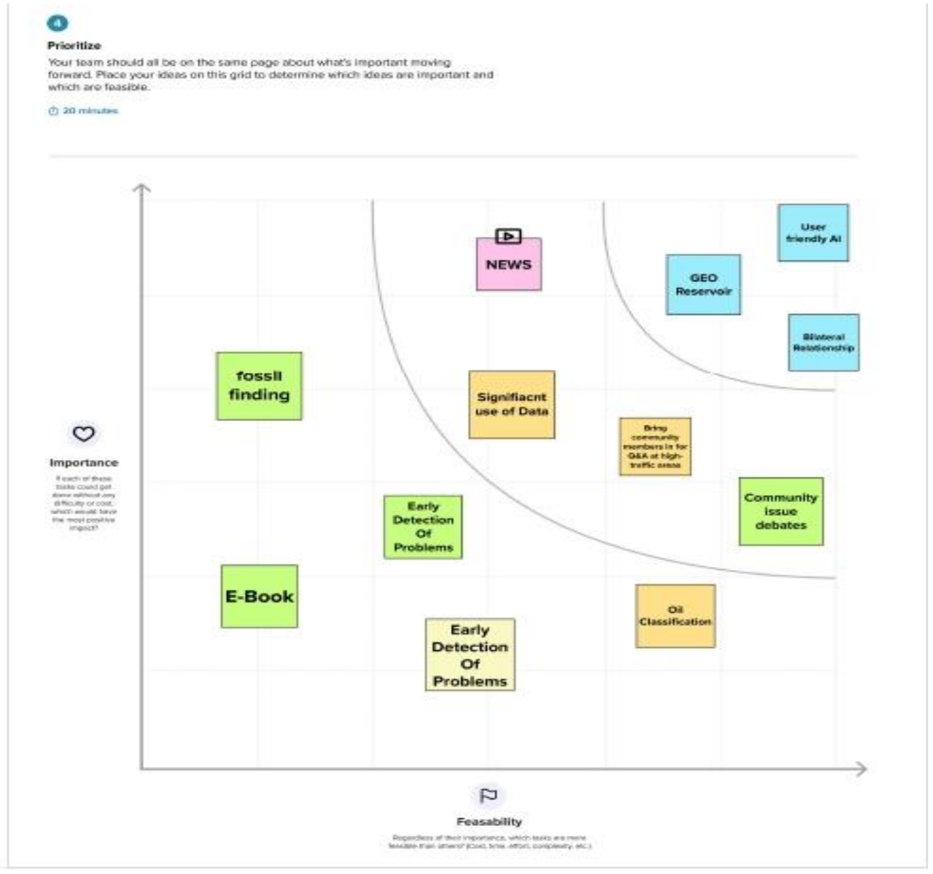


Group ideas

Use this space to group similar ideas from the brainstorm. Each group should have a title that describes what the ideas have in common. If a group is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

Step-3: Idea Prioritization



Prioritize

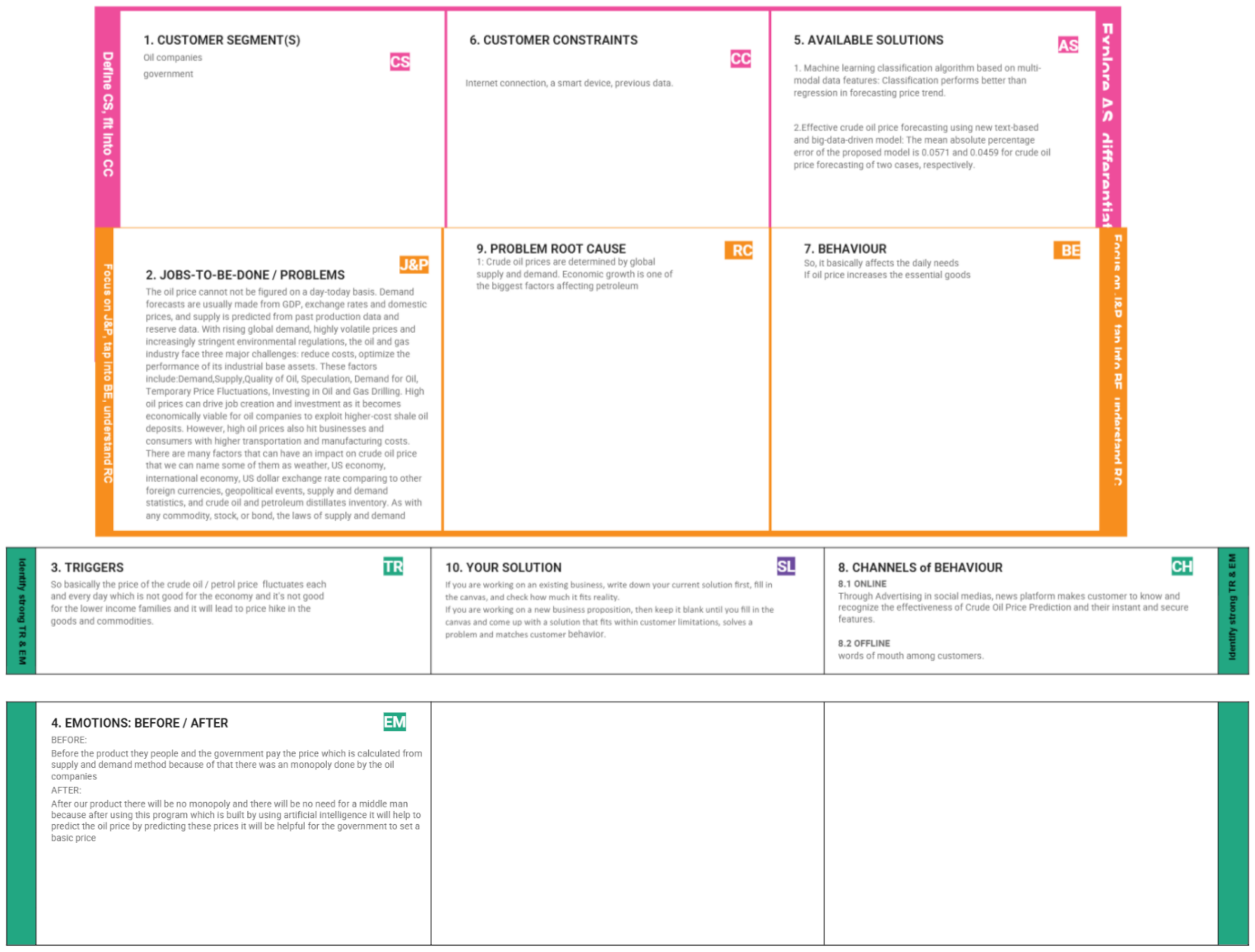
Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

3.3 PROPOSED SOLUTION

S. No.	Parameter	Description
1	Problem Statement (Problem tobe Solved)	Predicting the price of the crude oil is very difficult in recent days due to war time. The oil price which is controlled by the west Asian countries.
2	Idea / Solution description	This system is built by using the AI. By using this system, we can. Predict the oil price which can be useful for countries to buy the crude oil.
3	Novelty / Uniqueness	The application will make understand people how oil price is predicted. There is no mediator between buyer and seller.
4	Social Impact / CustomerSatisfaction	This system provides an effective support to both the buyer and seller & create a satisfaction between them. So, the oil price can be low compared to older price.
5	Business Model (RevenueModel)	This system covers a wide range because we can predict both crude oil, petroleum products & natural gas.
6	Scalability ofthe Solution	Implementing this system provides the performance and fulfillness within the market and creating a platform which can be used worldwide.

3.4 PROBLEM SOLUTION FIT



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Oil companies

government

6. CUSTOMER CONSTRAINTS CC

Internet connection, a smart device, previous data.

5. AVAILABLE SOLUTIONS AS

1. Machine learning classification algorithm based on multi-modal data features: Classification performs better than regression in forecasting price trend.

2.Effective crude oil price forecasting using new text-based and big-data-driven model: The mean absolute percentage error of the proposed model is 0.0571 and 0.0459 for crude oil price forecasting of two cases, respectively.

Explore AS, differentiat

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

The oil price cannot not be figured on a day-today basis. Demand forecasts are usually made from GDP, exchange rates and domestic prices, and supply is predicted from past production data and reserve data. With rising global demand, highly volatile prices and increasingly stringent environmental regulations, the oil and gas industry face three major challenges: reduce costs, optimize the performance of its industrial base assets. These factors include:Demand,Supply,Quality of Oil, Speculation, Demand for Oil, Temporary Price Fluctuations, Investing in Oil and Gas Drilling. High oil prices can drive job creation and investment as it becomes economically viable for oil companies to exploit higher-cost shale oil deposits. However, high oil prices also hit businesses and consumers with higher transportation and manufacturing costs. There are many factors that can have an impact on crude oil price that we can name some of them as weather, US economy, international economy, US dollar exchange rate comparing to other foreign currencies, geopolitical events, supply and demand statistics, and crude oil and petroleum distillates inventory. As with any commodity, stock, or bond, the laws of supply and demand

9. PROBLEM ROOT CAUSE RC

1: Crude oil prices are determined by global supply and demand. Economic growth is one of the biggest factors affecting petroleum

7. BEHAVIOUR BE

So, it basically affects the daily needs

If oil price increases the essential goods

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

So basically the price of the crude oil / petrol price fluctuates each and every day which is not good for the economy and it's not good for the lower income families and it will lead to price hike in the goods and commodities.

10. YOUR SOLUTION SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.

If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behavior.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

Through Advertising in social medias, news platform makes customer to know and recognize the effectiveness of Crude Oil Price Prediction and their instant and secure features.

8.2 OFFLINE

words of mouth among customers.

Identify strong TR & EM

4. EMOTIONS: BEFORE / AFTER EM

BEFORE:

Before the product they people and the government pay the price which is calculated from supply and demand method because of that there was an monopoly done by the oil companies

AFTER:

After our product there will be no monopoly and there will be no need for a middle man because after using this program which is built by using artificial intelligence it will help to predict the oil price by predicting these prices it will be helpful for the government to set a basic price

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Application	User Direct Open with Google Play Store App User Can Download The Crude Oil Price
FR-2	User Products Available	User Using the Application There Are So Many Products In Crude Oil Price App User Update the Energy and Oil Price Instant the Application
FR-3	User Additional Features	User Can Read Latest News and View Oil Price Charts User View Major Energy Quotes User Can Using A Multiple Color Themes
FR-4	User Exceptions	User Can Exchange Rates and Currency Converter

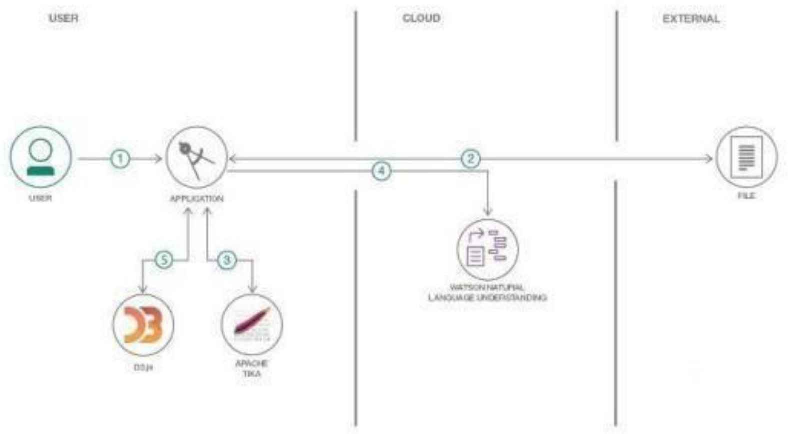
4.2 NON - FUNCTIONAL REQUIREMENT

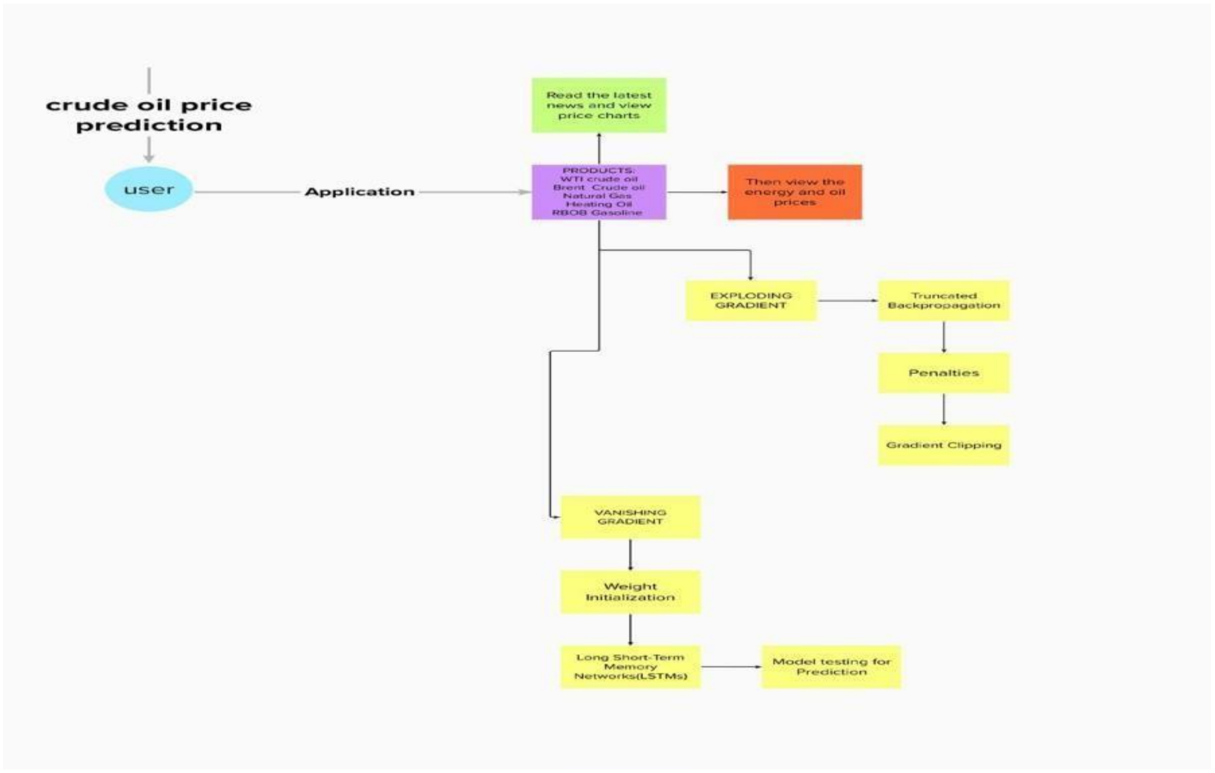
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Used to improve to the Accuracy of crude oil price prediction
NFR-2	Security	In the rising oil price can even shift economical/political power from oil importers to oil exporters communications will be secured
NFR-3	Reliability	Reliability of the pointing towards high –risk components

5. PROJECT DESIGN

5.1 PROJECT FLOW DIAGRAM

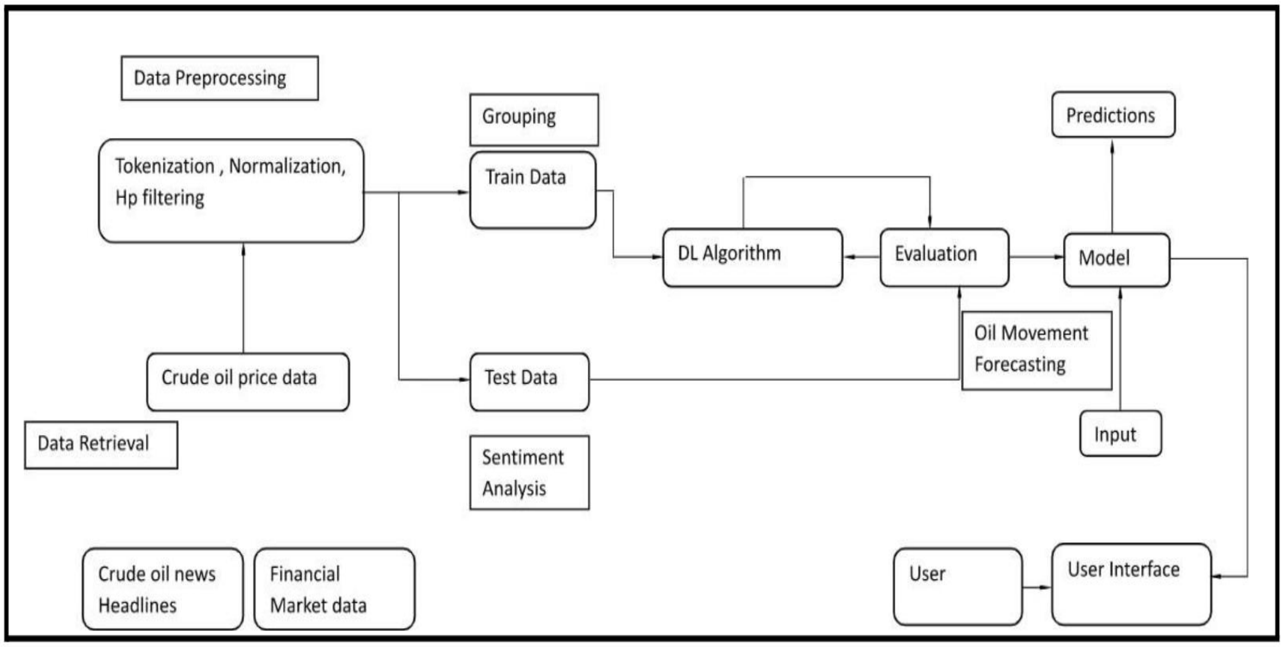
The classic visual representation of how information moves through a system is a data flow diagram (DFD). A tidy and understandable DFD can graphically represent the appropriate quantity of the system demand. It demonstrates how information enters and exits the system, what modifies the data, and where information is kept.



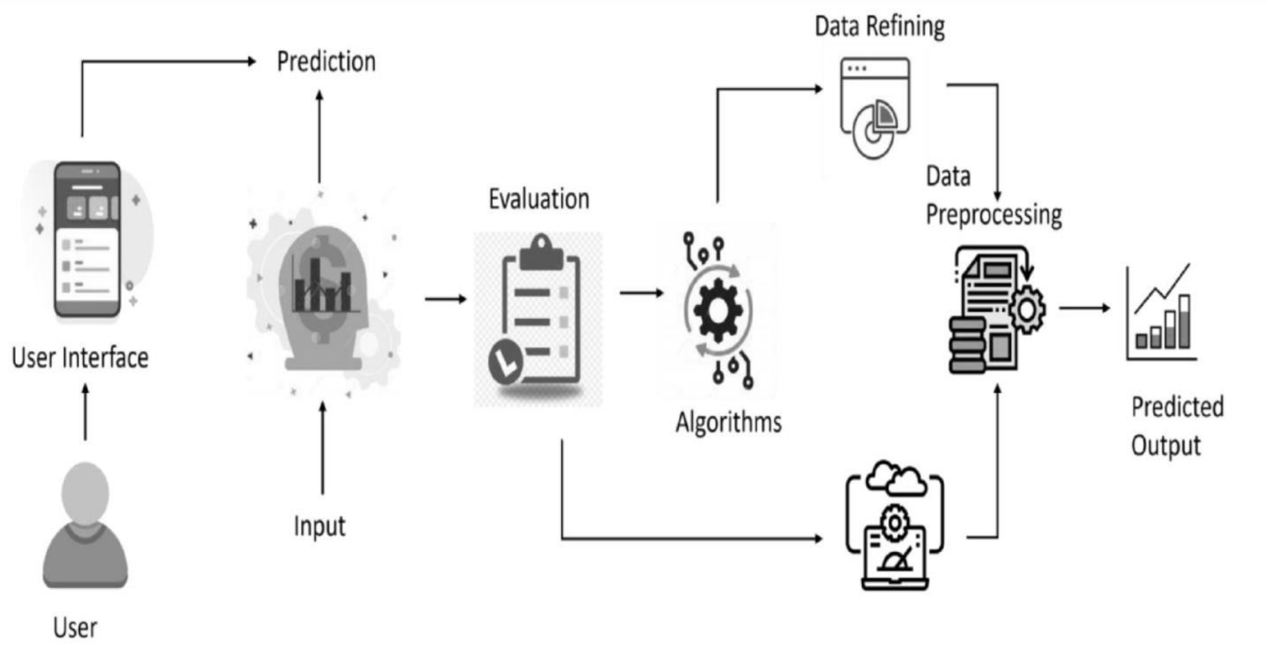

crude oil price prediction
user
Application
Read the latest news and view price charts
PRODUCTS: WTI crude oil Brent Crude oil Natural Gas Heating Oil RBOB Gasoline
Then view the energy and oil prices
EXPLODING GRADIENT
Truncated Backpropagation
Penalties
Gradient Clipping
VANISHING GRADIENT
Weight Initialization
Long Short-Term Memory Networks(LSTMs)
Model testing for Prediction

5.2 SOLUTION AND TECHNICAL ARCHITECTURE:

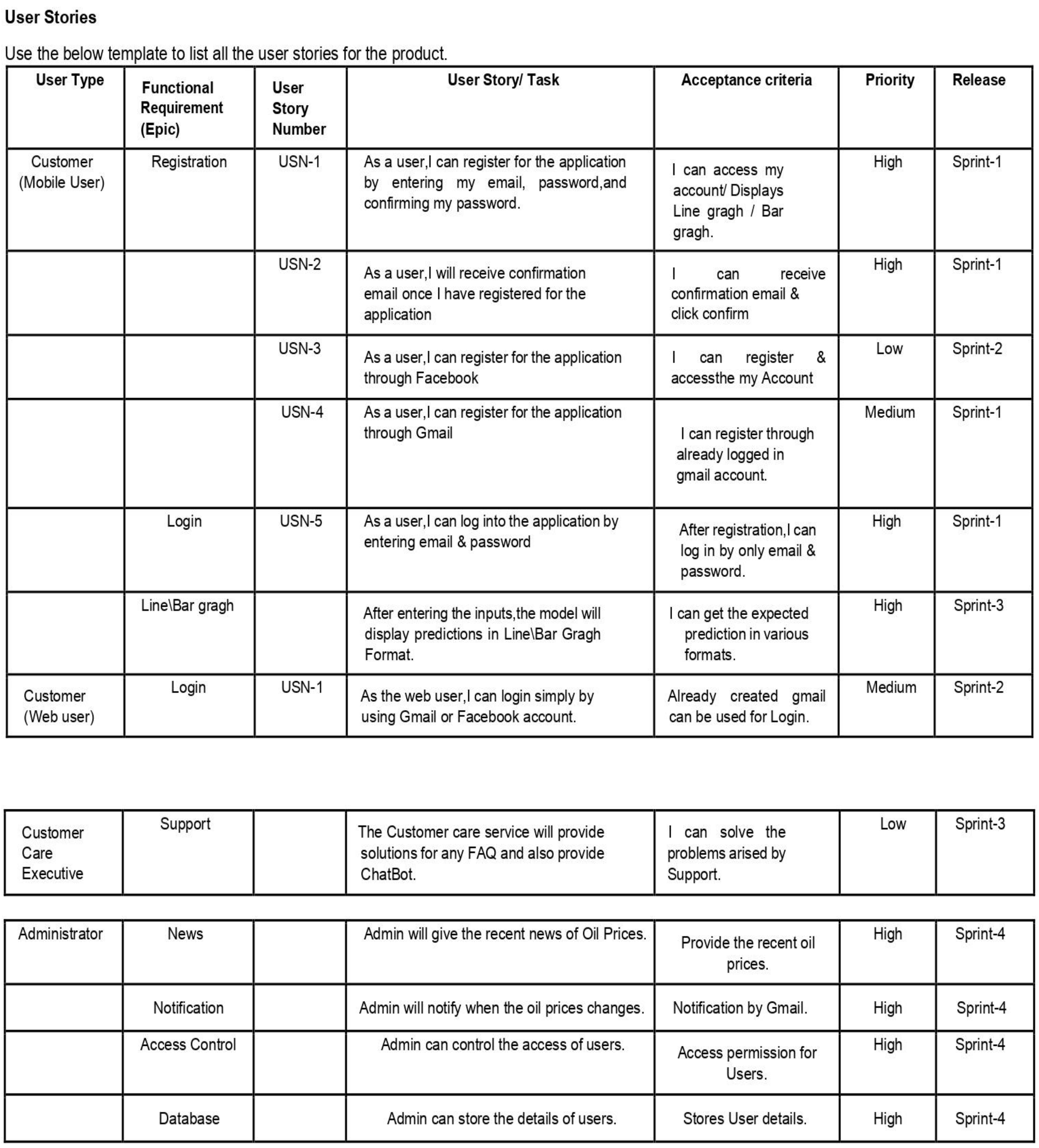
SOLUTION ARCHITECTURE



TECHNICAL ARCHITECTURE



5.3 USER STORIES:



User Stories

Use the below template to list all the user stories for the product.

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/ Displays Line gragh / Bar gragh.	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 & accessthe my Account	Low	Sprint-2
		USN-4	As a user,I can register for the application through Gmail	I can register through already logged in gmail account.	Medium	Sprint-1
	Login	USN-5	As a user,I can log into the application by entering email & password	After registration,I can log in by only email & password.	High	Sprint-1
	Line\Bar gragh		After entering the inputs,the model will display predictions in Line\Bar Gragh Format.	I can get the expected prediction in various formats.	High	Sprint-3
Customer (Web user)	Login	USN-1	As the web user,I can login simply by using Gmail or Facebook account.	Already created gmail can be used for Login.	Medium	Sprint-2
Customer Care Executive	Support		The Customer care service will provide solutions for any FAQ and also provide ChatBot.	I can solve the problems arised by Support.	Low	Sprint-3
Administrator	News		Admin will give the recent news of Oil Prices.	Provide the recent oil prices.	High	Sprint-4
	Notification		Admin will notify when the oil prices changes.	Notification by Gmail.	High	Sprint-4
	Access Control		Admin can control the access of users.	Access permission for Users.	High	Sprint-4
	Database		Admin can store the details of users.	Stores User details.	High	Sprint-4

6. PROJECT PLANNING ANDSCHEDULING

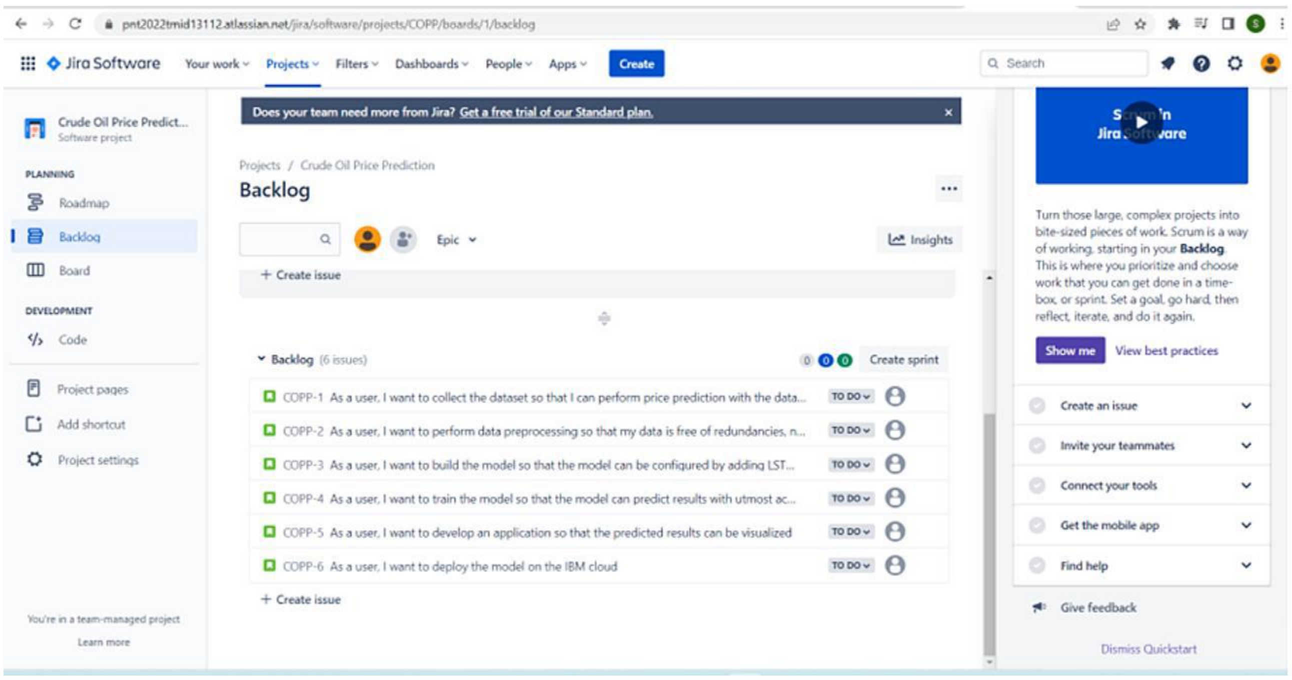
6.1 SPRINT PLANNING AND ESTIMATION

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	10	6 Days	24 Oct 2022	29 Oct 2022	8	29 Oct 2022
Sprint-2	10	6 Days	31 Oct 2022	05 Nov 2022	7	05 Nov 2022
Sprint-3	10	6 Days	07 Nov 2022	12 Nov 2022	8	12 Nov 2022
Sprint-4	10	6 Days	14 Nov 2022	19 Nov 2022	7	19 Nov 2022

6.2 SPRINT DELIVERY SCHEDULE

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Data Collection	USN-1	Collecting the Dataset	10	High
Sprint-1		USN-2	Data Pre-processing	7	Medium
Sprint-2	Model Building	USN-3	Import the required libraries, add the necessary layers and compile the model.	10	High
Sprint-2		USN-4	Training the data classification model using RNN and others systems.	7	Medium
Sprint-3	Training and Testing	USN-5	Training the model and testing the model'sperformance	10	High
Sprint-4		USN-6	Build the system and deploy the model in IBM Cloud	7	Medium

6.3 REPORT FROM JIRA


pnt2022tmid13112.atlassian.net/jira/software/projects/COPP/boards/1/backlog

Does your team need more from Jira? Get a free trial of our Standard plan.

Crude Oil Price Predict... Software project

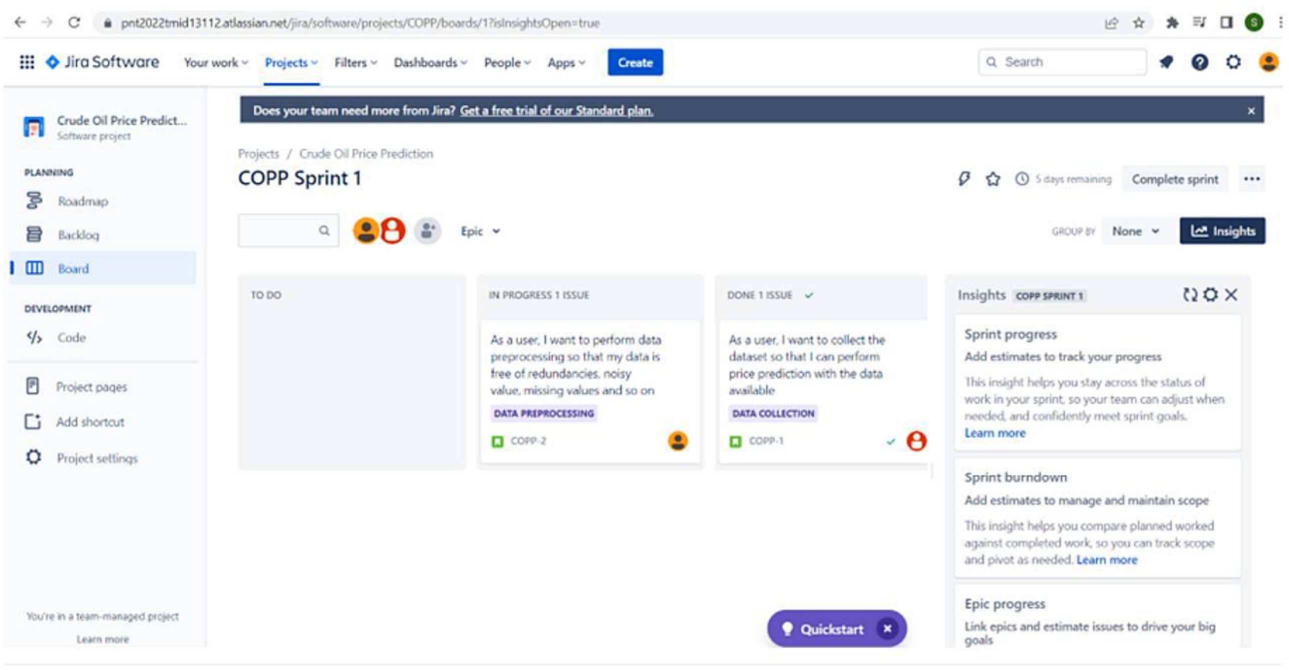
Projects / Crude Oil Price Prediction

Backlog

Backlog (6 issues)

COPP-1 As a user, I want to collect the dataset so that I can perform price prediction with the data... TO DO
COPP-2 As a user, I want to perform data preprocessing so that my data is free of redundancies, n... TO DO
COPP-3 As a user, I want to build the model so that the model can be configured by adding LST... TO DO
COPP-4 As a user, I want to train the model so that the model can predict results with utmost ac... TO DO
COPP-5 As a user, I want to develop an application so that the predicted results can be visualized TO DO
COPP-6 As a user, I want to deploy the model on the IBM cloud TO DO

Turn those large, complex projects into bite-sized pieces of work. Scrum is a way of working, starting in your **Backlog**. This is where you prioritize and choose work that you can get done in a time-box, or sprint. Set a goal, go hard, then reflect, iterate, and do it again.


pnt2022tmid13112.atlassian.net/jira/software/projects/COPP/boards/1?isInsightsOpen=true

Does your team need more from Jira? Get a free trial of our Standard plan.

Crude Oil Price Predict... Software project

Projects / Crude Oil Price Prediction

COPP Sprint 1

5 days remaining

TO DO

IN PROGRESS 1 ISSUE
As a user, I want to perform data preprocessing so that my data is free of redundancies, noisy value, missing values and so on
DATA PREPROCESSING
COPP-2

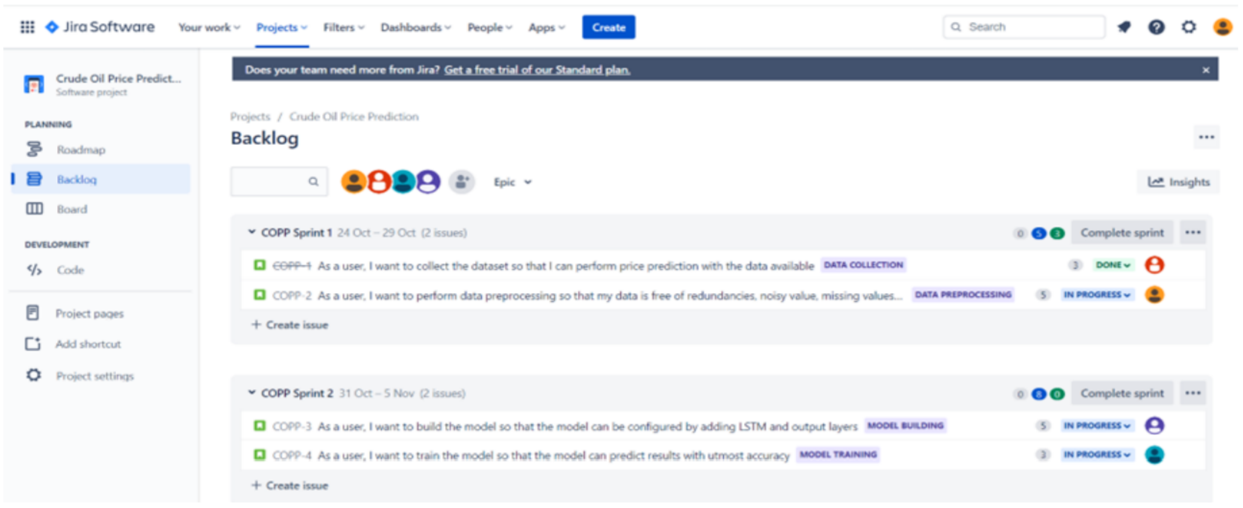
DONE 1 ISSUE
As a user, I want to collect the dataset so that I can perform price prediction with the data available
DATA COLLECTION
COPP-1

Insights COPP SPRINT 1

Sprint progress
Add estimates to track your progress
This insight helps you stay across the status of work in your sprint, so your team can adjust when needed, and confidently meet sprint goals. Learn more

Sprint burndown
Add estimates to manage and maintain scope
This insight helps you compare planned worked against completed work, so you can track scope and pivot as needed. Learn more

Epic progress
Link epics and estimate issues to drive your big goals



Does your team need more from Jira? Get a free trial of our Standard plan.

Projects / Crude Oil Price Prediction

Backlog

COPP Sprint 1 24 Oct – 29 Oct (2 issues) 0 5 3

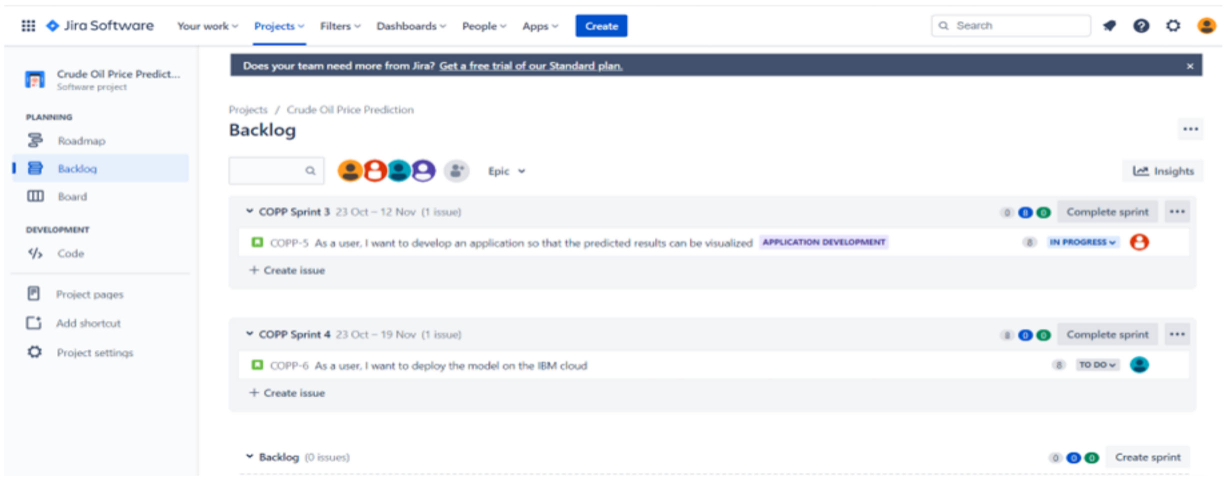
~~COPP-1~~ As a user, I want to collect the dataset so that I can perform price prediction with the data available DATA COLLECTION 3 DONE

COPP-2 As a user, I want to perform data preprocessing so that my data is free of redundancies, noisy value, missing values... DATA PREPROCESSING 5 IN PROGRESS

COPP Sprint 2 31 Oct – 5 Nov (2 issues) 0 8 0

COPP-3 As a user, I want to build the model so that the model can be configured by adding LSTM and output layers MODEL BUILDING 5 IN PROGRESS

COPP-4 As a user, I want to train the model so that the model can predict results with utmost accuracy MODEL TRAINING 3 IN PROGRESS



Does your team need more from Jira? Get a free trial of our Standard plan.

Projects / Crude Oil Price Prediction

Backlog

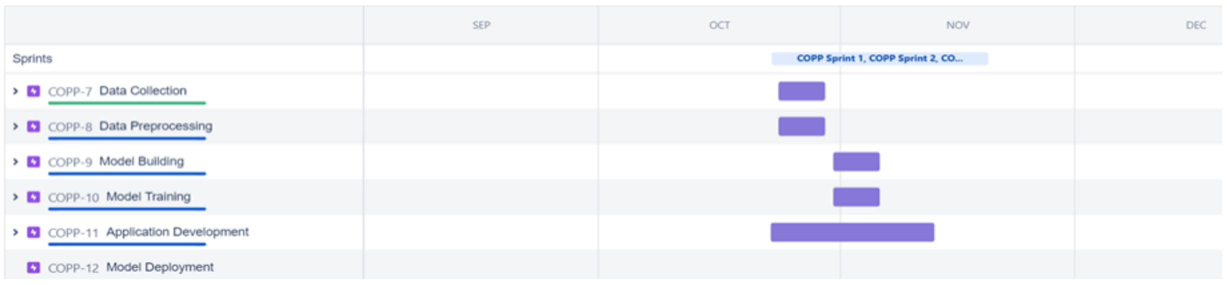
COPP Sprint 3 23 Oct – 12 Nov (1 issue) 0 8 0

COPP-5 As a user, I want to develop an application so that the predicted results can be visualized APPLICATION DEVELOPMENT 8 IN PROGRESS

COPP Sprint 4 23 Oct – 19 Nov (1 issue) 8 0 0

COPP-6 As a user, I want to deploy the model on the IBM cloud 8 TO DO

Backlog (0 issues) 0 0 0



7. CODING AND SOLUTION

7.1 FEATURE 1

```
import numpy as np                                                    In [1]:
import pandas as pd import
seaborn as sns
import matplotlib.pyplot as plt
                                                                       In [2]:
from google.colab import files
 uploaded = files.upload()                                             In [3]:
```

```
import io
ds = pd.read_excel(io.BytesIO(uploaded['Crude Oil Prices Daily.xlsx']))ds.head()
ds[:10]
```

Out[3]:

	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
5	1986-01-09	26.03
6	1986-01-10	25.65
7	1986-01-13	25.08
8	1986-01-14	24.97
9	1986-01-15	25.18

In [4]:

```
ds.isnull().sum() Date          0
Closing Value        7
dtype: int64
```

Out[4]:

```
ds.dropna(axis=0,inplace=True)
```

In [5]:

```
ds.isnull().sum()
```

In [6]:

```
Date                 0
Closing Value        0
dtype: int64
```

Out[6]:

```
data=ds.reset_index()['Closing Value']
```

In [7]:

```
data
```

Out[7]:

```
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 [8]:

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler(feature_range=(0,1))
data=scaler.fit_transform(np.array(data).reshape(-1,1))
```

In [9]:

```
data
```

Out[9]:

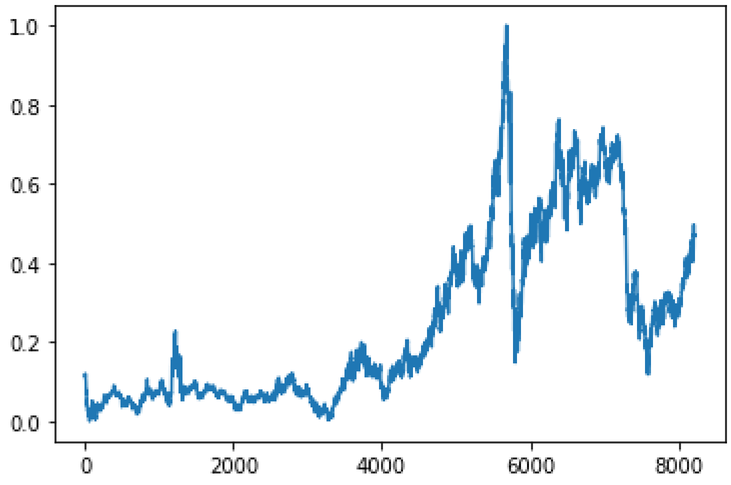
```
array([[0.11335703],
       [0.11661484],
       [0.12053902],
       ...,
       [0.46497853],
       [0.47038353],
       [0.47149415]])
```

In [10]:

```
plt.plot(data)
```

Out[10]:

```
[<matplotlib.lines.Line2D at 0x7f9e733ad2d0>]
```



In [11]:

```
training_size=int(len(data)*0.65)
test_size=len(data)-training_size
```

In [12]:

```
train_data,test_data=data[0:training_size,:],data[training_size:len(data),:1]
```

In [13]:

```
training_size,test_size
```

Out[13]:

```
(5340, 2876)
```

```
train_data.shape
```

Out[13]:

```
(5340,1 )
```

In [14]:

```
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 [15]:

```
time_step=10 x_train,y_train=create_dataset(train_data,time_step)
x_test,y_test=create_dataset (test_data,time_step)
```

In [16]:

```
print(x_train.shape),print(y_train.shape)(5329, 10)
(5329,)

(None, None)
```

Out[16]:

In [17]:

```
print(x_test.shape),print(y_test.shape)(2865, 10)
(2865,)

(None, None)
```

Out[17]:

```
x_train
```

In [18]:

Out[18]:

```
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 [19]:

```
x_test
```

Out[19]:

```
array([[0.38005331, 0.36872501,          0.37324152,     ...,      0.3537687 ,
0.35465719,
        0.3499926 ],
              [0.36872501, 0.37324152,  0.38205242,     ...,     0.35465719,    0.3499926
,
        0.3465867 ],
              [0.37324152, 0.38205242,  0.38042352,     ...,      0.3499926 ,    0.3465867
,
        0.34355101],
       ...,
              [0.40604176, 0.41218718,  0.41041019,     ...,     0.46794017,
0.47297497,
        0.47119799],
              [0.41218718, 0.41041019,  0.43513994,     ...,     0.47297497,
0.47119799,
0.47341922],
                    [0.41041019, 0.43513994, 0.4417296 , ..., 0.47119799,
   0.47341922,
                      0.46497853]])
```

In [20]:

```
x_train1=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 [21]:

```
x_train1
```

Out[21]:

```
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 [22]:

```
from tensorflow.keras.models import Sequentialfrom
tensorflow.keras.layers   import   Dense from
tensorflow.keras.layers import LSTM
```
INITIALIZING THE MODEL

In [23]:

```
model=Sequential()
```
ADDING LSTM AND OUTPUT LAYERS

In [24]:

```
model.add(LSTM(50,return_sequences=True,input_shape=(10,1)))
model.add(LSTM(50,return_sequences=True)) model.add(LSTM(50))
```

In [25]:

```
model.add(Dense(1))
```

In [26]:

```
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
_________________________________________________________________
```
CONFIGURING THE LEARNING PROCESS

In [27]:

```
model.compile(loss='mean_squared_error',optimizer='adam')
```
MODEL TRAINING

In [28]:

```
model.fit(x_train,y_train,validation_data=(x_test,y_test),epochs=3,batch_size=64,verbose=1)
```

```
Epoch 1/3
84/84 [==============================] - 9s 39ms/step - loss: 0.0017 -
val_loss: 8.1129e-04Epoch
2/3
84/84 [==============================] - 2s 24ms/step - loss: 1.2676e-
04 - val_loss: 7.8078e-04Epoch 3/3
84/84 [==============================] - 2s 23ms/step - loss: 1.2624e-
04 - val_loss: 7.7794e-04
```

Out[28]:

```
<keras.callbacks.History at 0x7f9e150bd650>
```
MODEL EVALUATION

In [29]:

```
##Transformback to original form
train_predict=scaler.inverse_transform(train_data)
test_predict=scaler.inverse_transform(test_data) ### Calculate RMSE
performance metrics
import math
from sklearn.metrics import mean_squared_error
math.sqrt(mean_squared_error(train_data,train_predict))
```

Out[29]:

```
29.347830443269938
```

MODEL SAVING

In [30]:

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

In [31]:

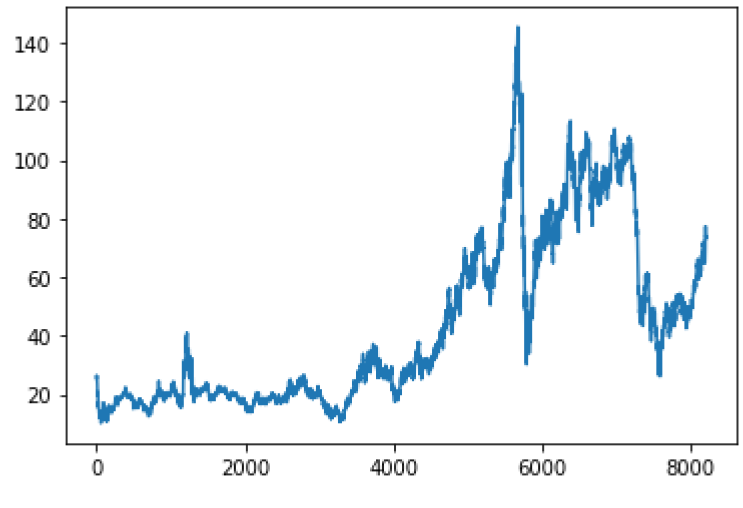
```
model.save("crude_oil.hs")
```

```
WARNING:absl:Found untraced functions such as lstm_cell_layer_call_fn,
lstm_cell_layer_call_and_return_conditional_losses, lstm_cell_1_layer_call_fn,
lstm_cell_1_layer_call_and_return_conditional_losses, lstm_cell_2_layer_call_fn while saving (showing
5 of 6). These functions will not be directly callable after loading.
```

MODEL TESTING

In [32]:

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



In [33]:

```
len(test_data)
```

Out[33]:

```
2876
```

In [34]:

```
x_input=test_data[2866:].reshape(1,-1)x_input.shape
```

Out[34]:

```
(1, 10)
```

In [35]:

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

In [36]:

```
temp_input
```

Out[36]:

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

In [37]:

```
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))
                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
```

```
[0.4805713]
11
1 day input [0.4811195   0.49726048 0.46794017 0.47297497 0.47119799
0.47341922
0.46497853 0.47038353 0.47149415 0.4805713 ]
1 day output [[0.4844224]]
2 day input [0.49726048 0.46794017 0.47297497 0.47119799 0.47341922
0.46497853
0.47038353 0.47149415 0.4805713   0.48442239]
2 day output [[0.4833879]]
3 day input [0.46794017 0.47297497 0.47119799 0.47341922 0.46497853
0.47038353
0.47149415 0.4805713   0.48442239 0.48338789]
```

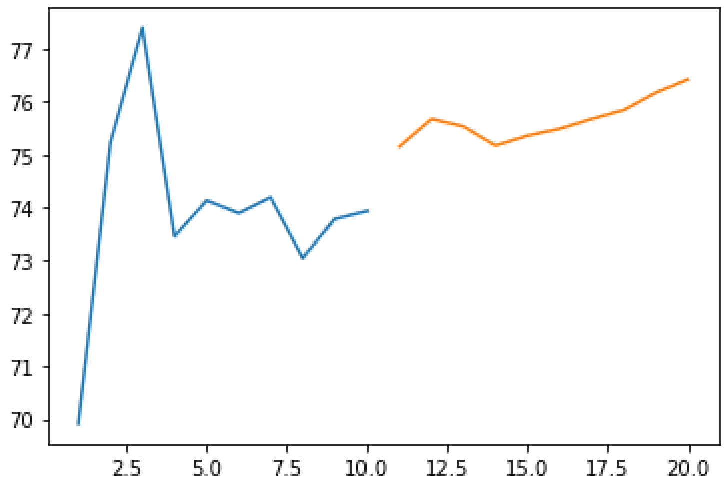
```
3 day output [[0.48069027]]
4 day input [0.47297497 0.47119799 0.47341922                    0.46497853          0.47038353
0.47149415
0.4805713   0.48442239 0.48338789 0.48069027]
4 day output [[0.4820817]]
5 day input [0.47119799 0.47341922 0.46497853                    0.47038353          0.47149415
0.4805713
0.48442239 0.48338789 0.48069027 0.48208171]
5 day output [[0.48304394]]
6 day input [0.47341922 0.46497853 0.47038353                    0.47149415
0.4805713   0.48442239
0.48338789 0.48069027 0.48208171 0.48304394]
6 day output [[0.48441863]]
7 day input [0.46497853 0.47038353 0.47149415                    0.4805713           0.48442239
0.48338789
0.48069027 0.48208171 0.48304394 0.48441863]
7 day output [[0.48566842]]
8 day input [0.47038353 0.47149415 0.4805713                     0.48442239          0.48338789
0.48069027
0.48208171 0.48304394 0.48441863 0.48566842]
8 day output [[0.48811078]]
9 day input [0.47149415 0.4805713   0.48442239                   0.48338789          0.48069027
0.48208171

0.48304394 0.48441863 0.48566842 0.48811078]
9 day output [[0.48995987]]
```

In [38]:

```
day_new=np.arange(1,11)
day_pred=np.arange(11,21)len(data)
plt.plot(day_new, scaler.inverse_transform(data[8206:]))plt.plot(day_pred,
scaler.inverse_transform(lst_output))
```

Out[38]:

```
[<matplotlib.lines.Line2D at 0x7f9e151ef6d0>]
```

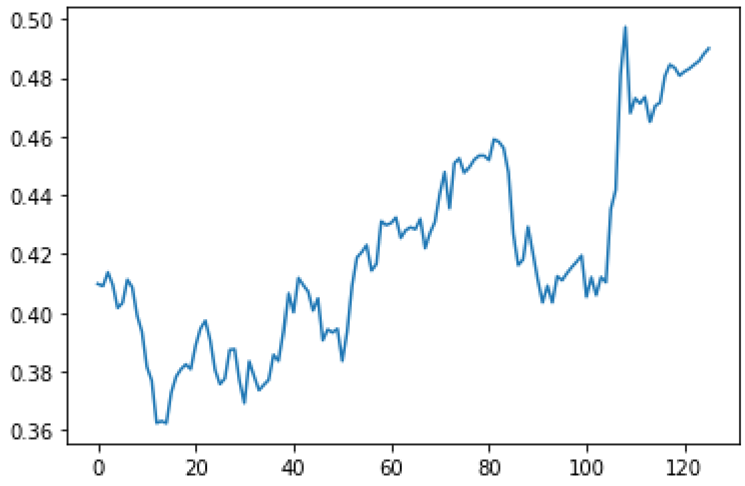


In [39]:

```
df3=data.tolist()
df3.extend(lst_output)
plt.plot(df3[8100:])
```

Out[39]:

```
[<matplotlib.lines.Line2D at 0x7f9e10cc3d10>]
```

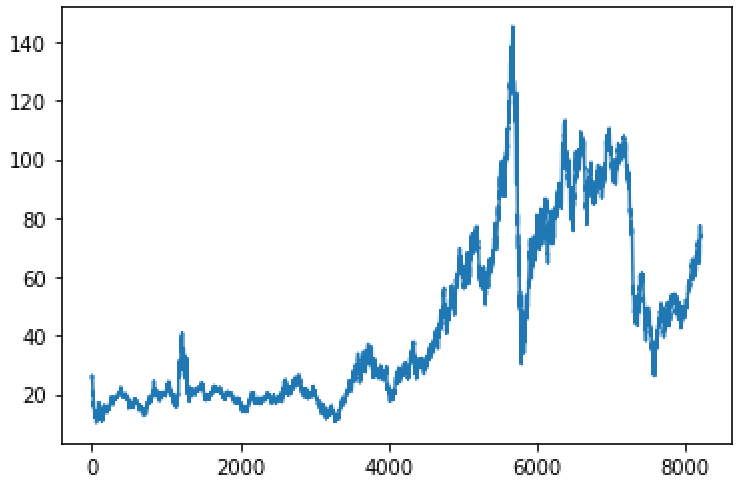


In [40]:

```
df3=scaler.inverse_transform(df3).tolist()plt.plot(scaler.inverse_transform(data))
```

Out[40]:

```
[<matplotlib.lines.Line2D at 0x7f9e10f89e10>]
```



7.2 FEATURE 2

login.html

```html
<!DOCTYPE html>
<html lan="en" and dir="Itr">
   <head>
      <meta charset="utf-8">
      <title>login form</title>
      <link rel="stylesheet" href="style.css">
       <script src ="login.js"></script>
   </head>
   <body>
      <form class="box" action="login.html" method="POST">
         <h1>CRUDE OIL PRICE PREDICTION</h1>
         <h2>
             LOGIN
         </h2>
         <input type="text" name="" placeholder="Enter Username" id="username">
         <input type="password" name="" placeholder="Enter Password" id="password">
         <input type="submit" name="" value="Login"  onclick="validate()">
         <h3><a href="register.html"> New User ? Register </a></h3>
      </form>

   </body>
</html>
```

register.html

```html
<!DOCTYPE html>
<html lan="en" and dir="Itr">
   <head>
      <meta charset="utf-8">
      <title>login form</title>
      <link rel="stylesheet" href="register.css">
       <script src ="login.js"></script>
   </head>
   <body>
      <form class="box" action="login.html" method="POST">
         <h1>CRUDE OIL PRICE PREDICTION</h1>
        <h2>
             Register
         </h2>
         <input type="text" name="" placeholder="Enter Username" id="username">
```

```
            <input type="email" name="" placeholder="Enter Your Email Id" id="Email">
           <input type="number" name="" placeholder="Enter Your Number" id="Number">
           <input type="password" name="" placeholder="Enter Password" id="password">
           <input type="submit" name="" value="Register"  onclick="validate()">
           <h3><a href="login.html"> Login </a></h3>
        </form>

    </body>
</html>
```

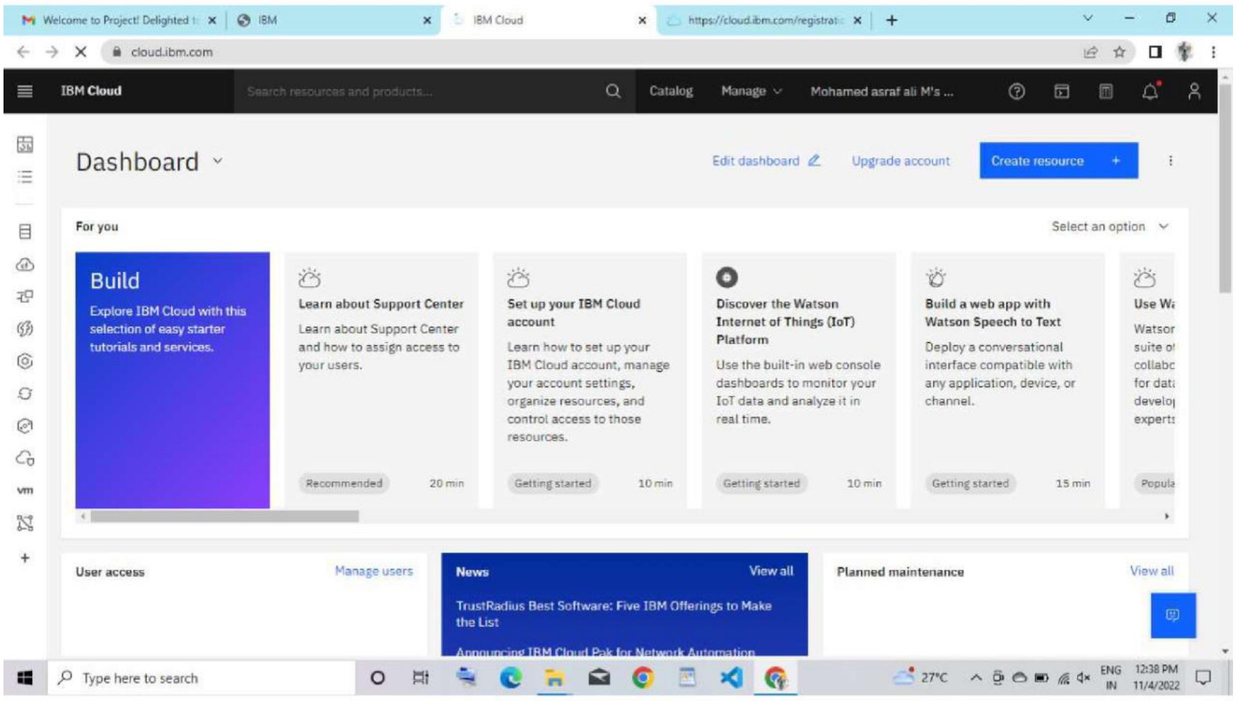
Style.css

```
body{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
    background: url(p2.jpg);
    background-size: cover;
}
.box{
    width: 300px;
    padding: 30px;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%,-50%);
    background: rgb(14, 14, 14);
    text-align: center;
}
.box h1
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}
.box h2
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}

.box input[type="text"],.box input[type="password"] ,.box input[type="date"],.box
input[type="Number"],.box input[type="Email"]
{
    border: 0;
    background: white;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid #2af003;
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: #fff6ff(18, 18, 179);
    border-radius: 24px;
    transition: 0.25px;
}
.box input[type="text"]:focus,.box input[type="password"]:focus{
    width: 270px;
    border-color: rgb(238, 26, 203);
    }
.box input[type="submit"]{
    border: 0;
    background: none;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid rgb(211, 15, 152);
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: rgb(73, 31, 224);
    border-radius: 24px;
    transition: 0.25px;
    cursor: pointer;
}
.box input[type="submit"]:hover{
    background: rgb(100, 182, 53);
}
h3{
    color: wheat;
}
```

Register.css

```
body{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
    background: url(ppp.jpg);
    background-size: cover;
}
.box{
    width: 300px;
    padding: 30px;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%,-50%);
    background: rgb(14, 14, 14);
    text-align: center;
}
.box h1
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}
.box h2
{
    color: rgb(253, 249, 251);
    text-transform: uppercase;
    font-weight: 700;
}

.box input[type="text"],.box input[type="password"] ,.box
input[type="date"],.box input[type="Number"],.box input[type="Email"]
{
    border: 0;
    background: white;



    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid #2af003;
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: #fff6ff(18, 18, 179);
    border-radius: 24px;
    transition: 0.25px;
}
.box input[type="text"]:focus,.box input[type="password"]:focus{
    width: 270px;
    border-color: rgb(238, 26, 203);
    }
.box input[type="submit"]{
    border: 0;
    background: none;
    display: block;
    margin: 28px auto;
    text-align: center;
    border: 3px solid rgb(211, 15, 152);
    padding: 14px 10px;
    width: 220px;
    outline: none;
    color: rgb(73, 31, 224);
    border-radius: 24px;
    transition: 0.25px;
    cursor: pointer;
}
.box input[type="submit"]:hover{
    background: rgb(100, 182, 53);
}
h3{
    color: wheat;
}
```

SREENSHOTS



IBM Cloud

Dashboard

Edit dashboard | Upgrade account | Create resource +

For you | Select an option

Build
Explore IBM Cloud with this selection of easy starter tutorials and services.

Learn about Support Center
Learn about Support Center and how to assign access to your users.
Recommended 20 min

Set up your IBM Cloud account
Learn how to set up your IBM Cloud account, manage your account settings, organize resources, and control access to those resources.
Getting started 10 min

Discover the Watson Internet of Things (IoT) Platform
Use the built-in web console dashboards to monitor your IoT data and analyze it in real time.
Getting started 10 min

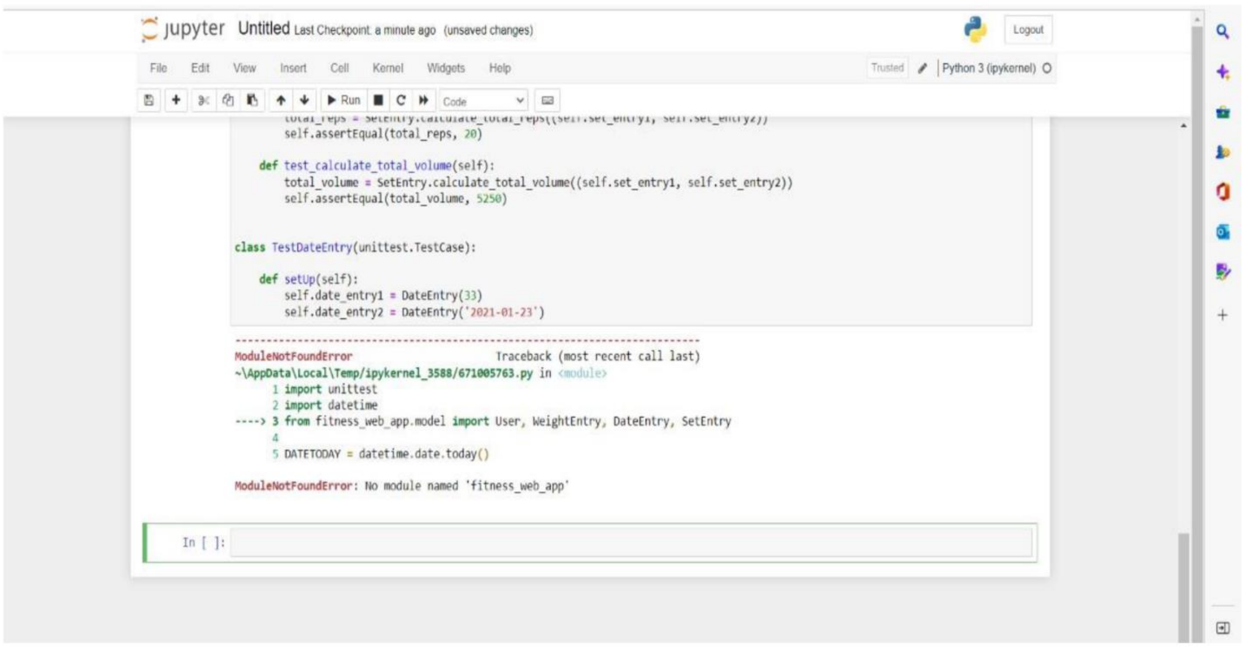
Build a web app with Watson Speech to Text
Deploy a conversational interface compatible with any application, device, or channel.
Getting started 15 min

User access | Manage users

News | View all
TrustRadius Best Software: Five IBM Offerings to Make the List
Announcing IBM Cloud Pak for Network Automation

Planned maintenance | View all

CLOUD ACCOUNT CREATION



jupyter Untitled Last Checkpoint: a minute ago (unsaved changes)

```
        total_reps = SetEntry.calculate_total_reps((self.set_entry1, self.set_entry2))
        self.assertEqual(total_reps, 20)

    def test_calculate_total_volume(self):
        total_volume = SetEntry.calculate_total_volume((self.set_entry1, self.set_entry2))
        self.assertEqual(total_volume, 5250)


class TestDateEntry(unittest.TestCase):

    def setUp(self):
        self.date_entry1 = DateEntry(33)
        self.date_entry2 = DateEntry('2021-01-23')
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_3588/671005763.py in <module>
      1 import unittest
      2 import datetime
----> 3 from fitness_web_app.model import User, WeightEntry, DateEntry, SetEntry
      4 
      5 DATETODAY = datetime.date.today()

ModuleNotFoundError: No module named 'fitness_web_app'
```

In [ ]:

8. TESTING

8.1 TEST CASES

A test case has components that describe input, action and an expected response, in order to determine if a feature of an application is working correctly. A test case is a set of instructions on “HOW” to validate a particular test objective/target, which when followed willtell us if the expected behavior of the system is satisfied or not.

Characteristics of a good test case:

- Accurate: Exacts the purpose.
- Economical: No unnecessary steps or words.
- Traceable: Capable of being traced to requirements.
- Repeatable: Can be used to perform the test over and over.
- Reusable: Can be reused if necessary

Test case analysis This report shows the number of test cases that have passed, failed, anduntested.

Section	Total Cases	Not Tested	Fail	Pass
ML Model	4	0	0	4
Flask Application	4	0	0	4
IBM cloud	4	0	0	4
Exception Reporting	2	0	0	2
Final Report output	4	0	0	4

8.2 USER ACCEPTANCE TESTING

This sort of testing is carried out by users, clients, or other authorized bodies to identify the requirements and operational procedures of an application or piece of software. The most crucial stage of testing is acceptance testing since it determines whether or not the customer will accept the application or programme. It could entail the application's U.I., performance, usability, and usefulness. It is also referred to as end-user testing, operational acceptance testing, and user acceptance testing (UAT). The purpose is to briefly explain the test coverage and open issues of the crude oil priceprediction project at the time of the release to user acceptance testing

Defect Analysis:

Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't fix	0	0	0	1	1
Totals	8	0	2	2	12

Test case analysis

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

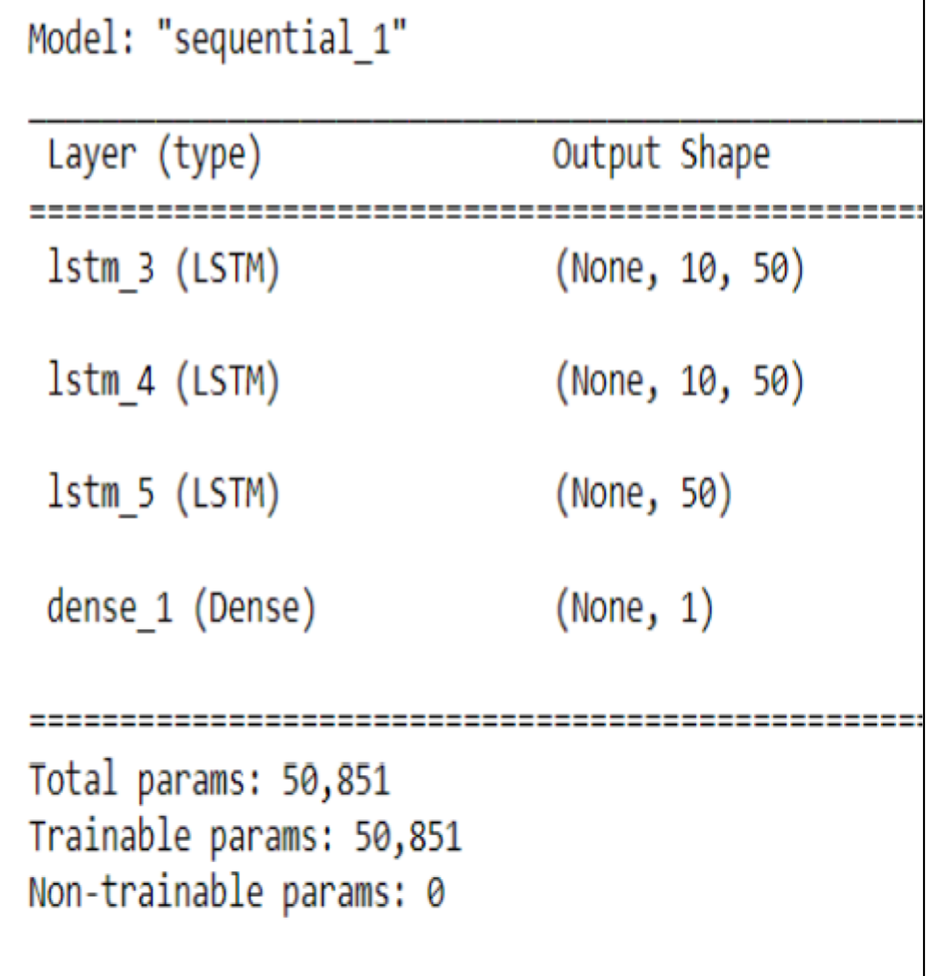
Section	Total Cases	Not Tested	Fail	Pass
ML Model	4	0	0	4
Flask Application	4	0	0	4
IBM Cloud	4	0	0	4
Exception Reporting	2	0	0	2
Final Report Output	4	0	0	4

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

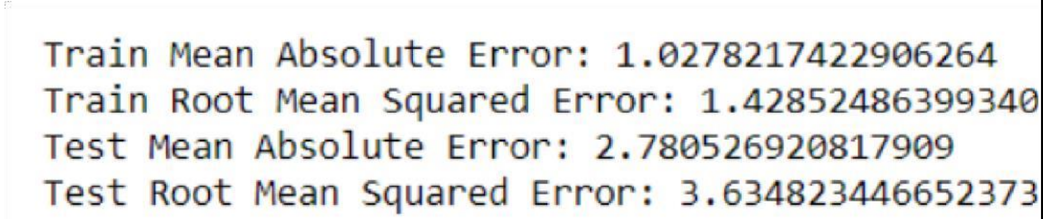
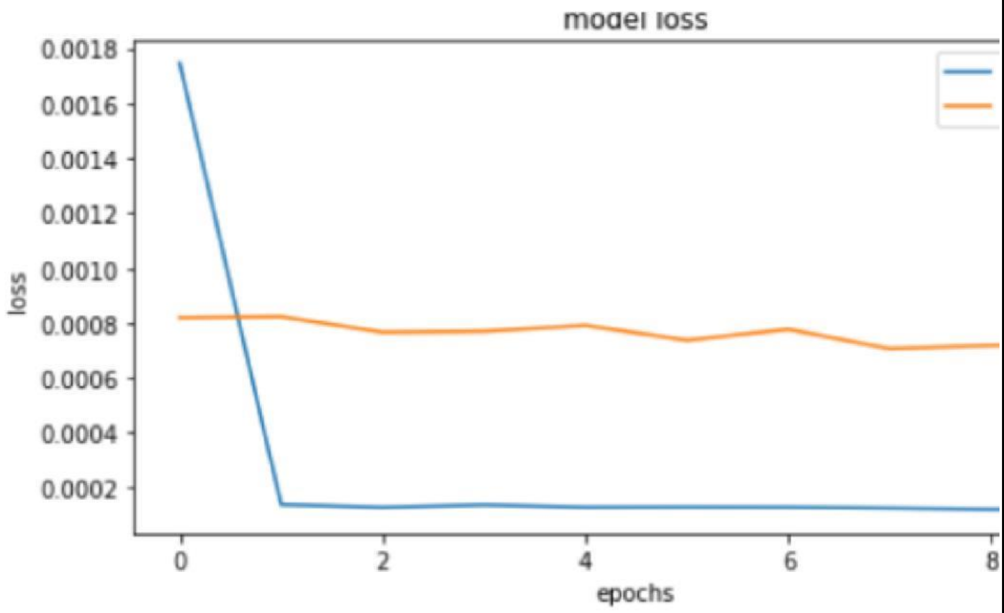
Resolution	**Severity 1**	**Severity 2**	**Severity 3**	**Severity 4**	**Subtotal**
By Design	3	0	0	0	3
Duplicate	1	0	1	0	2
External	0	0	0	0	0
Fixed	4	0	1	1	6

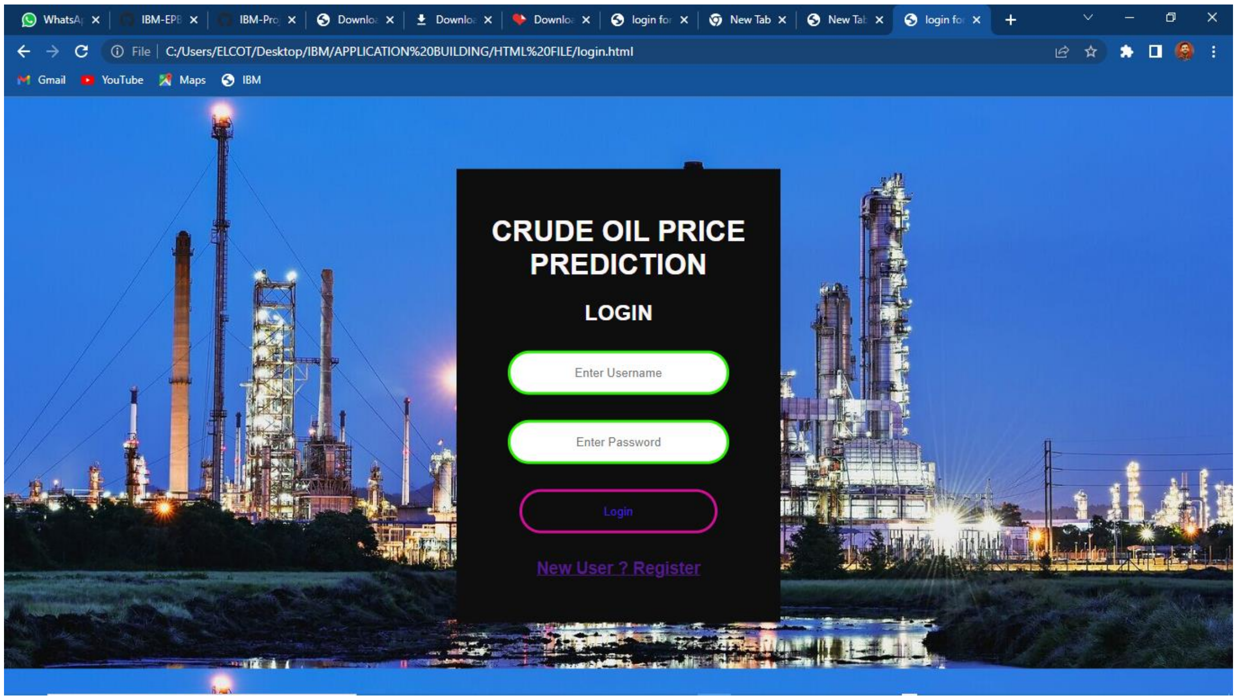
9. RESULTS

9.1 PERFORMANCE METRICS

S.No	Parameters	Values	Screenshot
1.	Model Summary		

```
Model: "sequential_1"
_________________________________________________________
 Layer (type)                Output Shape
=========================================================
 lstm_3 (LSTM)               (None, 10, 50)

 lstm_4 (LSTM)               (None, 10, 50)

 lstm_5 (LSTM)               (None, 50)

 dense_1 (Dense)             (None, 1)

=========================================================
Total params: 50,851
Trainable params: 50,851
Non-trainable params: 0
```

2.	Accuracy	 Train Mean Absolute Error: 1.0278217422906264 Train Root Mean Squared Error: 1.42852486399340 Test Mean Absolute Error: 2.780526920817909 Test Root Mean Squared Error: 3.634823446652373 



CRUDE OIL PRICE PREDICTION

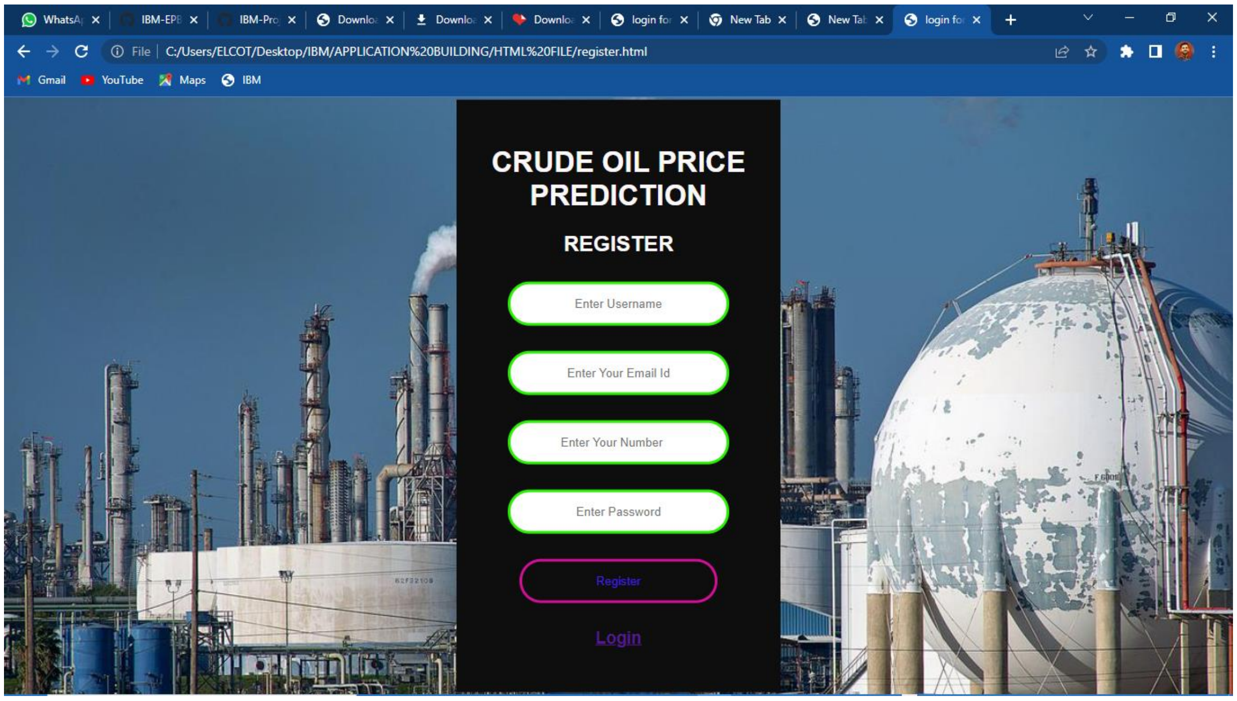
LOGIN

Enter Username

Enter Password

Login

New User ? Register



CRUDE OIL PRICE PREDICTION

REGISTER

Enter Username

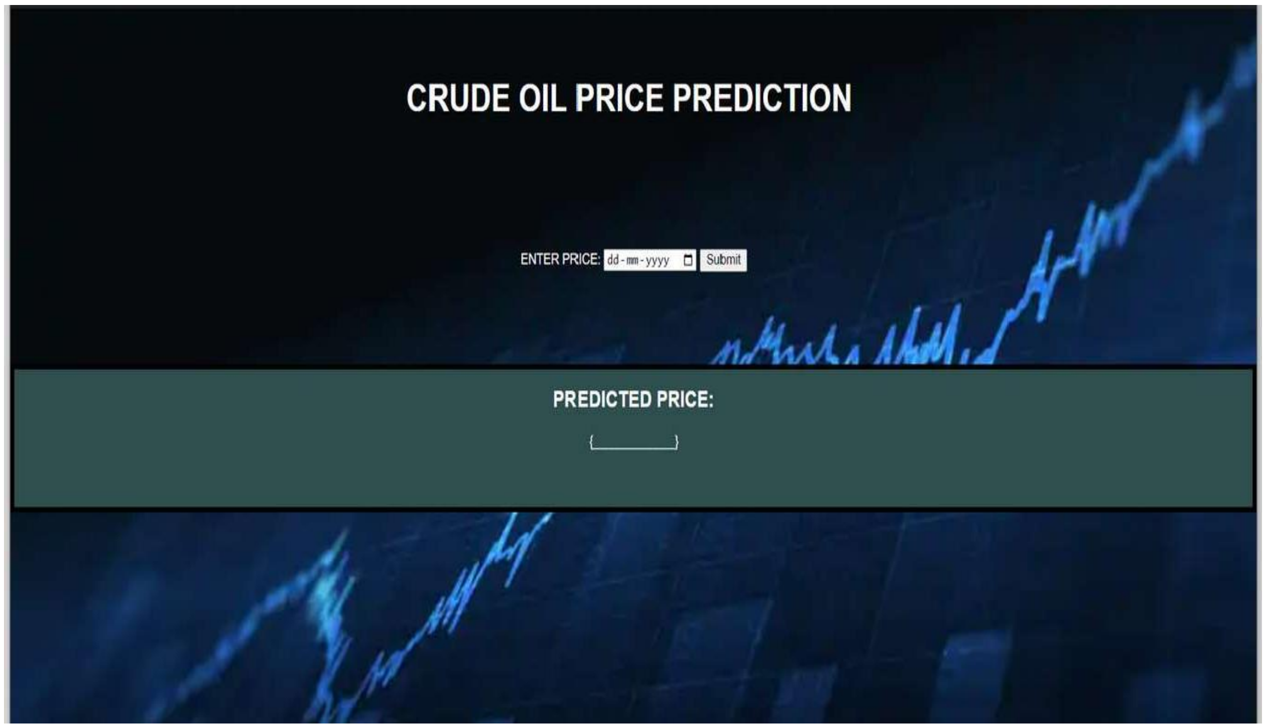
Enter Your Email Id

Enter Your Number

Enter Password

Register

Login


CRUDE OIL PRICE PREDICTION
ENTER PRICE: dd-mm-yyyy
Submit
PREDICTED PRICE:
{_______}

10. ADVANTAGES & DISADVANTAGES

ADVANTAGE

- Give accurate result
- Easy to access and get the price
- Effective with large datasets
- Prediction of crude oil price can help the importers to choose the right time to buy as they wait for the prices to fall down
- Prediction of crude oil prices can help the exporters to increase the demand
- It can even help in shifting the political powers
- can assist in minimizing the risks associated with volatility in oil prices

DISADVANTAGE

- Hard to find oil price
- Inefficient in accuracy
- Poor Customer support
- The prediction results may lack accuracy
- Volatility in prices may be misleading

11. CONCLUSION

Predicting Crude Oil prices is a very challenging problem due to the high volatility of oil prices. In this paper, we developed a new oil price prediction approach using ideas and tools from stream learning, a machine learning paradigm for analysis and inference of continuous flow of non-stationary data. Our stream learning model will be updated whenever new oil price dataare available, and provided to model, so the model continuously evolves over time, and can capture the changing pattern of oil prices. In addition, updating the model requires only a small constant time per new data example, the experiment results show that our stream learning model outperformed four other popular oil price prediction models over a variety of forecast time horizons. This process is used to Predict the oil Prices. The prediction model predicts continuous valued functions.

12. FUTURE SCOPE

Future research may extend our work by considering a richer set of market variables, such as political or commercial factors and phases of economic instability, which are often determinants of crude oil price. Moreover, another direction for future research is the application of the proposed model to forecast the price of other commodities. Moreover, it is a worthwhile direction to explore the consideration of one or more computational cost factors when comparing different forecasting models. Therefore, calculations based on operational research methods might be a good direction.

13. APPENDIX

SOURCE CODE

BUILDING PYTHON

IMPORTING LIBRARIES

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g.
pd.read_csv)import datetime
from pylab import rcParams
import matplotlib.pyplot as
pltimport warnings
import itertools
import statsmodels.api as sm
from keras.models import
Sequentialfrom keras.layers
import Dense
from keras.layers import
LSTM from keras.layers
import Dropout
from sklearn.metrics import mean_squared_error
from keras.callbacks import ReduceLROnPlateau, EarlyStopping,
ModelCheckpointfrom sklearn.metrics import mean_squared_error
from sklearn.metrics import
mean_absolute_errorimport seaborn as sns
sns.set_context("paper",  font_scale=1.3)
sns.set_style('white')
import math
from sklearn.preprocessing import MinMaxScaler
# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list all
files under the input directory
warnings.filterwarnings("ignore")
plt.style.use('fivethirtyeight')
```

```
import os
for dirname, _, filenames in
    os.walk('/kaggle/input'):for filename in
    filenames:
        print(os.path.join(dirname, filename))
```

IMPORTING DATA

```
dateparse = lambda x: pd.datetime.strptime(x, '%b %d,
%Y')#Read csv file
from google.colab import
filesuploaded =
files.upload()
```

```
Upload widget is only available when the cell has been executed in the current
browser session. Pleasererun this cell to enable.
Saving Crude Oil Prices Daily.xlsx to Crude Oil Prices
Daily.xlsximport io
df = pd.read_excel(io.BytesIO(uploaded['Crude Oil Prices
Daily.xlsx']))df.head()
df[:10]
```

	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
5	1986-01-09	26.03
6	1986-01-10	25.65
7	1986-01-13	25.08
8	1986-01-14	24.97
9	1986-01-15	25.18

```
#Sort dataset by column
Datedf =
df.sort_values('Date')
df = df.groupby('Date')['Closing


Value'].sum().reset_index()df.set_index('Date',
inplace=True)
df=df.loc[datetime.date(year=2000,month=1,day=1):]
df.head()
```

	Closing Value
Date	
2000-01-04	25.56
2000-01-05	24.65
2000-01-06	24.79
2000-01-07	24.79
2000-01-10	24.71

DATA PRE-PROCESSING

```
def DfInfo(df_initial):
    # gives some infos on columns types and numer of null values
    tab_info = pd.DataFrame(df_initial.dtypes).T.rename(index={0: 'column
    type'})tab_info =
tab_info.append(pd.DataFrame(df_initial.isnull().sum()).T.rename(index={0: 'null
values (nb)'}))
    tab_info =
tab_info.append(pd.DataFrame(df_initial.isnull().sum() /
df_initial.shape[0] * 100).T.
 rename(index={0: 'null values (%)'}))
    return
tab_info
DfInfo(df)
```

	Closing Value
Column type	float64
null values (nb)	0
null values (%)	0.0

```
df.index
DatetimeIndex(['2000-01-04', '2000-01-05', '2000-01-06', '2000-01-
                                                              07',
              '2000-01-10', '2000-01-11', '2000-01-12', '2000-01-13',
               '2000-01-14', '2000-01-18',
               ...
               '2018-06-26', '2018-06-27', '2018-06-28', '2018-06-29',
               '2018-07-02', '2018-07-03', '2018-07-04', '2018-07-05',
               '2018-07-06', '2018-07-09'],
              dtype='datetime64[ns]', name='Date', length=4673,
freq=None)y = df['Closing Value'].resample('MS').mean()
y.plot(figsize=(15, 6))
plt.show()

rcParams['figure.figsize'] = 18, 8
decomposition = sm.tsa.seasonal_decompose(y,
model='additive')fig = decomposition.plot()
plt.show()

sc = MinMaxScaler(feature_range =
(0, 1))df = sc.fit_transform(df)
```

TRAINING AND TESTING

```
train_size = int(len(df) * 0.70)
test_size = len(df) - train_size
train, test = df[0:train_size, :], df[train_size:len(df), :]def
create_data_set(_data_set, _look_back=1):
    data_x, data_y = [], []
    for i in range(len(_data_set) - _look_back -
        1):a = _data_set[i:(i + _look_back), 0]
        data_x.append(a)
        data_y.append(_data_set[i + _look_back,
    0])return np.array(data_x), np.array(data_y)
look_back =90
X_train,Y_train,X_test,Ytest = [],[],[],[]
X_train,Y_train=create_data_set(train,look_back)
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1],
1))X_test,Y_test=create_data_set(test,look_back)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
```

LSTM LAYER

```
regressor = Sequential()
regressor.add(LSTM(units = 60, return_sequences = True,
input_shape =(X_train.shape[1], 1)))
regressor.add(Dropout(0.1))
regressor.add(LSTM(units = 60, return_sequences =
True))regressor.add(Dropout(0.1))
regressor.add(LSTM(units =
60))
regressor.add(Dropout(0.1))
regressor.add(Dense(units = 1))
regressor.compile(optimizer = 'adam', loss =
'mean_squared_error')reduce_lr =
ReduceLROnPlateau(monitor='val_loss',patience=5) history
=regressor.fit(X_train, Y_train, epochs = 20, batch_size =
15,validation_data=(X_test, Y_test),
callbacks=[reduce_lr],shuffle=False)Epoch 1/20
212/212 [==============================] - 23s 88ms/step - loss:
0.0047 - val_loss:
0.0251 - lr: 0.0010
Epoch 2/20
```

```
212/212                            - 17s 82ms/ste  - loss:  0.012  - val_loss:
[============================                 p              2
===]
0.0478 - lr: 0.0010
Epoch 3/20
212/212                            - 17s 82ms/ste  - loss:  0.011  - val_loss:
[===========================                  p              5
====]
0.0505 - lr: 0.0010
Epoch 4/20
212/212                            - 17s 81ms/ste  - loss:  0.016  - val_loss:
[===========================                  p              3
====]
0.0461 - lr: 0.0010
Epoch 5/20
212/212                            - 19s 91ms/ste  - loss:  0.019  - val_loss:
[===========================                  p              3
====]
0.0461 - lr: 0.0010
Epoch 6/20
212/212                            - 17s 82ms/ste  - loss:  0.017  - val_loss:
[===========================                  p              4
====]
0.0605 - lr: 0.0010
Epoch 7/20
212/212                            - 18s 83ms/ste  - loss:  0.027  - val_loss:
[===========================                  p              5
====]
0.0047 - lr: 1.0000e-04
Epoch 8/20
212/212                            - 18s 83ms/ste  - loss:  0.004  - val_loss:
[===========================                  p              0
====]
0.0032 - lr: 1.0000e-04
Epoch 9/20
212/212                            - 17s 82ms/ste  - loss:  0.002  - val_loss:
[===========================                  p              9
====]
0.0021 - lr: 1.0000e-04
Epoch 10/20
212/212                            - 17s 81ms/ste  - loss:  0.002  - val_loss:
[===========================                  p              3
====]
0.0017 - lr: 1.0000e-04
Epoch 11/20
212/212                            - 17s 83ms/ste  - loss:  0.002  - val_loss:
[===========================                  p              0
====]
0.0016 - lr: 1.0000e-04
Epoch 12/20
212/212                            - 17s 82ms/ste  - loss:  0.001  - val_loss:
[===========================                  p              6
====]
0.0015 - lr: 1.0000e-04
Epoch 13/20
212/212                            - 17s 83ms/ste  - loss:  0.001  - val_loss:
[===========================                  p              4
====]
0.0014 - lr: 1.0000e-04
```

```
Epoch 14/20
212/212                           - 18s 83ms/ste  - loss:  0.001  - val_loss:
[============================          p                 3
====]
0.0014 - lr: 1.0000e-04
Epoch 15/20
212/212                           - 18s 83ms/ste  - loss:  0.001  - val_loss:
[============================          p                 2
====]

0.0013 - lr: 1.0000e-04
Epoch 16/20
212/212                           - 18s 84ms/ste  - loss:  0.001  - val_loss:
[============================          p                 1
====]
0.0014 - lr: 1.0000e-04
Epoch 17/20
212/212                           - 18s 86ms/ste  - loss:  0.001  - val_loss:
[============================          p                 1
====]
0.0014 - lr: 1.0000e-04
Epoch 18/20
212/212                           - 19s 87ms/ste  - loss:  0.001  - val_loss:
[============================          p                 1
====]
0.0015 - lr: 1.0000e-04
Epoch 19/20
212/212                           - 17s 82ms/ste  - loss:  0.001  - val_loss:
[============================          p                 1
====]
0.0013 - lr: 1.0000e-05
Epoch 20/20
212/212                           - 18s 83ms/ste  - loss:  0.001  - val_loss:
[============================          p                 0
====]
0.0013 - lr: 1.0000e-05
```

MODEL TRAINING

```
train_predict =
regressor.predict(X_train)test_predict =
regressor.predict(X_test)
100/100 [==============================] - 4s 27ms/step
41/41 [==============================] - 1s 28ms/step
train_predict =
sc.inverse_transform(train_predict)Y_train =
sc.inverse_transform([Y_train]) test_predict =
sc.inverse_transform(test_predict)
```

PREDICTION

```
print('Train Mean Absolute Error:', mean_absolute_error(Y_train[0],
train_predict[:,0]))
print('Train Root Mean Squared Error:',np.sqrt(mean_squared_error(Y_train[0],
train_predict[:,0])))
print('Test Mean Absolute Error:', mean_absolute_error(Y_test[0],
test_predict[:,0]))
print('Test Root Mean Squared Error:',np.sqrt(mean_squared_error(Y_test[0],
test_predict[:,0])))
plt.figure(figsize=(8,4)) plt.plot(history.history['loss'],
label='Train Loss') plt.plot(history.history['val_loss'],
label='Test Loss')plt.title('model loss')
plt.ylabel('loss') plt.xlabel('epochs')
plt.legend(loc='upper right')
plt.show();
Train Mean Absolute Error: 2.3165036988408305
Train Root Mean Squared Error: 3.285617879896689
Test Mean Absolute Error: 2.3989636110004624 Test
Root Mean Squared Error: 5.289593391043789

aa=[x for x in range(180)]
plt.figure(figsize=(8,4))
plt.plot(aa, Y_test[0][:180], marker='.', label="actual") plt.plot(aa,
test_predict[:,0][:180], 'r', label="prediction")plt.tight_layout()
sns.despine(top=True)
plt.subplots_adjust(left=0.07)
plt.ylabel('Price', size=15)
plt.xlabel('Time step', size=15)
plt.legend(fontsize=15) plt.show();
```

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-38682-1660384421