CHAPTER 1
INTRODUCTION

1.1 PROJECT OVERVIEW

Crude oil is a yellow-black naturally occurring liquid found in geological formations beneath the Earth's surface, it can be separated into various kinds of consumer fuels through the process of fractional distillation. Crude oil is amongst the most important energy resources on earth right now. So far, it remains the world's leading fuel, with nearly one-third of global energy consumption. Petroleum products are also made of refined crude oil. Encouraging usage of fossil fuels is getting highly unpopular as they're irrefutably responsible for global warming, and other severe impacts on ecosystems. We call crude oil and petroleum as Fossil fuels because they are mixtures of hydrocarbons that formed from the remains of animals and plants (diatoms) that lived millions of years ago in a marine environment before the existence of dinosaurs. Over millions of years, the remains of these animals and plants were covered by layers of sand, silt, and rock. Heat and pressure from these layers turned the remains into what we now call crude oil. Oil is the 3rd largest exported product and is 3.82% of the global trade in the world. The modern oil industry can trace its origins to Baku in 1837, where the first commercial oil refinery was established to distil oil into paraffin (used as lamp and heating oil). This was followed by the first modern oil well in1846, which reached a depth of 21 meters. It is believed that global oil reserves (1.67 trillion barrels of oil globally) will be exhausted sometime between 2050 and 2060.

Historically, energy resources are of strategic importance for social welfare and economic growth. So, predicting crude oil price fluctuations is an important issue. Since crude oil price changes are affected by many risk factors in markets, this price shows more complicated nonlinear behavior and creates more risk levels for investors than in the past. We propose a new method of prediction of crude oil price to model nonlinear dynamics. This study aims to present time series-based forecasting for crude oil prices using neural network algorithms. Daily prices of oil and currency exchange rates are tested as input features, in addition to crude oil prices. Efforts are focused on finding the optimal network structures for the modeling of crude palm oil price forecasting. Neural network structures with an appropriate selection of input and hidden nodes are tested.

1.2 PURPOSE

To maintain proper economy knowing crude oil price earlier became indispensable. In this project we will forecast the future crude oil price using Artificial Neural Network models and analyse the impact of production, consumption, exports and imports of crude oil. In this we will understand the similarity in the extent of impact of crude-oil prices on the economy. 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. Trading will be taken in efficient and cautiously way. Demand and supply will be functioned without any hindrances. Hence, Country Economy will gradually rise.

CHAPTER 2
LITERATURE SURVEY

2.1 EXISTING SOLUTION

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. We propose a contemporary and innovative method of predicting crude oil prices using the artificial neural network (ANN). The main advantage of this approach of ANN is that it continuously captures the unstable pattern of the crude oil prices which have been incorporated by finding out the optimal lag and number of the delay effect that controls the prices of crude oil. Variation of lag in a period of time has been done for the most optimum and close results, we then have validated our results by evaluating the root mean square error and the results obtained using the proposed model have significantly outperformed.

2.2 REFERENCES

- K. Kanchymalay, R. Sallehuddin, N. Salim and U. R. Hashim, "Time series-based forecasting for crude palm oil price utilizing neural network algorithms," 2017 6th ICT International Student Project Conference (ICT-ISPC), 2017, pp. 1-4, doi: 10.1109/ICT-ISPC.2017.8075334.
- C. Zhang, F. Jiang, S. Wang and W. Shang, "A Novel Hybrid Approach with A Decomposition Method and The RVFL Model for Crude Oil Price Prediction," 2020 IEEE International Conference on Big Data (Big Data), 2020, pp. 4446-4449, doi: 10.1109/BigData50022.2020.9377826.
- M. M. L. Heravi, M. Khorrampanah and M. Houshmand, "Forecasting Crude Oil Prices Using Improved Deep Belief Network (IDBN) and Long-Term Short-Term Memory Network (LSTM)," 2022 30th International Conference on Electrical Engineering (ICEE), 2022, pp. 823-826, doi: 10.1109/ICEE55646.2022.9827452.

2.3 PROBLEM STATEMENT DEFINITION

Because demand for oil is inelastic, a rise in price is good news for producers because it will boost their earnings. Oil importers, on the other hand, will face higher oil prices. Because oil is the most traded commodity, the consequences are considerable. Rising oil prices may even shift economic and political influence away from oil consumers and toward oil exporters. Crude oil price changes are influenced by a variety of factors.

CHAPTER 3
IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

An empathy map is a collaborative visualization used to express clearly what one knows about a particular type of user. It externalizes knowledge about users in order to create a shared understanding of user needs, and aid in decision making.

Empathy maps are split into 4 quadrants (Says, Thinks, Does, and Feels), with the user in the middle. Empathy maps provide a glance into who a user is as a whole. The ***Says*** quadrant contains what the user says or what he needs. The ***Thinks*** quadrant captures what the user is thinking throughout the experience. The ***Does*** quadrant encloses the actions the user takes. The ***Feels*** quadrant is the user's emotional state.

The empathy map for Crude Oil Price Prediction is shown in Fig 3.1

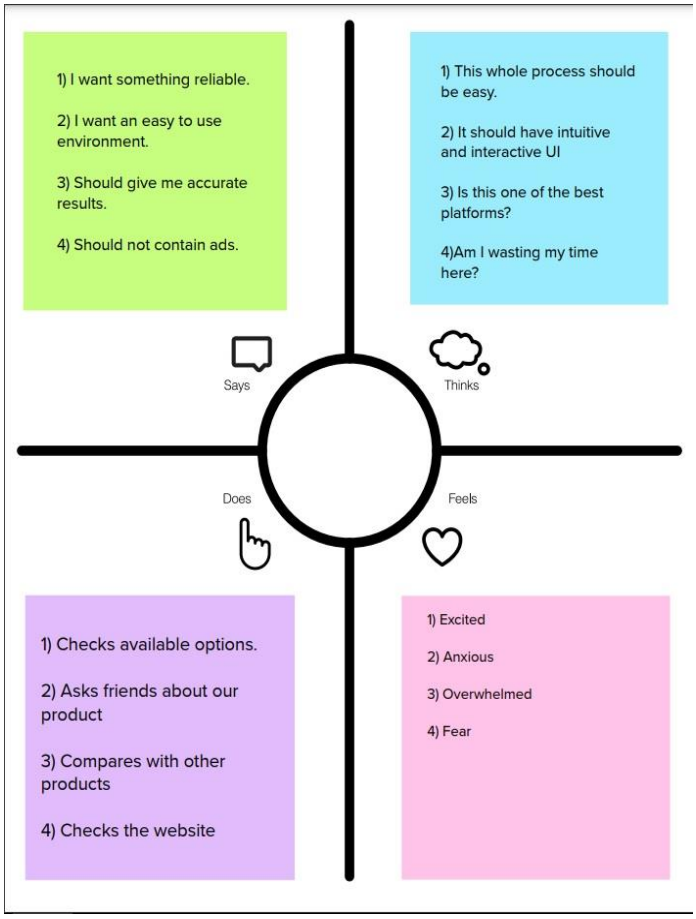

Fig 3.1 Empathy map

3.2 IDEATION AND BRAINSTORMING

Ideation is often closely related to the practice of brainstorming, a specific technique that is utilized to generate new ideas. Brainstorming is usually conducted by getting a group of people together to come up with either general new ideas or ideas for solving a specific problem or dealing with a specific situation. A principal difference between ideation and brainstorming is that ideation is commonly more thought of as being an individual pursuit, while brainstorming is almost always a group activity. Both brainstorming and ideation are processes invented to create new valuable ideas, perspectives, concepts and insights, and both are methods for envisioning new frameworks and systemic problem solving.

The Brainstorming chart for Crude Oil Price Prediction is shown in Fig 3.2.

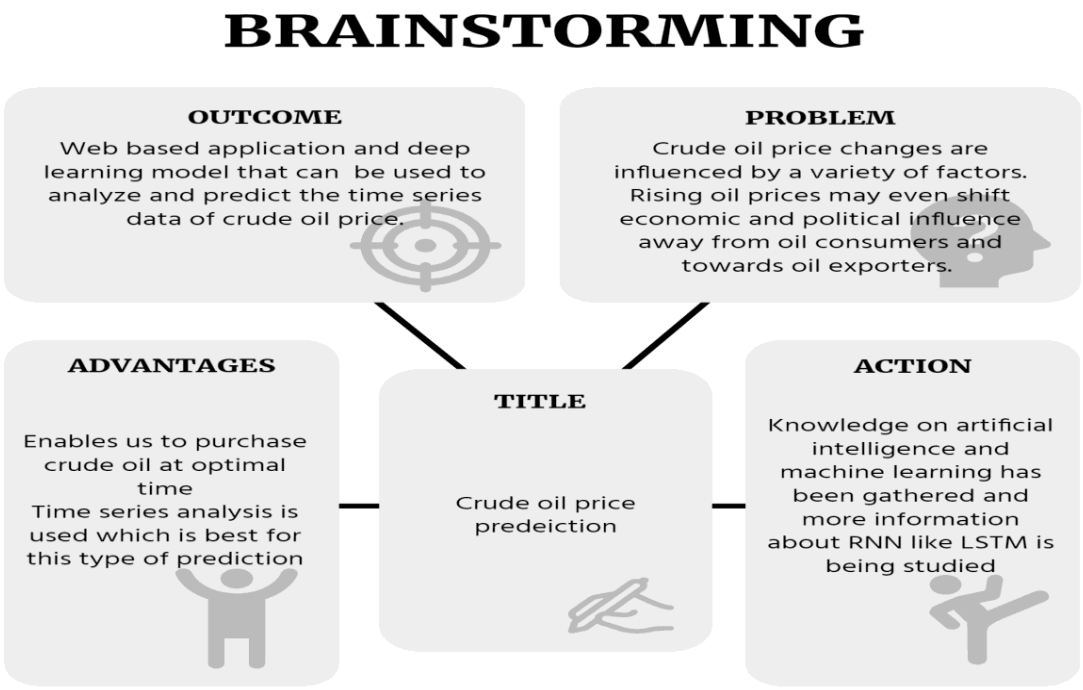

Fig 3.2 Ideation and Brainstorming

3.3 PROPOSED SOLUTION

The proposed solution for Crude Oil Price Prediction is shown in table 3.1

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The price of crude oil, the most important fuel in the world, has a significant impact on the environment globally, thus forecasts are extremely useful for governments, corporations, and individuals. Continuous application of statistical and econometric tools, including AI, may reduce forecast accuracy.
2.	Idea / Solution description	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. So, we would be implementing RNN (Recurrent Neural Network) to achieve the task. The mean squared error is used to calculate the cost's effectiveness. The suggested model's performance is evaluated using pricing information from WTO crude oil materials.
3.	Novelty / Uniqueness	The uniqueness of our project lies in the RNN model that we use to solve this problem. We have chosen LSTM to predict the oil prices. LSTMs are good when it comes to performing time series analysis.
4.	Social Impact / Customer Satisfaction	Our model helps customers to buy crude oil at the proper time and gives proper insights about crude oil prices. It is used to forecast future prices and utilise oil in accordance with such predictions. This price has a direct impact on a variety of goods and products, and its fluctuations have an impact on the stock markets. Significant events, in addition to economic causes, can have an impact on oil prices.
5.	Business Model (Revenue Model)	It can be useful for decision-makers who are businesses, individual investors, or both when deciding whether to buy or sell crude oil. Crude oil is one of the most profitable commodities to

		trade. The benchmark model for predicting crude oil prices is RNN and LSTM.
6.	Scalability of the Solution	The PCA, MDS, and LLE algorithms are used to minimise the dimensionality of the data. The accuracy of RNN and LSTM models should be improved.

Table 3.1 Proposed Solution

3.4 PROBLEM SOLUTION FIT

The Problem solution fit simply means that one has found a problem with the customer and that the solution one has realised for it actually solves the customers problem. The problem solution fit is an important step towards the Product-Market Fit. The structure of problem solution fit is given below.

Customer state fit: To make sure one understands the target group, their limitations and their currently available solutions, against which one is going to compete.

Problem-Behavior fit: To help one to identify the most urgent and frequent problems, understand the real reasons behind them and see which behavior supports it.

Communication-Channel fit: To help one to sharpen the communication with strong triggers, emotional messaging and reaching customers via the right channels.

Solution guess: Translate all the validated data one has gathered into a solution that fits the customer state and his/her limitations, solves a real problem and taps into the common behavior of the target group.

Crude oil is one of the most important resources in today's world; it is the primary fuel, and the price of crude oil has a direct impact on the global environment, our economy, and oil exploration, exploitation, and other activities. Prediction of oil prices has become a necessity; it benefits many large and small enterprises, individuals, and the government. Because of the evaporative nature of crude oil, predicting its price becomes incredibly complex and difficult. There are numerous elements that influence crude oil pricing. Using an artificial neural network, we offer a modern and unique approach of predicting crude oil

prices (ANN). The key advantage of this ANN technique is that it continually captures the volatile pattern of crude oil prices that have been incorporated by determining the ideal lag and number of the delay effect that regulates crude oil prices. We then tested our results by assessing the root mean square error and found that the results produced using the suggested model greatly beat the results obtained using the previous model.

The problem solution fit for Crude Oil Price Prediction is shown in Fig 3.3

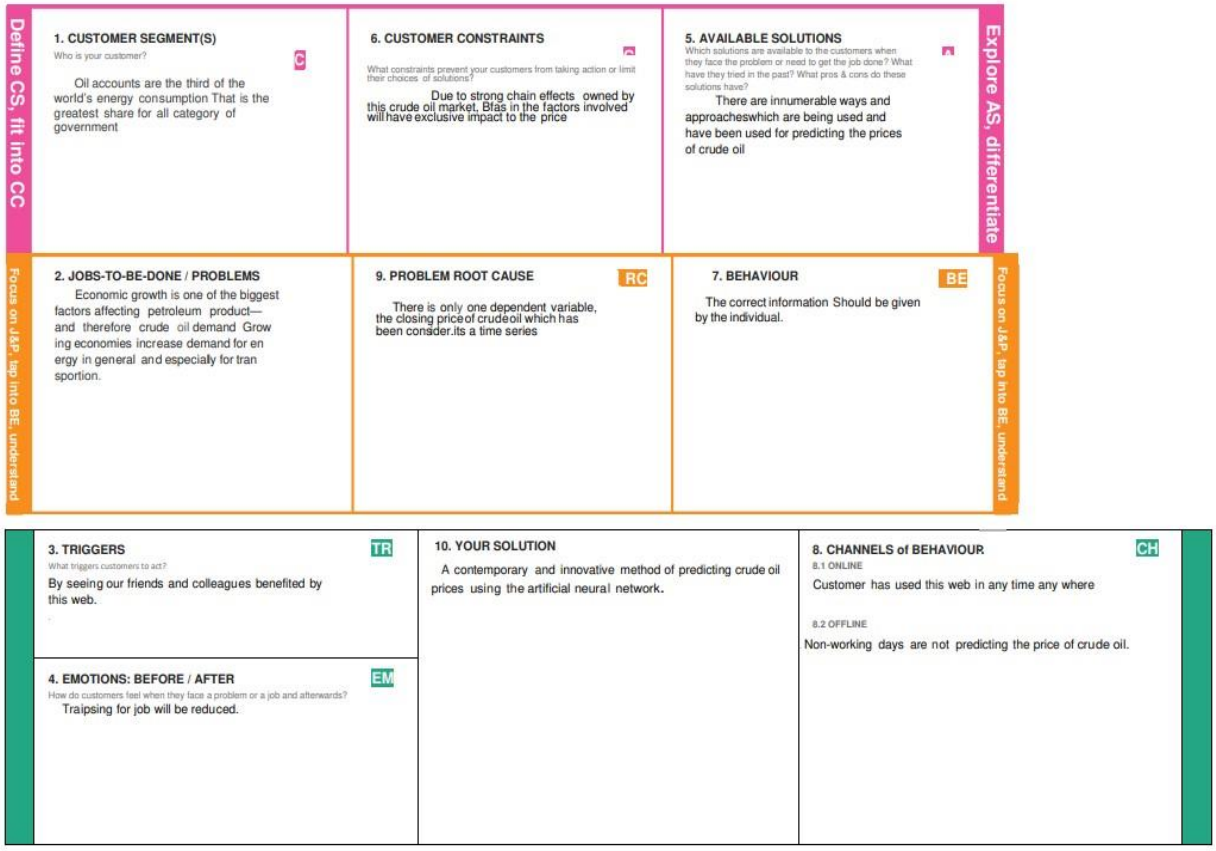

Fig 3.3 Problem Solution fit

CHAPTER 4
REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

The functional requirements for Crude Oil Price Prediction are shown in Table 4.1

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User registration	The price of crude oil can be downloaded directly from the Google Play Store application by users.
FR-2	User Confirmation	The Application's User There are numerous products in the Crude Oil Price App, and users can instantly update the energy and oil prices.
FR-3	Interface sensor	The user can view oil price charts and the most recent news. Major Energy Quotes User View The user may use various colour schemes.
FR-4	Accessing datasets	The User Can Exchange Rates and Currency Converter

Table 4.1 Functional Requirements

4.2 NON-FUNCTIONAL REQUIREMENTS

The non-functional requirements for Crude Oil Price Prediction are shown in Table 4.2

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	used to increase crude oil price prediction accuracy
NFR-2	**Security**	Communications will be secure even if rising oil prices cause economic and political power to shift from oil importers to oil exporters.
NFR-3	**Reliability**	The dependability of the components pointing to high-risk components
NFR-4	**Performance**	Performance of this project is to improve to the accuracy of crude oil price prediction
NFR-5	**Availability**	The Availability Solution is More Benefit for and the Importers and exporters in the crude oil price prediction.
NFR-6	**Scalability**	The scalability is 90%-95%

Table 4.2 Non-functional requirements

CHAPTER 5
PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

The data flow diagram for Crude Oil Price Prediction is shown in Fig 5.1

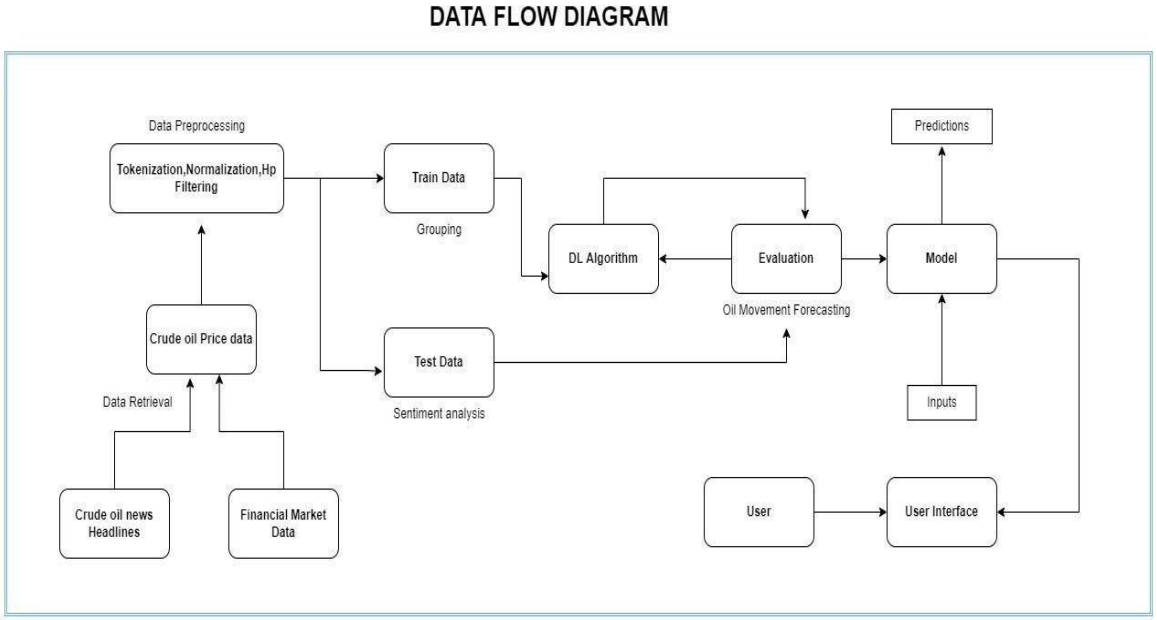

Fig 5.1 Data flow diagram

5.2 SOLUTION AND TECHNICAL ARCHITECTURE

The Deliverable shall include the architectural diagram for Crude Oil Price Prediction as shown in Fig 5.2

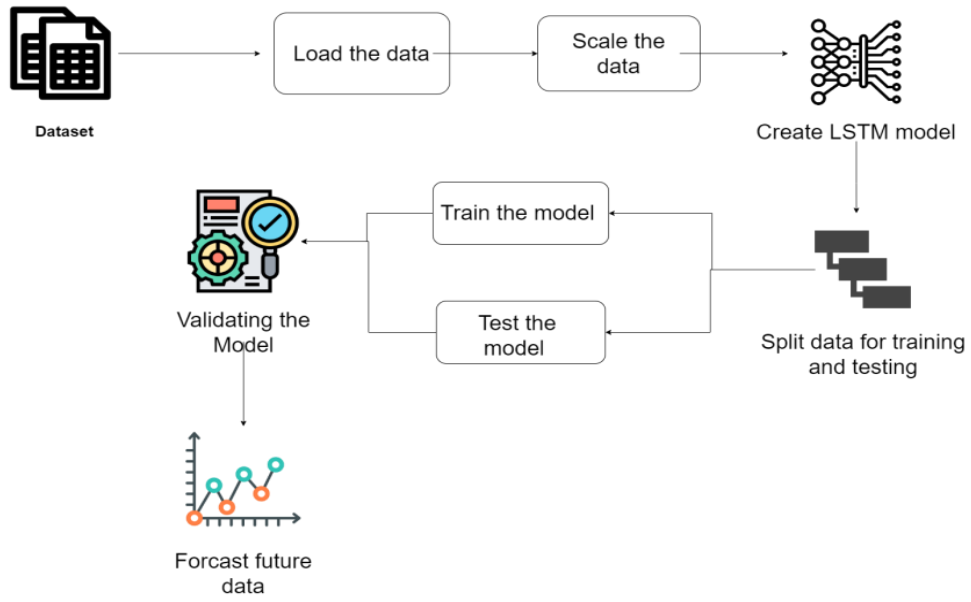

Fig 5.2 Technical Architecture

The components and technologies used for Crude Oil Price Prediction are shown in Table 5.1

S.No	Component	Description	Technology
1.	User Interface	Web ui	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Oil price prediction	Deep learning/Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem

8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc.
9.	External API-2	Purpose of External API used in the application	Aadhar API, etc.
10.	Machine Learning Model	Purpose of Machine Learning Model	Object Recognition Model, etc.
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration:	Local, Cloud Foundry, Kubernetes, etc.

Table 5.1 Components and Technologies

The application characteristics for Crude Oil Price Prediction are shown in Table 5.2

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Technology of Opensource framework
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g., SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Technology used
4.	Availability	Justify the availability of application (e.g., use of load balancers, distributed servers etc.)	Technology used
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Technology used

Table 5.2 Application characteristics

The user task and the acceptance criteria for Crude Oil Price Prediction is shown in Table 5.3

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 the password.	I can access my account/ Displays Line graph / Bar graph.	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I 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 my Account	Low	Sprint-2
	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 graph		After entering the inputs, the model will display predictions in Line\Bar Graph 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 so that it can be user interactive.	I can solve the problems raised by Support.	Low	Sprint-3

Administrator	News		Admin will give the recent news of crude oil prices.	Provide the recent crude oil prices.	High	Sprint-4
	Notification		Admin will notify when the crude oil price changes.	Notification by Gmail regarding the price fluctuations of crude oil.	High	Sprint-4
	Access Control		Admin can control the access of the users.	Access permission for users.	High	Sprint-4
	Database		Admin can store the details of users.	Stores user details.	High	Sprint-4

Table 5.3 User task and Acceptance criteria

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Sprint planning is an event in scrum that kicks off the sprint. The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved.

The sprint is a set period of time where all the work is done. However, before leap into action it is necessary to set up the sprint. It needs to decide on how long the time box is going to be, the sprint goal, and where it is going to start. The sprint planning session kicks off the sprint by setting the agenda and focus. If done correctly, it also creates an environment where the team is motivated, challenged, and can be successful.

The Table 6.1 shows the sprint planning and estimation of Crude Oil Price Prediction.

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	10	High
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	10	High
Sprint-1	Login	USN-3	As a user, I can log into the application by entering email & password.	15	High
Sprint-2	Input Necessary Details	USN-4	As a user, I can give Input Details to Predict Likeliness of crude oil	15	High
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password.	1	High
Sprint-2	Dashboard Pre-processing	USN-6	Transform raw data into suitable format for prediction.	15	High
Sprint-3	Prediction of Crude Oil Price	USN-7	As a user, I can predict Crude oil using machine learning model.	20	High
Sprint-3		USN-8	As a user, I can get accurate prediction of crude oil	5	Medium
Sprint-4	Review	USN-9	As a user, I can give feedback of the application.	20	High

Table 6.1 Sprint planning and Estimation

6.2 SPRINT DELIVERY SCHEDULE

The sprint delivery plan is scheduled accordingly as shown in the below table 6.2 which consists of the sprints with respective to their duration, sprint start and end date and the releasing data.

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	20	6 Days	31 Oct 2022	05 Nov 2022	20	3 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Table 6.2 Sprint delivery schedule

CHAPTER 7

CODING AND SOLUTIONING

7.1 FEATURE 1

Importing the libraries:

The required libraries to be imported to Python script are:

NumPy

NumPy is an open-source numerical Python library. It contains a multi-dimensional array and matrix data structures. It can be used to perform mathematical operations on arrays such as trigonometric, statistical, and algebraic routines.

Pandas

Pandas is a fast, powerful, flexible, and easy to use open-source data analysis and manipulation tool, built on top of the Python programming language.

Seaborn

Seaborn is a library that uses Matplotlib underneath to plot graphs. It will be used to visualize random distributions.

Matplotlib

Matlab is a visualization with python. It is a comprehensive library for creating static, animated, and interactive visualizations in Python.

The above libraries can be imported by using following syntax shown in Fig 7.1

```
In [2]:
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Fig 7.1 Libraries required

Importing the dataset:

The dataset will be in the format of .xlsx or .csv (Crude Oil Price Daily.xlsx) is used to load the excel file into pandas using read_excel() function. The file directory of the excel file is given in the read_excel function to import the dataset in the Fig 7.2

```
In [1]: import pandas as pd

In [2]: ds=pd.read_excel(r"C:\Users\Dhyalan\Desktop\Crude Oil Prices Daily1.xlsx")

In [3]: ds.shape

Out[3]: (8223, 2)

In [4]: ds.head()
```

Out[4]:

	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

Fig 7.2 Importing dataset

Handling missing data:

It is important to check for any null values in any row or a column. The total number of null values in the closing function can be calculated by using isnull(). sum (). The checking for null values before dropping is shown in Fig 7.3.

```
In [8]: ds.isnull().sum()

Out[8]: Date            0
        Closing Value   7
        dtype: int64
```

Fig 7.3 Null values before dropping

The null records found in the closing function can be eliminated or dropped by using dropna() which is assigned to a different variable ‘hd’ which is shown in Fig 7.4

```
In [11]: hd=ds.dropna()
```

Fig 7.4 Dropping the null values

After dropping the null values from the closing function, we can find that there are no null values to be found when sum () function is used which can be inferred from the Fig 7.5 below.

```
In [12]: hd.isnull().sum()

Out[12]: Date            0
         Closing Value   0
         dtype: int64
```

Fig 7.5 Null values after dropping

Feature scaling:

Feature scaling is a method used to normalize the range of independent features or variables of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step. Scaling the features makes the flow of gradient descent smooth and helps algorithms quickly reach the minima of the cost function. The crude oil prices between (0,1) is scaled to avoid intensive computation using standardization and normalization.

MinMaxScaler

It is used to transform features by scaling each feature to a given range. This estimator scales and translates each feature individually such that it is in the given range on the training set the range of features to scale the range in [0, 1] or [−1, 1].

The results are shown in Fig 7.6 after MinMax scaling.

```
In [36]: min_max_scaler = preprocessing.MinMaxScaler(feature_range =(0, 1))
         x_after_min_max_scaler = min_max_scaler.fit_transform(x)
         print ("\nAfter min max Scaling : \n", x_after_min_max_scaler)

After min max Scaling :
 [[0.66062503]
 [0.66187012]
 [0.66685047]
 ...
 [0.65511108]
 [0.65395493]
 [0.68294765]]
```

Fig 7.6 Results after min-max scaling

StandardScaler

Each feature or variable is scaled to unit variance once the mean has been removed by StandardScaler. This process is carried out independently based on feature. Since StandardScaler involves the estimation of the empirical mean and standard deviation of each feature, outliers can have an impact on it.

The results after standardization are shown in Fig 7.7

```
In [37]: Standardisation = preprocessing.StandardScaler()
         x_after_Standardisation = Standardisation.fit_transform(x)
         print ("\nAfter Standardisation : \n", x_after_Standardisation)

After Standardisation :
 [[0.86576727]
 [0.87197137]
 [0.89678778]
 ...
 [0.83829196]
 [0.83253101]
 [0.97699794]]
```

Fig 7.7 Results after Standardization

Data visualization:

The graphic display of information and data is known as data visualization. Data visualization tools offer an easy approach to observe and analyze trends, outliers, and patterns in data by utilizing visual elements like charts, graphs, and maps. The dataset can be visualized by using matplotlib and seaborn. The data visualization is done in the following ways

stripplot()

Stripplot is a graphic data analysis approach that is used to summarize the univariate data collection. It has a horizontal axis with the response variable's value and a vertical axis with all values set to one is shown in Fig 7.8

```
In [14]: sns.stripplot(y='Closing Value',data=ds)
```

Out[14]:

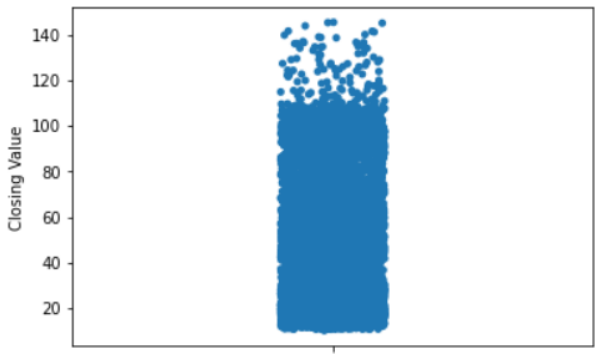

Fig 7.8 Data visualization using stripplot

plt.scatter()

In an effort to demonstrate how much one variable is influenced by another, scatter () function is used to plot data points on horizontal and vertical axes. The position of each marker in the data table, which represents each row, is determined by the values of the columns that are specified on the X and Y axes is shown in Fig 7.9

```
In [16]: plt.scatter(ds.index,ds['Closing Value'])
         plt.show()
```

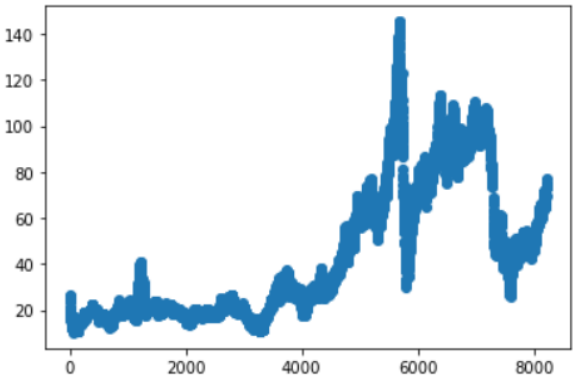

Fig 7.9 Data visualization using scatter

plt.hist()

In Matplotlib, we use the hist () function to create histograms is shown in Fig 7.10. The hist () function will use an array of numbers to create a histogram, the array is sent into the function as an argument.

```
In [18]: plt.hist(ds['Closing Value'])

Out[18]: (array([3372., 1304.,  794.,  744.,  585.,  470.,  692.,  182.,   45.,
                  28.]),
          array([ 10.25 ,  23.756,  37.262,  50.768,  64.274,  77.78 ,  91.286,
                 104.792, 118.298, 131.804, 145.31 ]),
          )
```

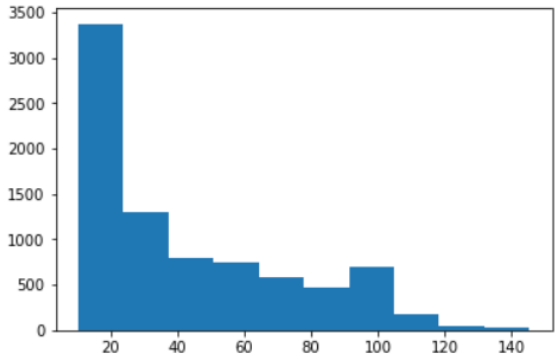

Fig 7.10 Data visualization using hist

From the graphs we can infer that the price of crude oil changes day by day and is increasing with significant drop in some years.

Splitting data into train and test:

A dataset is used to train a model. However, the testing of the model is done on some test dataset after training. A dataset that is distinct from the training set previously utilized is used for testing. However, having so much data available throughout the development stage might not always be practical. For this the dataset can then be divided into two sets: one for training and the other for testing.

train_test_split()

The train_test_split() method is used to split our data into train and test sets which is imported from the library sklearn.model_selection is shown in Fig 7.11. The data is divided into features (X) and labels (y). The data frame gets divided into X_train, X_test, y_train, and y_test. X_train and y_train sets are used for training and fitting the model. The X_test and y_test sets are used for testing the model if it's predicting the right outputs/labels.

```
In [62]: training_size=int(len(data)*0.65)
         test_size=len(data)-training_size
         train_data,test_data=data[0:training_size,:],data[training_size:len(data),:1]
```

Fig 7.11 Splitting data to train and test

The data can be trained and test using the following syntax is shown in Fig 7.12

```
In [63]: training_size,test_size

Out[63]: (5340, 2876)

In [64]: train_data.shape

Out[64]: (5340, 1)
```

Fig 7.12 Size of train and test data

The data is split in such a way that 5340 values (i.e., 65% of the values available are utilized) are used train the model and the remaining 2876 values are used to test the model.

Creating a dataset with sliding window:

The function takes two arguments, the dataset which is a NumPy array that we want to convert into a dataset and the time_step which is the number of previous time steps to use as input variables to predict the next time period, in this case, defaulted to 1.

This default will create a dataset where X is the price of crude oil at a given time (t) and Y is the price of crude oil at the next time (t + 1).

The creation of dataset with sliding window is shown in Fig 7.13

```
In [65]:
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)
```

Fig 7.13 Creating dataset with sliding window

The function is applied on the training data and test data is shown in Fig 7.14

```
In [66]:
time_step=10
x_train,y_train=create_dataset(train_data,time_step)
x_test,y_test=create_dataset(test_data,time_step)
```

Fig 7.14 Training data and test data

After applying the function, the shape of the training data is found and it is clearly shown in Fig 7.15

```
In [67]:
print(x_train.shape),print(y_train.shape)

(5329, 10)
(5329,)
Out[67]: (None, None)

In [68]:
print(x_test.shape),print(y_test.shape)

(2865, 10)
(2865,)
Out[68]: (None, None)
```

Fig 7.15 Shaping training data and test data

The data of x_train is as follows and it is shown in Fig 7.16

```
In [69]: x_train

Out[69]: 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]])
```

Fig 7.16 Data of x_train

The data of x_test is as follows and it is shown in Fig 7.17

```
In [70]: x_test

Out[70]: 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]])
```

Fig 7.17 Data of x_test

It is important to reshape the x_train and x_test into 3-dimensional array before building the LSTM model.

The reshaping of array is shown in Fig 7.18

```
In [71]: 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)
```

Fig 7.18 Reshaping of array

7.2 FEATURE 2

Import model building libraries:

The required libraries to be imported to Python script are:

Sequential

For a simple stack of layers where each layer has precisely one input tensor and one output tensor, a sequential approach is suitable. When your model has several inputs or numerous outputs, a sequential model is inappropriate.

LSTM

Keras LSTM stands for the Long Short-Term Memory layer. An LSTM layer learns long-term dependencies between time steps in time series and sequence data. The layer produces additive interactions, which during training might enhance gradient flow over lengthy periods.

Dropout

In order to avoid overfitting, the Dropout layer randomly sets input units to 0 at a frequency of rate at each step during training. The total of all inputs is maintained by scaling up non-zero inputs by 1/ (1 - rate).

Dense

Dense Layer is used to classify image based on output from convolutional layers. Each Layer in the Neural Network contains neurons, which compute the weighted average of its input and this weighted average is passed through a non-linear function.

The libraries used for Model building is shown in Fig 7.19

In [33]:

```
import numpy as np
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers import Dropout
from keras.layers import Dense
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_absolute_error as mae
from sklearn.metrics import mean_squared_error as mse
from sklearn.metrics import r2_score as r2s
```

Fig 7.19 Libraries used for model building

Initializing the model:

The model is initialized using the functions that are defined previously is shown in Fig 7.20 and the values are plotted with day and price as x-axis and y-axis respectively is shown in Fig 7.21.

In [24]:

```
def plotCurve(x,y,xlable,ylabel,clabel):
    fig, ax = plt.subplots(figsize=(5, 3))
    fig.subplots_adjust(bottom=0.15, left=0.2)
    ax.plot(x,y,label=clabel)
    ax.set_xlabel(xlable)
    ax.set_ylabel(ylabel)
    plt.grid()
    ax.legend()
    plt.show()
```

Fig 7.20 Defining plotCurve()

In [25]:

```
def plotTwoCurves(x1,x2,y1,y2,xlable,ylabel,clabel1,clabel2):
    fig, ax = plt.subplots(figsize=(5, 3))
    fig.subplots_adjust(bottom=0.15, left=0.2)
    ax.plot(x1,y1,color='blue',label=clabel1)
    ax.plot(x2,y2,color='red',label=clabel2)
    ax.set_xlabel(xlable)
    ax.set_ylabel(ylabel)
    plt.legend()
    plt.show()
```

Fig 7.21 Defining plotTwoCurves()

The plots obtained using the above defined coding parts are shown in Fig 7.22

```
In [27]: plotCurve(index1,ds['Value'],'Time(Days)','Price','Price Series')
```

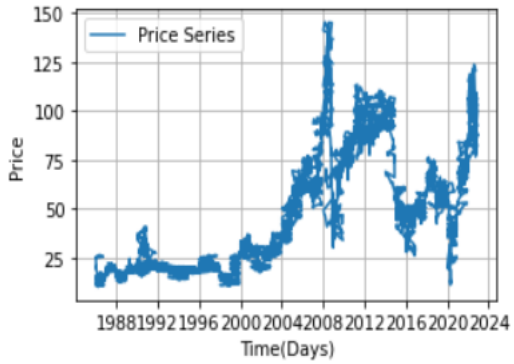

Fig 7.22 Plot for price series

From the above plot, it can be inferred that only the price values of crude oil for various time intervals are shown in Fig 7.23

```
In [29]: plotTwoCurves(index1,index1,ds['Value'],ds_price_scaled,'Time(Days)','Price','Price Series','Scaled Price')
```

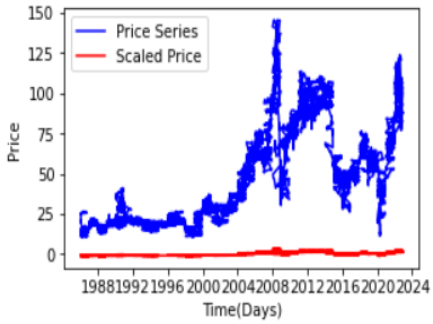

Fig 7.23 Plot for price given and scaled price

From the above plot, it can be inferred that the price values of crude oil from the dataset given as well as the scaled price for various time intervals is shown.

Adding LSTM layers:

The LSTM layers are added to the existing model is shown in Fig 7.24. Units is the number of neurons present in the layer which gives high model dimensionality that is enough to capture upward and downward trends.

In [13]:

```
model=Sequential()
model.add(LSTM(100, activation='relu', input_shape=(oilPX.shape[1], oilPX.shape[2]), return_sequences=True))
model.add(LSTM(50, activation='relu', return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(oilPY.shape[1]))
```

Fig 7.24 Adding LSTM layer

An epoch indicates the number of passes of the entire training dataset the machine learning algorithm has completed and it is shown in Fig 7.25

In [16]:

```
plotTwoCurves(index2,index2,history.history['loss'],history.history['val_loss'],'Epochs','Loss','Loss Series','Value loss Series')
```

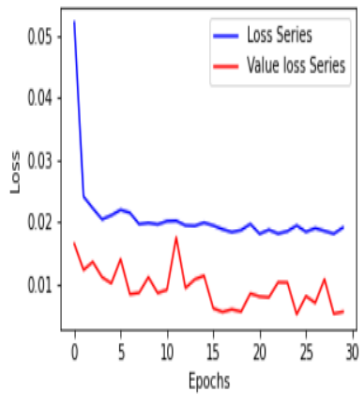

Fig 7.25 Plot for epoch and obtained loss

Understanding the model is a very important phase to properly use it for training and prediction purposes. The summary of the models and its layers used in keras is shown in Fig 7.26

In [14]:

```
model.compile(optimizer='adam',loss='mse')
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 14, 100)           40800

 lstm_1 (LSTM)               (None, 50)                30200

 dropout (Dropout)           (None, 50)                0

 dense (Dense)               (None, 1)                 51

=================================================================
Total params: 71,051
Trainable params: 71,051
Non-trainable params: 0
_________________________________________________________________
```

Fig 7.26 Summary of models used

Adding output layers:

The output layer is added with output dimension as 1 since we are predicting one price each time after adding the dense layers is shown in Fig 7.27 and the plot is shown in Fig 7.28.

In [18]:

```
predicted_data_NN=scaler.inverse_transform(predicted_data)
index3=range(0,len(predicted_data_NN))
plotTwoCurves(index3,index3,ds['Value'][-forecast_date:],predicted_data_NN,'Time(Days)','Price','Actual Oil Price','Predicted Oil Price')
```

Fig 7.27 Adding output layer

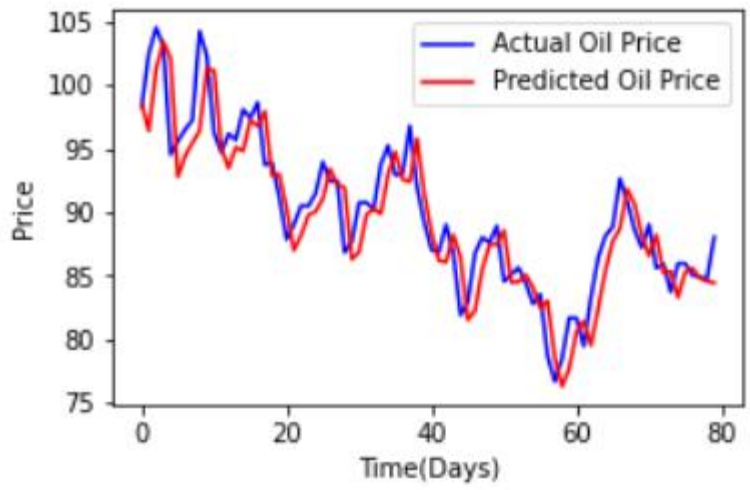

Fig 7.28 Plot after adding the output layer of predicted price

Model evaluation:

The model is evaluated to check its performance on the test data using regression evaluation metrics is shown in Fig 7.29 and predicted oil prices while model evaluation is shown in Fig 7.30

```
In [19]: index4=range(0,len(ds)-forecast_date)
         index5=range(len(ds)-forecast_date,len(ds))
         plotTwoCurves(index4,index5,ds['Value'][0:len(ds)-forecast_date],predicted_data_NN,'Time(Days)','Price','Actual Oil Price','Predicted Oil Price')
```

Fig 7.29 Model evaluation

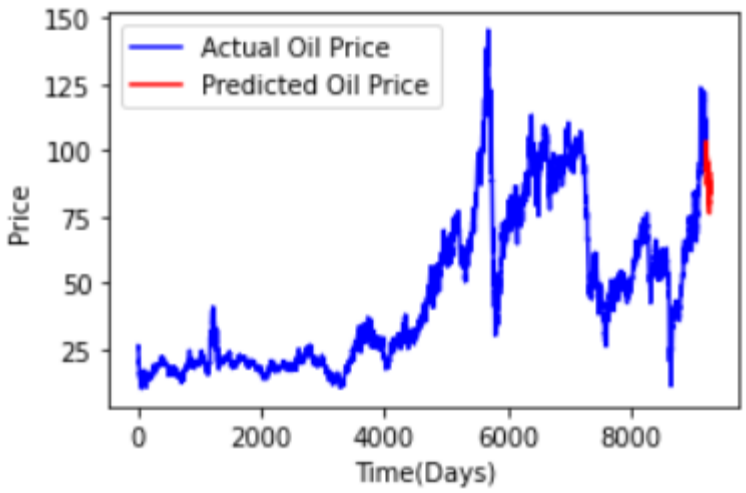

Fig 7.30 Predicted oil prices while model evaluation

Testing model:

The model is tested and the predicted output is plotted in the form of a graph between time and price is shown in Fig 7.31

```
In [21]: predict_index1=range(len(ds)-128,len(ds)-forecast_date)
         predict_index2=range(len(ds)-forecast_date,len(ds))
         plotTwoCurves(predict_index1,predict_index2,ds['Value'][len(ds)-128:len(ds)-forecast_date],predicted_data_NN,'Time(Days)','Price','Actual Oil Price','
```

Fig 7.31 Model testing

The model predicts the future values of the price of crude oil which can be seen in Fig 7.32

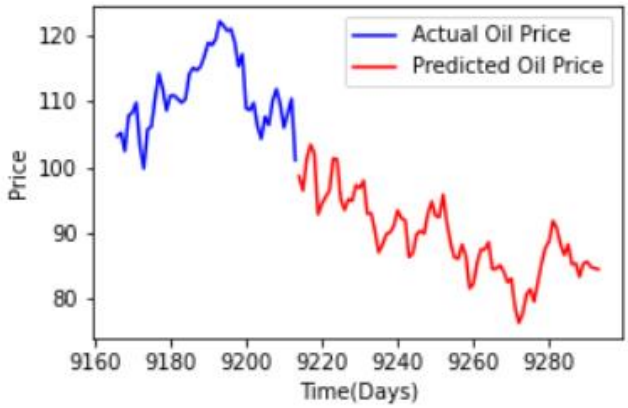

Fig 7.32 Predicted output

Saving the model:

The model created can be saved by the following method and it is shown in Fig 7.33

```
In [23]: model.save('saved_model/my_model')
```

Fig 7.33 Saving model

The model can also be saved in .h5 extension. An H5 file is a data file saved in the Hierarchical Data Format (HDF). It contains multidimensional arrays of scientific data.

CHAPTER 8

TESTING OF MODEL

From the below figure 8.1 and 8.2 it can be seen that actual oil price and the predicted oil price of the Crude Oil is shown with respect to number of days.

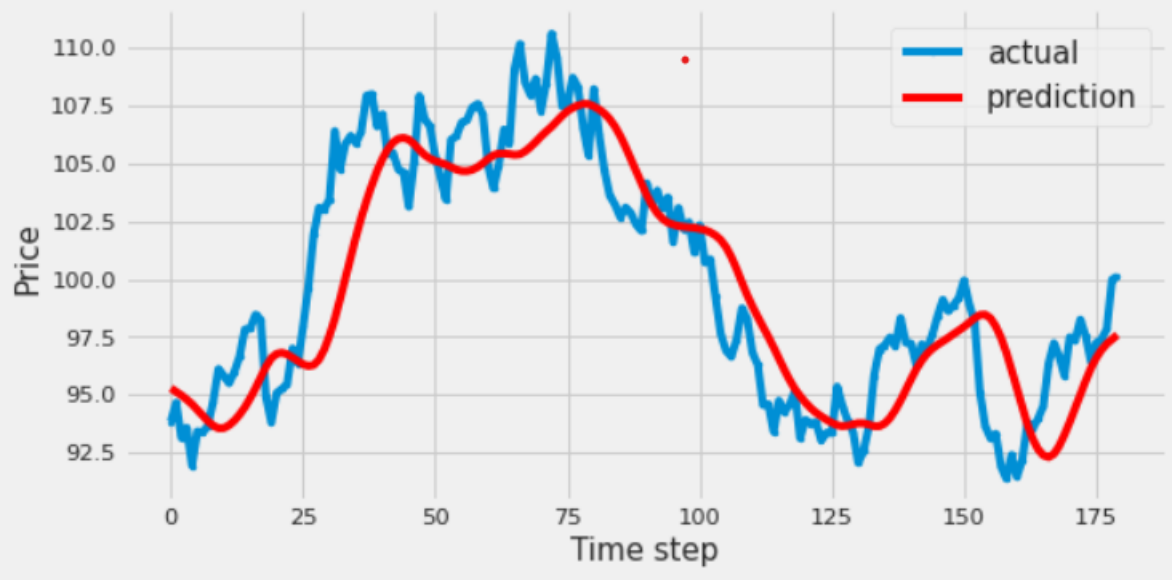

Fig 8.1 Actual oil price and predicted oil price with respect to number of days

Fig 8.2 The training loss and testing loss of the model

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICES

Mean Square Error, R-Squared and adjacent result are the three evaluation matrices in Crude Oil Price Prediction.

MEAN SQUARE ERROR

The Mean Squared Error (MSE) is perhaps the simplest and most common loss function, often taught in introductory Machine Learning courses. To calculate the MSE, you take the difference between your model's predictions and the ground truth, square it, and average it out across the whole dataset.

R SQUARED

R-squared is a statistical measure that represents the goodness of fit of a regression model. The ideal value for r-square is 1. The closer the value of r-square to 1, the better is the model fitted.

The evaluation metrices for Crude Oil Price Prediction is shown below in Fig 9.1

```
[ ]  mae1=mae(actual_data,predicted_data_NN)
     mse1=mse(actual_data,predicted_data_NN)
     print(mse1)
     rmse1=sqrt(mse1)
     r2_score=r2s(actual_data,predicted_data_NN)
     print(r2_score)
     n=forecast_date
     k=1
     adjacent_result=1-(((1-r2_score)*(n-1))/(n-k-1))
     print(adjacent_result)

     7.668108013724597
     0.7909621762566658
     0.7882822041573923
```

Fig 9.1 Evaluation metrices

Accuracy is also one of the performances metrices is shown in the Fig 9.2

```
[ ]  accuracy=100-(adjacent_result+r2_score+mse1)
     print("Accuracy = "'%.2f' %accuracy + " %")

     Accuracy = 90.75 %
```

Fig 9.2 Accuracy of the model

The website have been created for the prediction of crude oil and the user login is shown in Fig 9.3

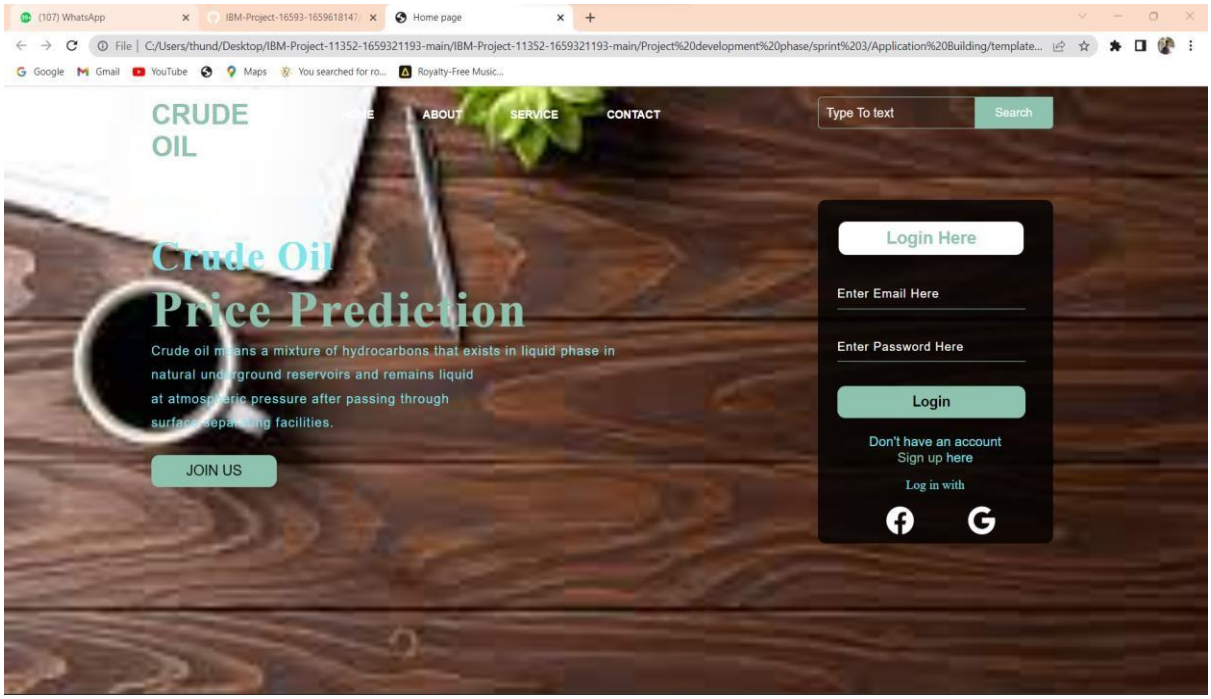

Fig 9.3 User login

The website created for the prediction of crude oil prices is shown in Fig 9.4. By using this website the user can enter the date in the specified area and click on submit and he will be provided with the predicted price at the space given in the image. This date can be in present, past or future format.

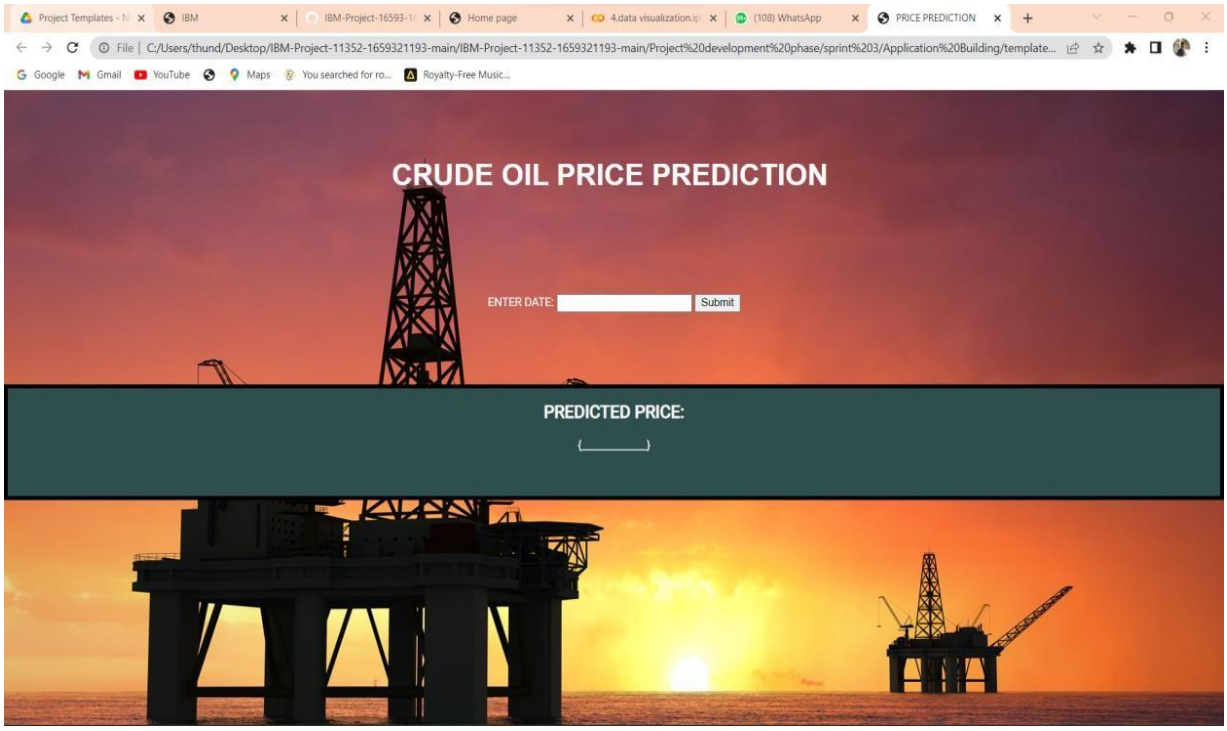

Fig 9.4 Website for the prediction of crude oil prices

CHAPTER 10

ADVANTAGES AND DISADVANTAGES

10.1 ADVANTAGES

Crude oil, also known as petroleum, is an energy-rich liquid consisting mainly of hydrocarbons. Oil is an important part of daily life in American households and all over the world. This powerful source of energy moves us, heats our homes and creates jobs and makes up an important component of everyday consumer products. All forms of energy solutions are needed to support a growing world population and improve the quality of life. Crude oil will play an important role in meeting these needs.

Oil supplies more than just fuel for our cars and heat for our homes. You'd be surprised. Modern life is inconceivable without crude oil. The world consumes almost 14 billion litres of oil each day. Within our daily lives oil is used almost everywhere.

The usage of crude oil could go on and on and on. Oil is a very important resource as one of major sources of energy. It can generate heat, drive machinery, and fuel vehicles and airplanes. Its components are used to manufacture almost all chemical products, such as plastics, detergents, paints, and even medicines.

10.2 DISADVANTAGES

Several oil companies have invested crores of dollars in oil exploration by taking bank loans chances of default if these companies are not able to get fair price for their commodity. Indian companies having tie-up with foreign oil exploration companies will suffer if the price falls further as they won't have enough cash flow to run their business. Companies with business in petroleum products, polymers, chemicals are stuck up with high inventory due to fall in prices not able to sell it because they have brought when prices were high

Ideally this fall should have led to substantial fall in petrol, diesel prices but the govt has not passed the benefit to consumers. This may be due to the fact that govt. has to meet its fiscal target. But this looks a good option in short-term, but in the long run govt should pass on the benefit to consumers who can spend the surplus elsewhere which can add to growth of the nation.

Reduced FDI, FII inflow, reduced petroleum led exports, rupee depreciation, de-motivation for oil companies. As predicted by some global economic organizations,

continued persistence of the reduced price may lead to recession. The International impacts are reduced remittances as coming mainly from gulf countries, oil explorations abroad effected ex. OVL. Other factors are neglect of other environment friendly means, stock market fluctuations as can be seen from the recent trends causing public unrest.

CHAPTER 11

CONCLUSION

The refining of crude oil takes place all over the world. In the early days of the petroleum refining industry, production of desired gasoline and kerosene products. An artificial neural network model is presented with the task of determining the most favourable lag. In the crude oil price data. It is evident, that the prediction is accurate till there is a massive and sudden change in the actual data, where it becomes challenging to predict the exact new price with the change, however, the proposed model has efficiently taken into consideration these patterns.

The ANN model is an effective tool for crude oil price prediction and can be efficiently used for short term price forecasting by determining the optimal lags. The proposed model is powerful and highly suggested because investors can use it not only to initiate trades but also as an effective tool to judge various strategies relating to investments. This work is carried out on the closing price of crude oil. So, there are various other factors which also affect the crude oil prices like change in the prices and quantities (demand and supply), change in the economy and current affairs as shown by the media. The main advantage of this research is in capturing the changing pattern of these prices. The effectiveness and accurateness of data selection also helps to extensively deliberate the input variables combination for ANN-Q model. Data represented in One-step Returns function had successfully proved to cleanse and uniform the data from errors and noises hence, the crisp Prediction result will be obtained.

CHAPTER 12

FUTURE SCOPE

Recent years, the crude oil market has entered a new period of development and the core influence factors of crude oil have also been a change. Thus, we develop a new research framework for core influence factors selection and forecasting. Firstly, this paper assesses and selects core influence factors with the elastic-net regularized generalized linear Model (GLMNET), spike-slab lasso method, and Bayesian model average (BMA). Secondly, the new machine learning method long short-term Memory Network (LSTM) is developed for crude oil price forecasting. Then six different forecasting techniques, random walk (RW), autoregressive integrated moving average models (ARMA), Elman neural Networks (ENN), ELM Neural Networks (EL), wallet neural networks (WNN) and generalized regression neural network Models (GRNN) were used to forecast the price. Finally, we compare and analyse the different results with root mean squared error (RMSE), mean absolute percentage error (MAPE), directional symmetry.

The oil market has taken on new features that affect the development of the global economy, national strategic security, and investor sentiment significantly. Especially as the primary alternative energy resources, the US tight oil production has been significant macroeconomic effects on the oil price. In 2014, US shale oil producers plundered market share, leading to a change in global crude oil supply and demand balance. According to EIA, US shale oil production increased from 4.96 million barrels per day in 2017 to 5.59 million barrels per day in 2022.

Testing the performance of the proposed variable selection and machine learning framework based on variable selections and individual forecasts. In future research, we may introduce more independent variables with the help of internet search data, test our framework performance. Moreover, investor sentiment can be quantified in this process. In addition, different variable selection methods can be introduced more.

APPENDIX I

SOURCE CODE

```
import numpy as np
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers import Dropout
from keras.layers import Dense
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_absolute_error as mae
from sklearn.metrics import mean_squared_error as mse
from sklearn.metrics import r2_score as r2s
from google.colab import files
from math import sqrt

//initializing the model

def plotCurve(x,y,xlable,ylabel,clabel):
    fig, ax = plt.subplots(figsize=(5, 3))
    fig.subplots_adjust(bottom=0.15, left=0.2)
    ax.plot(x,y,label=clabel)
    ax.set_xlabel(xlable)
    ax.set_ylabel(ylabel)
    plt.grid()
    ax.legend()
    plt.show()
def plotTwoCurves(x1,x2,y1,y2,xlable,ylabel,clabel1,clabel2):
    fig, ax = plt.subplots(figsize=(5, 3))
    fig.subplots_adjust(bottom=0.15, left=0.2)
    ax.plot(x1,y1,color='blue',label=clabel1)
```

```
    ax.plot(x2,y2,color='red',label=clabel2)
    ax.set_xlabel(xlable)
    ax.set_ylabel(ylabel)
    plt.legend()
    plt.show()

ds=pd.read_csv('Crude_Oil_Prices.csv')
ds=ds.set_index(ds['Date'])
ds=ds.dropna()
print(ds)
ds['Date']=pd.to_datetime(ds['Date'])
print(ds['Value'].head())
index1=ds['Date']
plotCurve(index1,ds['Value'],'Time(Days)','Price','Price Series')
ds_price=ds['Value'].astype(float)
scaler=StandardScaler()
scaler=scaler.fit(ds_price.values.reshape(-1, 1))
ds_price_scaled=scaler.transform(ds_price.values.reshape(-1, 1))
ds_price_scaled
plotTwoCurves(index1,index1,ds['Value'],ds_price_scaled,'Time(Days)','Price','Price
Series','Scaled Price')
oilPX=[]
oilPY=[]
predicted_data=0
actual_data=0
next_period=1
window_size=14
for i in range(window_size, len(ds_price_scaled)-next_period+1):
    oilPX.append(ds_price_scaled[i-window_size:i])
    oilPY.append(ds_price_scaled[i+next_period-1:i+next_period,0])
oilPX,oilPY=np.array(oilPX),np.array(oilPY)
print('shape= {}.'.format(ds.shape))
print('Price Scaled shape= {}.'.format(ds_price_scaled.shape))
print('oilPX shape== {}.'.format(oilPX.shape))
```

```
print('oilPY shape== {}.'.format(oilPY.shape))

//adding lstm layers

model=Sequential()
model.add(LSTM(100, activation='relu', input_shape=(oilPX.shape[1], oilPX.shape[2]),
return_sequences=True))
model.add(LSTM(50, activation='relu', return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(oilPY.shape[1]))

model.compile(optimizer='adam',loss='mse')
model.summary()
history=model.fit(oilPX,oilPY,epochs=30,batch_size=16, validation_split=0.1,verbose=1)
index2=range(0,len(history.history['loss']))
plotTwoCurves(index2,index2,history.history['loss'],history.history['val_loss'],'Epochs','Loss',
'Loss Series','Value loss Series')

//adding output layers

forecast_date=80
predicted_data=model.predict(oilPX[-forecast_date:])
actual_data=ds['Value'][-forecast_date:]

predicted_data_NN=scaler.inverse_transform(predicted_data)
index3=range(0,len(predicted_data_NN))
plotTwoCurves(index3,index3,ds['Value'][-
forecast_date:],predicted_data_NN,'Time(Days)','Price','Actual Oil Price','Predicted Oil
Price')
index4=range(0,len(ds)-forecast_date)
index5=range(len(ds)-forecast_date,len(ds))
plotTwoCurves(index4,index5,ds['Value'][0:len(ds)-
forecast_date],predicted_data_NN,'Time(Days)','Price','Actual Oil Price','Predicted Oil Price')
```

```
//evaluating and testing data

mae1=mae(actual_data,predicted_data_NN)
mse1=mse(actual_data,predicted_data_NN)
print(mse1)
rmse1=sqrt(mse1)
r2_score=r2s(actual_data,predicted_data_NN)
print(r2_score)
n=forecast_date
k=1
adjacent_result=1-(((1-r2_score)*(n-1))/(n-k-1))
print(adjacent_result)
predict_index1=range(len(ds)-128,len(ds)-forecast_date)
predict_index2=range(len(ds)-forecast_date,len(ds))
plotTwoCurves(predict_index1,predict_index2,ds['Value'][len(ds)-128:len(ds)-
forecast_date],predicted_data_NN,'Time(Days)','Price','Actual Oil Price')

//accuracy prediction

accuracy=100-(adjacent_result+r2_score+mse1)
print("Accuracy = "'%.2f' %accuracy + " %")

//saving model

model.save('saved_model/my_model')
```

PREDICITION CODE:

```
import numpy as np

import os

from flask import Flask, render_template, request

from tensorflow.keras.models import load_model
```

```
from werkzeug.utils import secure_filename, redirect
from gevent.pywsgi import WSGIServer
from flask import send_from_directoryx
model = load_model("crude_oil.h5")
app = Flask (_name_,template_folder='templates')
@app.route('/')
def home():
    return render_template("index.html")
@app.route('/about')
def home1():
    return render_template("index.html")
@app.route('/predict')
def home2():
    return render_template("web.html")
@app.route('/predict', methods=['POST'] )
def login() :
     x_input=str(request.form['year'])
     x_input=x_input.split(',')
     print(x_input)
     for i in range(0, len(x_input)):
         x_input[i] = float(x_input[i])
     print(x_input)
     x_input=np.array(x_input).reshape(1,-1)
     temp_input=list(x_input)
     temp_input=temp_input[0].tolist()
     lst_output=[]
     n_steps=10
```

```
    i=0
    while(i<1):
      if(len(temp_input)>10):
          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))
          yhat = model.predict(x_input, verbose=0)
          print("{} day output {}".format(i,yhat))
          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)
          print(yhat[0])
          temp_input.extend(yhat[0].tolist())
          print(len(temp_input))
          lst_output.extend(yhat.tolist())
          i=i+1
      print(lst_output)
      return render_template("web.html", showcase = 'The next day predicted value
is:'+str(lst_output))


if _name_ == '_main_' :
   app.run(debug = True,port=5000)
```

GITHUB LINK: https://github.com/IBM-EPBL/IBM-Project-16593-1659618147

DEMO LINK: https://github.com/IBM-EPBL/IBM-Project-16593-1659618147blob/main/final%20deliverables/demo/DEMO_crude.mp4

REFERENCES

- K. Kanchymalay, R. Sallehuddin, N. Salim and U. R. Hashim, "Time series-based forecasting for crude palm oil price utilizing neural network algorithms," 2017 6th ICT International Student Project Conference (ICT-ISPC), 2017, pp. 1-4, doi: 10.1109/ICT-ISPC.2017.8075334.
- C. Zhang, F. Jiang, S. Wang and W. Shang, "A Novel Hybrid Approach with A Decomposition Method and The RVFL Model for Crude Oil Price Prediction," 2020 IEEE International Conference on Big Data (Big Data), 2020, pp. 4446-4449, doi: 10.1109/BigData50022.2020.9377826.
- M. M. L. Heravi, M. Khorrampanah and M. Houshmand, "Forecasting Crude Oil Prices Using Improved Deep Belief Network (IDBN) and Long-Term Short-Term Memory Network (LSTM)," 2022 30th International Conference on Electrical Engineering (ICEE), 2022, pp. 823-826, doi: 10.1109/ICEE55646.2022.9827452.