PSG COLLEGE OF TECHNOLOGY

TEAM ID: PNT2022TMID13112

CRUDE OIL PRICE PREDICTION

TEAM MEMBERS:

19I314-GOKULKANNAN R
19I356- SURIYA C
19I360- YADHU NANDHINI S E
19I362- SOWMYAA SHRI S

INDEX

1. INTRODUCTION
    1.1 Project Overview
    1.2 Purpose
2. LITERATURE SURVEY
    2.1 Existing problem
    2.2 References
    2.3 Problem Statement Definition
3. IDEATION & PROPOSED SOLUTION
    3.1 Empathy Map Canvas
    3.2 Ideation & Brainstorming
    3.3 Proposed Solution
    3.4 Problem Solution fit
4. REQUIREMENT ANALYSIS
    4.1 Functional requirement
    4.2 Non-Functional requirements
5. PROJECT DESIGN
    5.1 Data Flow Diagrams
    5.2 Solution & Technical Architecture
    5.3 User Stories
6. PROJECT PLANNING & SCHEDULING
    6.1 Sprint Planning & Estimation
    6.2 Sprint Delivery Schedule
    6.3 Reports from JIRA
7. CODING & SOLUTIONING
    7.1 Feature 1
    7.2 Feature 2
8. TESTING
    8.1 Test Cases
    8.2 User Acceptance Testing
9. RESULTS
    9.1 Performance Metrics
10. ADVANTAGES & DISADVANTAGES
11. CONCLUSION
12. FUTURE SCOPE
13. APPENDIX
    Source Code
    GitHub & Project Demo Link

1. INTRODUCTION

1.1 Project Overview

Oil is the largest traded commodity and its demand is highly inelastic. Both the oil importers and exporters have a major concern towards the increasing or decreasing price. The rise in price is good news for producers because they will see an increase in their revenue. Oil importers, however, will experience increased costs of purchasing oil. A rising oil price can even shift economic or political power from oil importers to oil exporters. The crude oil price movements are subject to diverse influencing factors. Therefore, prediction of crude oil prices are highly important.

The proposed work mainly focuses on applying Neural Networks to predict the Crude Oil Price. This decision helps importers to buy crude oil at the proper time. Time series analysis is the best option for this kind of prediction as it utilizes the Previous history of crude oil prices to predict future crude oil.

1.2 Purpose

Crude oil prices are controlled by many factors which include the difficulty for industrialists to make profit out of volatility, downfall in the market because of sudden fall in prices, and so on. This problem affects the industrialists, crude oil investors, supply and demand.It is important to fix the problem as it can stabilize the market. The boundaries of the problem are to increase the profit and to predict crude oil investments. Therefore, Prediction of crude oil can help in dealing with the sudden rise and fall of prices.

2. LITERATURE SURVEY

2.1 Existing problem

2.1.1 A new approach for crude oil price prediction based on stream learning

Shuang Gao et al., proposed a novel approach based on a new machine learning paradigm called steam learning to predict the price of crude oil. The main advantage of stream learning approach is that the prediction model can capture the changing pattern of oil prices since the model is continuously updated whenever new oil price data are available, with very small constant overhead. The stream learning approach has been compared with other models in order to test the accuracy of results and it is observed that this model yields a higher performance result in terms of mean squared error and directional ratio over a variety of horizons. Two types of prices are predicted which includes the U.S. refiner acquisition cost for crude oil imports and the WTI crude oil spot price. The stream learning model handles applications where continuous data streams are generated from non-stationary processes. Since the oil prices are non-stationary, a technique to handle continuous flow of data as opposed to a fixed sample of independent and identically distributed data is required. The oil prices of previous time slots are used to predict the oil prices of the current time slot. For non-stationary time series data such as oil prices, a forgetting mechanism (e.g., sliding windows, fading factors) will be deployed when updating the machine learning model. MOA (massive online analysis) has been used to develop stream learning models for oil price prediction. Training data is split into 90% as training set and 10% as the development set. The stream learning model achieves the lowest mean squared prediction error (MSPE) and highest directional accuracy ratio (DAR) among all the other prediction models

2.1.2 Forecasting the price of crude oil

Ramesh Bollapragada et al., developed a forecasting model to predict the oil prices that aid management to reduce operational costs, increase profit and enhance competitive advantage. This work implemented ARIMA models through MINITAB 19 software. The plot of the data of the oil prices from 1986 to 2017 showed an increasing trend. The autocorrelations were compared with their error limits, the only significant autocorrelation was at lag 1. Similarly, only the lag 2 partial autocorrelation was significant. Neither pattern appears to die out in a declining manner at low lags. So, it was decided to fit both ARIMA (2, 1, 0) and ARIMA (0, 1, 1) models to the data. Implemented the combination method also. After obtaining the values of the forecasted crude oil prices, the values are compared with the prices of the previous period. It was inferred that the Combination Method performed better than ARIMA (2, 1, 0) across all metrics, and is better than ARIMA (0, 1, 1) on the MSE metric. ARIMA (0, 1, 1) performs slightly better than the Combination Method on the MAD, MAPE and MPE metrics.

2.1.3 Crude oil price prediction using artificial neural network

Nalini Gupta et al., proposed a contemporary and innovative method of predicting crude oil prices using artificial neural networks (ANN). The main advantage 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. Back propagation algorithm and the error signal were cultivated through the network in the backward direction by changing and managing weights of the network to maximize the performance of the network. Root mean square error is used to evaluate the performance of the model. The results are taken out by varying the lag value, the optimal lag is obtained with the least RMSE values. The flaw with the model is that it is difficult to find the exact price of crude oil with exact change in price.

2.1.4 Analysis and forecasting of crude oil price based on the variable selection-LSTM integrated model

Quanying Lu et al., proposed 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. Six different forecasting techniques were used to assess the price. Finally, the different results with root mean squared error (RMSE), mean absolute percentage error (MAPE), directional symmetry (DS) were compared and analyzed. The empirical results show that the variable selection-LSTM method outperforms the benchmark methods in both level and directional forecasting accuracy.

2.1.5 Beta Clustering Of Impact Of Crude-Oil Prices On The Indian Economy

Sumit Ghosh et al., proposed the methodology of study on the impact of crude-oil prices on various economic parameters across the globe. The study was based on the Null hypothesis. Null Hypothesis states that the extent of impact of change in crude-oil prices on selected economic parameters were similar. Statistical techniques were also used. Regression equations were developed to study the impact of crude oil prices on the selected economic parameters. To understand whether the beta values of all regressions were equal, the mean beta was subjected to a t test along with calculating its confidence interval and other descriptive statistics. Beta clustering is a statistical technique of finding clusters of betas when variables are regressed against a common independent variable. Beta clustering is useful to understand the varying impact of the independent variable on several dependent variables. The results of the study not only show the significance of crude-oil for the Indian economy, but also provide pointers to policy makers for effective use of the beta clusters for better policy making. The paper has shown that the impact of crude-oil prices is not similar across various economic parameters.

2.2 References

I. Gao, S., & Lei, Y. (2017). A new approach for crude oil price prediction based on stream learning. Geoscience Frontiers, 8(1), 183-187.

II. Bollapragada, R., Mankude, A., & Udayabhanu, V. (2021). Forecasting the price of crude oil. Decision, 48(2), 207-231.

III. Gupta, N., & Nigam, S. (2020). Crude oil price prediction using artificial neural network. Procedia Computer Science, 170, 642-647.

IV. Lu, Q., Sun, S., Duan, H., & Wang, S. (2021). Analysis and forecasting of crude oil price based on the variable selection-LSTM integrated model. Energy Informatics, 4(2), 1-20.

V. Ghosh, S., & Sivakumar, N. (2015). Beta clustering of impact of crude-oil prices on the Indian economy. J. Appl. Management Investments, 4(1), 24-34.

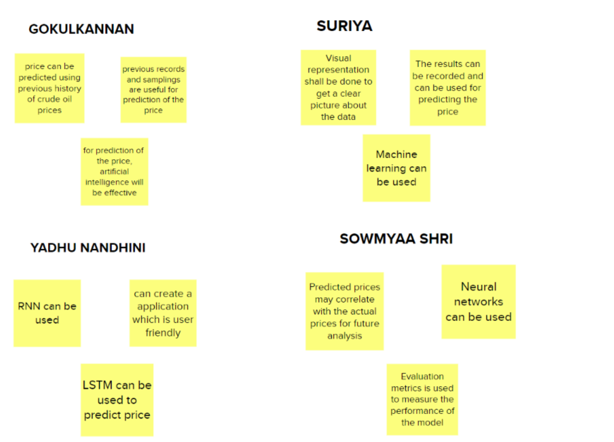
2.3 Problem Statement Definition

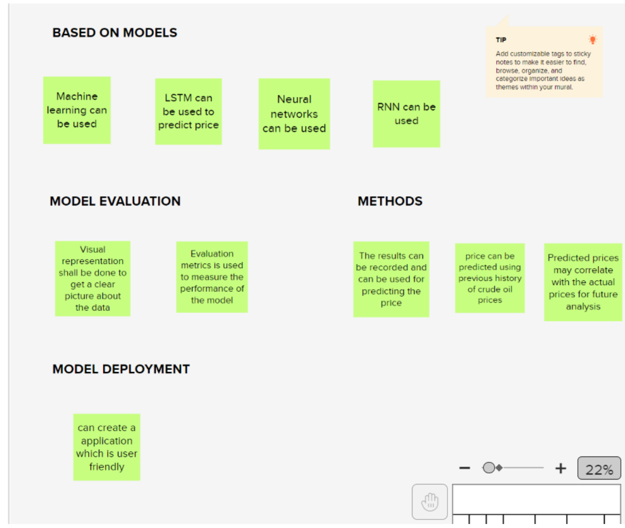
Crude oil price prediction is a challenging task in oil producing countries. Its price is among the most complex and tough to model because fluctuations of price of crude oil are highly irregular, nonlinear and vary dynamically with high uncertainty. The problem statement is to propose a hybrid model for crude oil price prediction that uses the RNN(Recurrent Neural Network) with long short-term memory (LSTM) to predict the crude oil price.

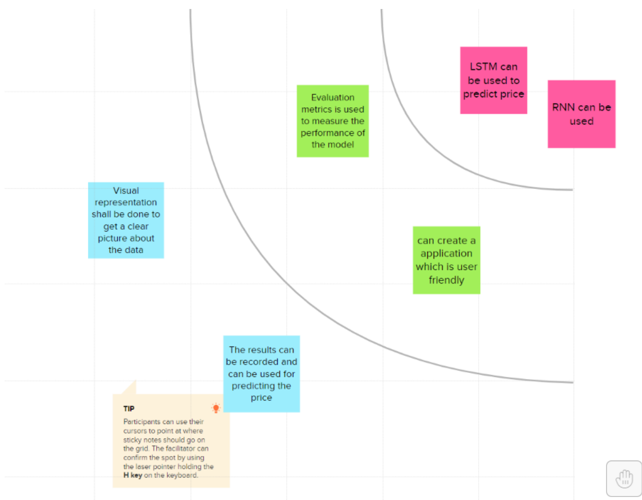
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming







3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement	Crude oil demand is inelastic, therefore the rise in price will increase the revenue of Oil investors. Oil importers will have a high purchasing cost on the rise of crude oil prices. Oil is the largest traded commodity, the effects are quite significant. A rising oil price can even shift economic/political power from oil importers to oil exporters. Therefore, an accurate crude oil price prediction mechanism is needed to allow researchers and stakeholders to understand and predict future

		crude oil prices
2.	Idea	- For prediction problems, traditional neural network models like Recurrent Neural Networks (RNN) can be used. - But RNN suffers from the vanishing gradient problem - Advanced versions of RNN like Long short term memory (LSTM) with a small window to reference from can be used - Attention mechanism can be incorporated with the neural network model to solve the vanishing gradient problem - It has an infinite window to reference from, It has the capacity to map a word with its related word to increase the remembrance
3.	Novelty	- Long Short term memory (LSTM) is used - LSTM works well on the time series data - LSTMs solve the problem of vanishing gradient using a unique additive gradient structure that includes direct access to the forget gate's activations, enabling the network to encourage desired behavior from the error gradient using frequent gates update on every time step of the learning process - Two layers are used for optimizing the structure of the neural network
4.	Social Impact	- It is majorly used in the supply and demand - The oil importers has to meet the demand of oil exporters, so it is important for the oil importers to understand the price fluctuations - Oil exporters must also be ready to meet the downfall of prices before making investments
5.	Business Model	The business models are retailers, manufacturers.

6.	Scalability of the Solution	• Economic growth is the important factor. • The accuracy of the model can be improved.

3.4 Problem Solution fit

Project Title: Crude Oil Price Prediction **Project Design Phase-I - Solution Fit Template** **Team ID:** PNT2022TMID13112

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

This project is to predict the future price of the crude oil and find the right time to purchase the crude oil.

6. CUSTOMER CONSTRAINTS CC

The customers need to enter the correct values to get better results.

5. AVAILABLE SOLUTIONS AS

Buying the oversold oil.

Explore AS, differentiate

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

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

Application should be user friendly.

Website should accessible at anytime.

9. PROBLEM ROOT CAUSE RC

Varying pattern of crude oil price

7. BEHAVIOUR BE

The information provided should be true.

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

3. TRIGGERS

TR
Saving money

4. EMOTIONS: BEFORE / AFTER

EM
Fear, Insecurity, Surprise, Happiness

10. YOUR SOLUTION

SL
This project mainly focuses on applying Neural Networks to predict the Crude Oil Price. This decision helps us to buy crude oil at the proper time. Time series analysis is the best option for this kind of prediction because we are using the Previous history of crude oil prices to predict future crude oil. So we would be implementing RNN(Recurrent Neural Network) with LSTM(Long Short Term Memory) to achieve the task.

8. CHANNELS of BEHAVIOUR

CH

8.1 **ONLINE**
Watching live news, watching youtube

8.2 **OFFLINE**
Analyzing the previous records
Risk management

4. REQUIREMENT ANALYSIS

4.1 Functional requirements

FR NO.	Functional Requirement	Sub Requirement
FR-1	User Registration	• Registration through Gmail • Registration through phone number
FR-2	User Confirmation	• Confirmation via Gmail • Confirmation via OTP
FR-3	Login	User can login with the registered email and phone number

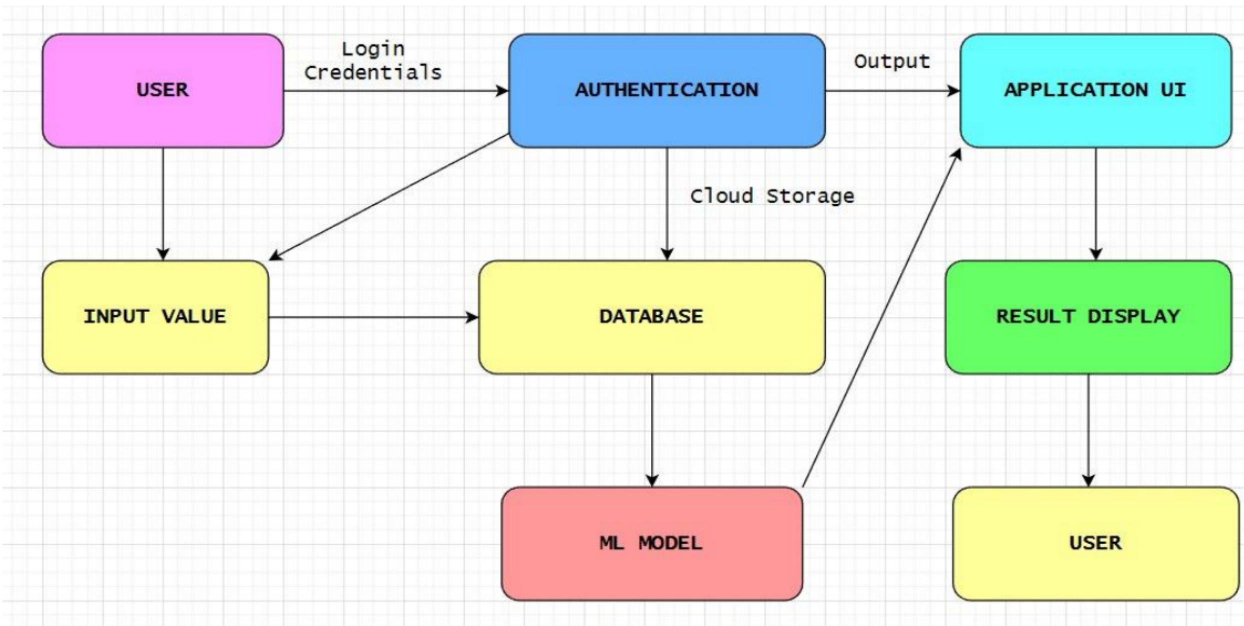
4.2 Non-Functional requirements

FR NO.	Non Functional Requirement	Description
NFR-1	Usability	UI is user friendly.
NFR-2	Security	User protocols like OTP verification and user credentials are used
NFR-3	Reliability	The predicted results are accurate
NFR-4	Performance	Accuracy of the predicted data

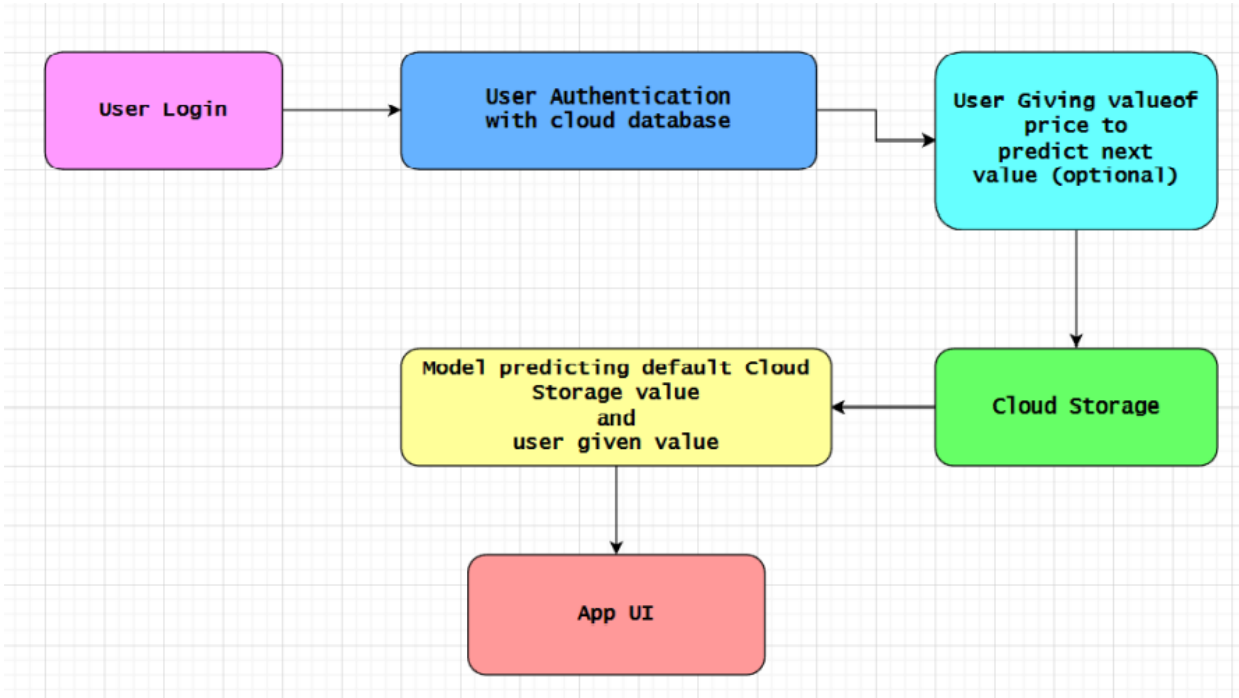
NFR-5	Availability	Available in all the devices

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture



Components & Technologies:

S.No	Component	Description	Technology
1	User Interface	How user interacts with application	HTML, CSS, React Js
2	Application Logic-1	Logic for a process in the application	Python (Flask)
3	Database	Data Type, Configurations etc	MySQL

4	Cloud Database	Database Service on Cloud	IBM Cloud
5	File Storage	File storage requirements	IBM Block Storage
6	External API-1	For standalone server	Firebase
7	Machine Learning Model	To predict the price of crude oil	Recurrent neural network & LSTM
8	Infrastructure (Server / Cloud)	Application Deployment on Local Server and Firebase	Local , Firebase

Application Characteristics:

S.No	Characteristics	Description	Technology
1	Open-Source Framework-1	Python	Pandas,flask,numpy,tensorflow
2	Open-Source Framework-2	ReactJs	App module
3	Open-Source Framework-3	HTML & CSS	<div>module
4	Scalable Architecture	IBM cloud and firebase	IBM Watson, Firebase, Mysql

5	Availability	Handle huge requests	Effective coding
6	Performance	Handles 100 to 10000 users to use server at a time	Flask

5.3 User Stories

User Type	Functional Requirement (epic)	User Story Number	User Story/ Task	Acceptance Criteria	Priority	Release
Customer (Website 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 database	High	Sprint-1
Customer (cloud user)	Access	USN-2	As a user, I can access the model database	Getting confirmation email	Medium	Sprint-2
Administrator	Login	USN-3	As an admin, I can log into application by entering	I can access the model directly	High	Sprint-1

			email and password			
Customer (User)	Gadgets (computer/ mobile/ laptop)	USN-4	As a user I can view the pictorial representation of crude oil price	I can insight the crude oil price	High	Sprint-4
Customer (User)	Internet Facility	USN-5	As a user I can give input to the model through the website	I can get the crude oil price	High	Sprint-3

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	As a user, I want to collect the dataset so that I can perform	3	High	Gokul Kannan R

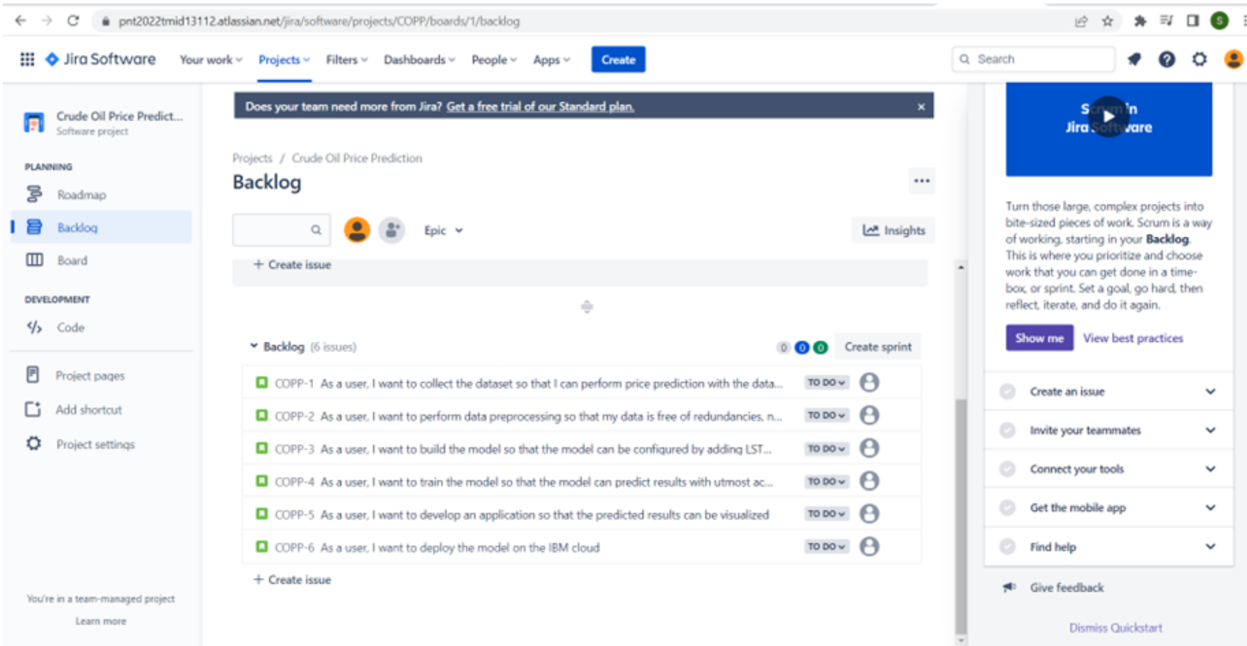
			price prediction with the data available			
Sprint-1	Data preprocessing	USN-2	As a user, I want to perform data preprocessing so that my data is free of redundancies, noisy value, missing values and so on	5	Medium	Sowmyaa Shri S
Sprint-2	Model Building	USN-3	As a user, I want to build the model so that the model can be configured by adding LSTM and output layers	5	Medium	Suriya C
Sprint-2	Model Training	USN-4	As a user, I want to train the model so that the model can predict results with utmost accuracy	3	Medium	Yadhu Nandhini S E
Sprint-3	Application development	USN-5	As a user, I want to develop an application so that the predicted results can be visualized	8	High	Gokul Kannan R Sowmyaa Shri S
Sprint-4	Model Deployment	USN-6	As a user, I want to deploy the model on the IBM cloud	8	High	Yadhu Nandhini S E

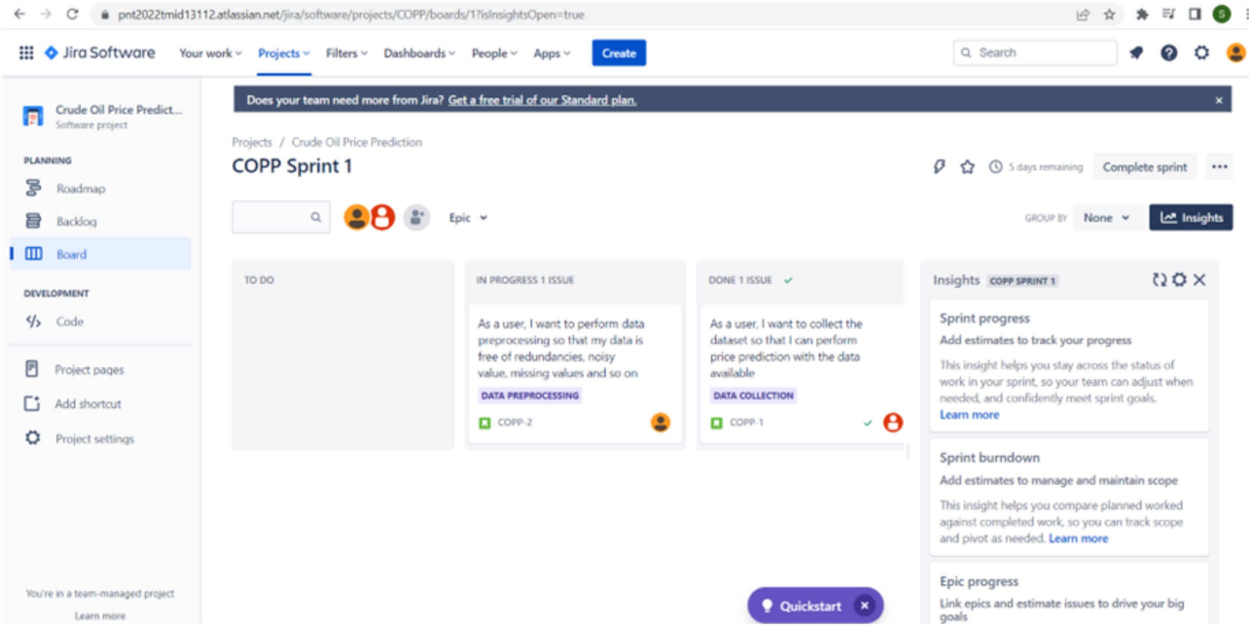
						Suriya C

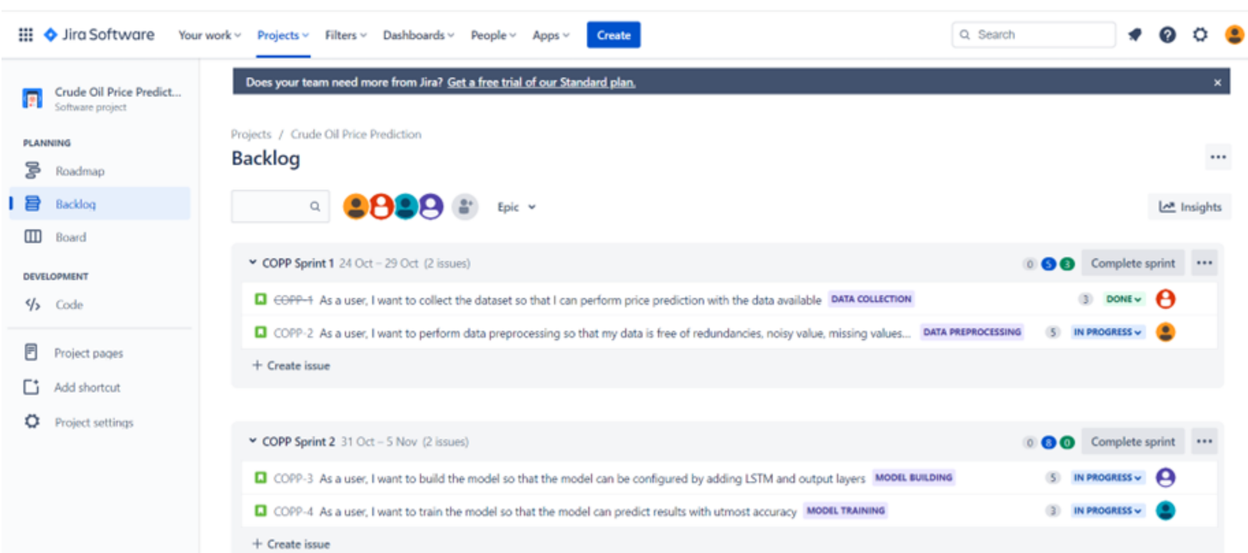
6.2 Sprint Delivery Schedule

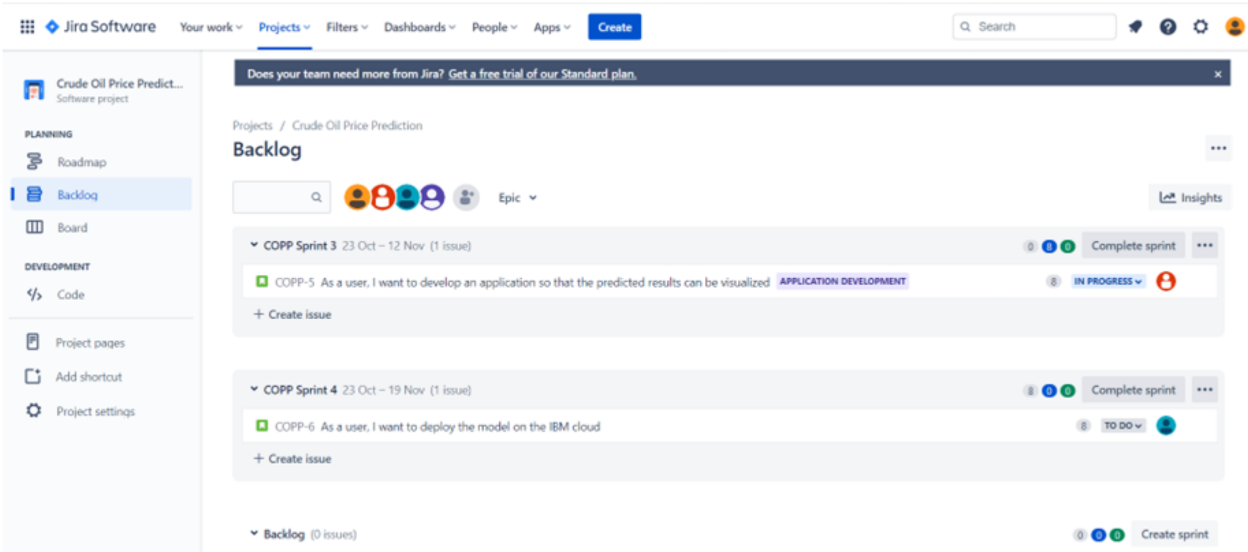
Sprint	**Total Story Points**	**Durat -ion**	**Sprint Start Date**	**Sprint End Date (Planned)**	**Story Points Completed (as on Planned End Date)**	**Sprint Release Date (Actual)**
Sprint-1	8	6 Days	24 Oct 2022	29 Oct 2022	8	29 Oct 2022
Sprint-2	8	6 Days	31 Oct 2022	05 Nov 2022	8	05 Nov 2022
Sprint-3	8	6 Days	07 Nov 2022	12 Nov 2022	8	12 Nov 2022
Sprint-4	8	6 Days	14 Nov 2022	19 Nov 2022	8	19 Nov 2022

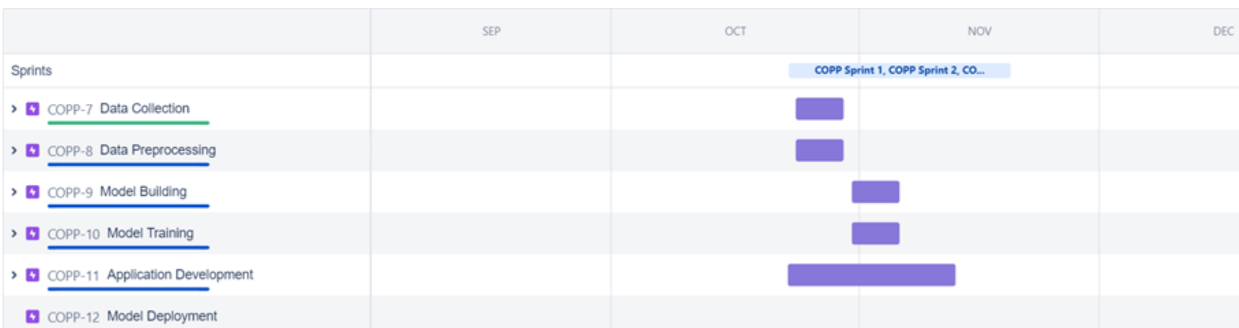
6.3 Reports from JIRA




pnt2022tmid13112.atlassian.net/jira/software/projects/COPP/boards/1?isInsightsOpen=true
Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create
Search
Does your team need more from Jira? Get a free trial of our Standard plan.
Crude Oil Price Predict...
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Crude Oil Price Prediction
COPP Sprint 1
5 days remaining
Complete sprint
Epic
GROUP BY
None
Insights
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
Quickstart


Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create
Search
Does your team need more from Jira? Get a free trial of our Standard plan.
Crude Oil Price Predict...
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
Projects / Crude Oil Price Prediction
Backlog
Epic
Insights
COPP Sprint 1 24 Oct – 29 Oct (2 issues)
Complete sprint
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
+ Create issue
COPP Sprint 2 31 Oct – 5 Nov (2 issues)
Complete sprint
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
+ Create issue





7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature

- The crude oil prediction website provides two options
    - Home
    - predict

- The home allows the user to have an insight on the importance of crude oil price prediction
- The predict allows the user to give the 10 days input and arrive at the prediction results

The above image is the home page.

The above image the predict page.

The input is given and predict button is clicked.

The output is displayed.

Code:

Index.html:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Crudeoil price prediction</title>
    <!--<link rel="stylesheet" href="in.css">
```

```
        <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&d
isplay=swap"rel="stylesheet">
        <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.c
ss">-->
      <style>
         ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  border: 1px solid #e7e7e7;
  background-color: #057514;
}

li {
  float: left;
}

li a {
  display: inline-block;
  color: rgb(78, 3, 3);
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  background-color:rgb(18, 116, 5) ;
}
li a:hover{
    border:1px solid;
    background-color: lightseagreen;
}
      </style>
    </head>
    <body>
        <nav class="navbar navbar-inverse">
            <div class="container-fluid">
                <ul>
                  <li class="parts"><a href="#">Home</a></li>
```

```
            <li class="parts"><a href="predict.html">predict</a></li>
          </ul>
        </div>
      </nav>

    <h1 >Crudeoil price prediction</h1>

    <style>
       body {
         background-image: url('static/css/image.jpeg');
         background-repeat: no-repeat;
         background-attachment: fixed;
       background-size: 100% 100%;

       }
       </style>

      <h3 style="font-family:system-ui;">
         Demand for oil is inelastic, therefore the rise in
         price is good news for producers because they will see an
         increase in their revenue. Oil importers, however, will experience
         increased costs of purchasing oil. Because oil is the largest traded commodity,
         the effects are quite significant. A rising oil price can even shift
economic/political
       power from oil importers to oil exporters. The crude oil rice movements are
subject to
       diverse influencing factors</h3>
    </body>
    </html>
```

Predict.html:

```
<html>
  <head>
    <link rel="stylesheet" href="static/css/style.css">
    <style>
       body {
         background-image: url('static/css/image3.jpg');
         background-repeat: no-repeat;
```

```
          background-attachment: fixed;
         background-size: 100% 100%;

         }
         </style>

   </head>
   <script>
      document.getElementByID("demo").innerHTML =
document.getElementById("ten");
   </script>
<body>
<form action="/method" method="POST" enctype = "multipart/form-data">
<div class="container">
   <!--<div class="brand-logo"></div>-->
   <div class="brand-title">predict the oil price</div>
   <div class="inputs">
     <label>Enter Price</label>
     <input type="text" placeholder="Enter ten days price" id="ten" name="val"/>
     <button type="submit">Predict</button><br><br>
      The next day price is :  {{prediction}}
   </div>
  </div>
</form>
</body>
</html>
```

App.py:

```
from flask import Flask, render_template, request, redirect
import numpy as np
# from tensorflow.k
from keras.saving.save import load_model
app = Flask(__name__,template_folder='template')

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

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

```python
@app.route('/method', methods=["POST", "GET"])
def method():
    if request.method == "POST":
        string = request.form['val']
        string = string.split(',')
        temp_input = [eval(i) for i in string]

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

        lst_output = []
        n_steps = 10
        i = 0
        while (i < 10):
            if (len(temp_input) > 10):
                x_input = np.array(temp_input[1:])
                x_input = x_input.reshape(1, -1)
                x_input = x_input.reshape((1, n_steps, 1))
                yhat = model.predict(x_input, verbose=0)
                temp_input.extend(yhat[0].tolist())
                temp_input = temp_input[1:]
                lst_output.extend(yhat.tolist())
                i = i + 1

            else:
                x_input = x_input.reshape((1, n_steps, 1))
                yhat = model.predict(x_input, verbose=0)
                temp_input.extend(yhat[0].tolist())
                lst_output.extend(yhat.tolist())
                i = i + 1
        val = lst_output[9]
        return render_template('predict.html', prediction=val)
    if request.method == "GET":
        return render_template('predict.html')

if __name__ == "__main__":
    model = load_model(r'crudeoilprediction.h5')
    app.run(debug=True)
```

8. TESTING

8.1 Test Cases

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

8.2 User Acceptance Testing

The purpose is to briefly explain the test coverage and open issues of the crude oil price prediction project at the time of the release to user acceptance testing

Defect Analysis:

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

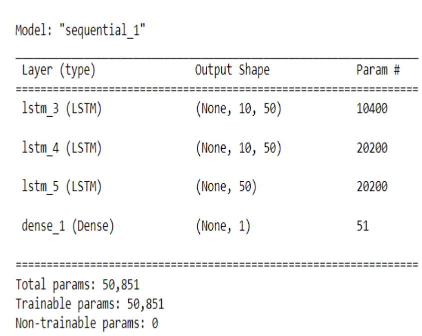
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't fix	0	0	0	1	1
Totals	8	0	2	2	12

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

9. RESULTS

9.1 Performance Metrics

S.No	Parameters	Values	Screenshot
1.	Model Summary		

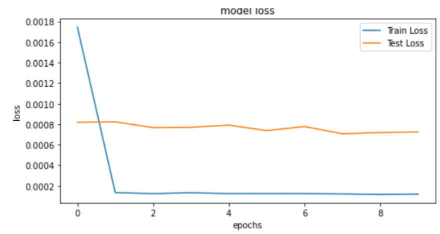
```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_3 (LSTM)               (None, 10, 50)            10400

 lstm_4 (LSTM)               (None, 10, 50)            20200

 lstm_5 (LSTM)               (None, 50)                20200

 dense_1 (Dense)             (None, 1)                 51

=================================================================
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: 1.4285248639934083 Test Mean Absolute Error: 2.780526920817909 Test Root Mean Squared Error: 3.6348234466523737 

10. ADVANTAGES & DISADVANTAGES

Advantages:

- 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

Disadvantages

- The prediction results may lack accuracy
- Volatility in prices may be misleading

11. CONCLUSION

LSTM network is better than other traditional neural networks for forecasting prices as it aims in using a back propagation model. Traditional neural networks such as CNN on the other hand predicts the next outgoing but doesn't necessarily save the previous data or connection which is based on feed-forwarding, in the sense the previous data is not necessary to predict the future data. LSTM focuses on storing the previous data and prediction which is rather encouraging and more approximate. The outcomes derived are relatively encouraging. The results show that large lookups do not necessarily improve the accuracy of the predictions of crude oil prices. Hence it can be concluded, the model with a single LSTM model is definitely the most accurate.

12. FUTURE SCOPE

The project's future potential is enormous. The project can be implemented with the real-time functionalities that are necessary.Because it is quite versatile in terms of expansion, the project can be upgraded in the near future as and when the need arises. The complete prediction value can be increased in a much better, accurate, and error-free manner with the proposed approach. The project can be enhanced with real time data.

APPENDIX

Source Code

MODEL:

```
# DATA PREPROCESSING

## Importing the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
data=pd.read_excel(r"Crude Oil Prices Daily.xlsx")
data.head()
## Handling missing values
data.isnull().any()
data.isnull().sum()
data.dropna(axis=0,inplace=True)
data_oil=data.reset_index()['Closing Value']
data_oil
data.isnull().any()
## Feature Scaling
from sklearn.preprocessing import MinMaxScaler
scalar=MinMaxScaler(feature_range=(0,1))
data_oil=scalar.fit_transform(np.array(data_oil).reshape(-1,1))

## Data Visualization
plt.title('Crude oil price')
plt.plot(data_oil)
## Splitting data into Train and Test Data
```

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

Creating a dataset with sliding windows

```
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)
time_step = 10

X_train, y_train=create_dataset(train_data,time_step)

X_test, y_test = create_dataset(test_data,time_step)
print(X_train.shape),print(y_train.shape)
print(X_test.shape),print(y_test.shape)
X_train
X_train.shape
X_train=X_train.reshape(X_train.shape[0],X_train.shape[1],1)
X_test=X_test.reshape(X_test.shape[0],X_test.shape[1],1)
```

 # MODEL BUILDING

 # Importing the model building libraries

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
# Initializing the model
model=Sequential()
```

 # Adding LSTM Layers

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

 # Adding Output Layers

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

model.summary()
 # Configure The Learning Process
model.compile(loss='mean_squared_error',optimizer='adam')
 # Train The Model
model.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=10,batch_size=64,verb
ose=1)
 # Model Evaluation
train_predict=model.predict(X_train)
test_predict=model.predict(X_test)
train_predict=scalar.inverse_transform(train_predict)
test_predict=scalar.inverse_transform(test_predict)
import math
from sklearn.metrics import mean_squared_error

math.sqrt(mean_squared_error(y_train,train_predict))
 # Save The Model
from tensorflow.keras.models import load_model
model.save("crudeoilprediction.h5")
 # Test The Model
look_back= 10
trainPredictPlot = np.empty_like(data_oil)
trainPredictPlot[:, :]= np.nan
trainPredictPlot[look_back:len(train_predict)+look_back, :]= train_predict
testPredictPlot =np.empty_like(data_oil)
testPredictPlot[:, :]= np.nan
testPredictPlot[len(train_predict)+(look_back*2)+1:len(data_oil)-1, :]= test_predict
plt.plot(scalar.inverse_transform(data_oil))
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()
len(test_data)
x_input=test_data[2866:].reshape(1,-1)
x_input.shape
temp_input=list(x_input)
temp_input=temp_input[0].tolist()
temp_input
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

print (lst_output)
day_new=np.arange(1,11)
day_pred=np.arange(11,21)
len(data_oil)
plt.plot(day_new,scalar.inverse_transform(data_oil[8206:]))
plt.title("Review of prediction")
plt.plot(day_pred,scalar.inverse_transform(lst_output))
plt.show()
df3=data_oil.tolist()
df3.extend(lst_output)
plt.title("Past data nad next 10 days output prediction")
plt.plot(df3[8100:])
df3=scalar.inverse_transform(df3).tolist()
plt.title("Past data nad next 10 days output prediction after reversing the scaled values")
```

plt.plot(df3)

Index.html:

```
<!DOCTYPE html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>Crudeoil price prediction</title>
      <!--<link rel="stylesheet" href="in.css">
      <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&d
isplay=swap"rel="stylesheet">
      <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.c
ss">-->
     <style>
       ul {
 list-style-type: none;
 margin: 0;
 padding: 0;
 overflow: hidden;
 border: 1px solid #e7e7e7;
 background-color: #057514;
}

li {
 float: left;
}

li a {
 display: inline-block;
 color: rgb(78, 3, 3);
 text-align: center;
 padding: 14px 16px;
 text-decoration: none;
 background-color:rgb(18, 116, 5) ;
}
li a:hover{
```

```
    border:1px solid;
    background-color: lightseagreen;
}
      </style>
    </head>
    <body>
        <nav class="navbar navbar-inverse">
            <div class="container-fluid">
                <ul>
                  <li class="parts"><a href="#">Home</a></li>
                  <li class="parts"><a href="predict.html">predict</a></li>
              </ul>
            </div>
          </nav>

        <h1 >Crudeoil price prediction</h1>

        <style>
            body {
              background-image: url('static/css/image.jpeg');
              background-repeat: no-repeat;
              background-attachment: fixed;
            background-size: 100% 100%;

            }
            </style>

          <h3 style="font-family:system-ui;">
             Demand for oil is inelastic, therefore the rise in
             price is good news for producers because they will see an
             increase in their revenue. Oil importers, however, will experience
             increased costs of purchasing oil. Because oil is the largest traded commodity,
             the effects are quite significant. A rising oil price can even shift
economic/political
            power from oil importers to oil exporters. The crude oil rice movements are
subject to
            diverse influencing factors</h3>
        </body>
        </html>
```

Predict.html:

```
<html>
   <head>
      <link rel="stylesheet" href="static/css/style.css">
      <style>
         body {
           background-image: url('static/css/image3.jpg');
           background-repeat: no-repeat;
           background-attachment: fixed;
         background-size: 100% 100%;

         }
         </style>

   </head>
   <script>
      document.getElementByID("demo").innerHTML =
document.getElementById("ten");
   </script>
<body>
<form action="/method" method="POST" enctype = "multipart/form-data">
<div class="container">
   <!--<div class="brand-logo"></div>-->
   <div class="brand-title">predict the oil price</div>
   <div class="inputs">
     <label>Enter Price</label>
     <input type="text" placeholder="Enter ten days price" id="ten" name="val"/>
     <button type="submit">Predict</button><br><br>
      The next day price is :  {{prediction}}
   </div>
  </div>
</form>
</body>
</html>
```

App.py:

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

```
from keras.saving.save import load_model
app = Flask(__name__,template_folder='template')

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

@app.route('/predict.html', methods=["POST", "GET"])
@app.route('/method', methods=["POST", "GET"])
def method():
    if request.method == "POST":
        string = request.form['val']
        string = string.split(',')
        temp_input = [eval(i) for i in string]

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

        lst_output = []
        n_steps = 10
        i = 0
        while (i < 10):
            if (len(temp_input) > 10):
                x_input = np.array(temp_input[1:])
                x_input = x_input.reshape(1, -1)
                x_input = x_input.reshape((1, n_steps, 1))
                yhat = model.predict(x_input, verbose=0)
                temp_input.extend(yhat[0].tolist())
                temp_input = temp_input[1:]
                lst_output.extend(yhat.tolist())
                i = i + 1

            else:
                x_input = x_input.reshape((1, n_steps, 1))
                yhat = model.predict(x_input, verbose=0)
                temp_input.extend(yhat[0].tolist())
                lst_output.extend(yhat.tolist())
                i = i + 1
        val = lst_output[9]
        return render_template('predict.html', prediction=val)
```

```
    if request.method == "GET":
        return render_template('predict.html')

if __name__ == "__main__":
    model = load_model(r'crudeoilprediction.h5')
    app.run(debug=True)
```

#cloud deployment code in ml model

```
!pip install ibm_watson_machine_learning

from ibm_watson_machine_learning import APIClient
wml_credentials = {
          "url" : "https://us-south.ml.cloud.ibm.com",
          "apikey" : "cRkqykhsnLO1Ogs_xoYjgLkNTtTS1Qxyi0Mn1GSlQ1P5"
}
client = APIClient(wml_credentials)

#for creating a deployment phase
def guid_from_space_name(client, space_name):
    space=client.spaces.get_details()
    #print(space)
return(next(item for item in space['resources'] if item['entity']['name'] ==
space_name)['metadata']['id'])
space_uid = guid_from_space_name(client, 'models')
print("Space UID = "+space_uid)
client.set.default_space(space_uid)
client.software_specifications.list()
software_spec_uid=
client.software_specifications.get_uid_by_name("tensorflow_rt22.1-py3.9")
software_spec_uid
```

GitHub & Project Demo Link

GITHUB: https://github.com/IBM-EPBL/IBM-Project-8197-1664354492

PROJECT DEMO LINK:
https://drive.google.com/file/d/1GfkcmeU1RDwZZwji28tnJJI0LAEeJEjN/view?usp=sharing