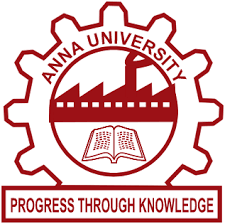
Icon

Description automatically generated

**CRUDE OIL PRICE PREDICTION**

TEAM ID : PNT2022MID08704

Submitted by

|  |  |  |
| --- | --- | --- |
| **Role** | **Name** | **Roll Number** |
| Team Leader | Michkel Anglo J | 727619BEC029 |
| Team Member | Krisha S | 727619BEC021 |
| Team Member | Surendrakumar B | 727619BEC051 |
| Team Member | Priyank Siddarth M J | 727620BEC303 |

**In partial fulfillment for the award of the degree of**

**BACHELOR OF ENGINEERING**

**in**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**Dr . MAHALINGAM COLLEGE OF ENGINEERING AND TECHNOLOGY An Autonomous Institution Affiliated to ANNAUNIVERSITY CHENNAI – 600 025**

**ABSTRACT**

In these 2022 transportation plays major role in human’s life especially Road transportation. For that petrol and Diesel are indispensable thing which was produced from crude oil. So many industries and Government yield lot from this crude oil businesses.By this price of crude oil become root source to decide many curve points in industry as well as country economy. But prediction of future crude oil price is considered a significant challenge due to the extremely complex, chaotic, and dynamic nature of the market and stakeholder's perception. The crude oil price changes every minute, and millions of shares ownership's are traded every day. The market price for commodity such as crude oil is influenced by many factors including news, supply-and-demand gap, labour costs, number of remaining resources, as well as stakeholders' perception. Therefore, various indicators for technical analysis have been utilized for the purpose of predicting the future crude oil price. Recently, many researchers have turned to Artificial Intelligence approach to cater to this problem. This study demonstrated the use of RNN with LSTM layered network for predicting the crude oil price based on historical data alongside other technical analysis indicators. This study aims to certify the capability of a prediction model built based on the RNN with LSTM layered network to predict the future price of crude oil. The developed model is trained and evaluated against accuracy matrices to assess the capability of the network to provide an improvement of the accuracy of crude oil price prediction as compared to other strategies. The result obtained from the model shows a promising prediction capability algorithm for predicting crude oil price movement.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER**  **NO.** | **TITLE** | **PAGE**  **NO.** |
| **1** | **INTRODUCTION**  1.1 PROJECT OVERVIEW  1.2 PURPOSE | **1 - 3**  2  3 |
| **2** | **LITERATURE SURVEY**  2.1 EXISTING PROBLEM  2.2 REFERENCES  2.3 PROBLEM STATEMENT DEFINITION | **4 - 6**  4  5  6 |
| **3** | **IDEATION & PROPOSED SOLUTION**  3.1 EMPATHY MAP CANVAS  3.2 IDEATION & BRAINSTORMING  3.3 PROPOSED SOLUTION  3.4 PROBLEM SOLUTION FIT | **7 - 10**  7  8  9  10 |
| **4** | **REQUIREMENT ANALYSIS**  4.1 FUNCTIONAL REQUIREMENT  4.2 NON-FUNCTIONAL REQUIREMENTS | **11 - 12**  11  12 |
| **5** | **PROJECT DESIGN**  5.1 DATA FLOW DIAGRAMS  5.2 USER STORIES  5.3 SOLUTION & TECHNICAL ARCHITECTURE | **13 - 15**  13  14  15 |
| **6** | **PROJECT PLANNING & SCHEDULING**  6.1 SPRINT PLANNING & ESTIMATION  6.2 SPRINT DELIVERY SCHEDULE  6.3 REPORTS FROM JIRA | **18 - 20**  18  19  20 |
| **7** | **CODING AND SOLUTIONING**  7.1 MODEL BUILDING  7.2 FLASK | **21 - 23**  21  23 |
| **8** | **TESTING**  8.1 TEST CASES  8.2 USER ACCEPTANCE TESTING | **27 - 31**  27  31 |
| **9** | **RESULTS**  9.1 PERFOMANCE METRICS | **35**  35 |
| **10** | **ADVANTAGES & DISADVANTAGES** | **37** |
| **11** | **CONCLUSION** | **38** |
| **12** | **FUTURE SCOPE** | **39** |
| **13** | **APPENDIX**  SOURCE CODE  GITHUB & PROJECT DEMO LINK | **40 - 53**  40  53 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Fig.NO** | **NAME** | **PAGE NO** |
| 1.1 | Emapthy map canvas | 7 |
| 1.2 | Brainstroming & Ideation | 8 |
| 1.3 | Problem solution fit canvas | 10 |
| 1.4 | Data flow map | 13 |
| 1.5 | Solution map | 15 |
| 1.6 | Technical flow map | 16 |
| 1.7 | Jira milestone | 20 |
| 1.8 | Metrics | 35 |
| 1.9 | Tuning | 36 |

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| LSTM | Long Short Term Memory |
| RNN | Recurrent Neural Network |
| MAE | Mean Absolute Error |
| MSE | Mean Squared Error |
| UAT | User Acceptance Test |
| UI | User Interface |
| HTML | HyperText Markup Language |
| CSS | Cascade Style Sheet |
| DDOS | Distributed Denial Of Service |
| XSS | Cross Side Scripting |
| GDP | Gross Domestic Product |
| OPEC | Organization of the Petroleum Exporting Countries |
| ML | Machine Learning |
| AI | Artificial Intelligence |

**CHAPTER 1**

**INTRODUCTION**

Prediction of crude oil prices has been a wide topic for ages. People use their intuition and lot of techniques to guess the prices of crude oil. It takes a lot of knowledge about the crude oil to accurately predict it. Predicting the crude oil price is very significant in various economic, political and industrial areas, both for crude oil importer and exporter countries. Since the crude oil is important strategic resource around the globe; it has become the crucial commodity for the world’s economy. Thus, prediction of prices of crude oil has always been considered as a very exciting and challenging task which drew the curiosity of professionals, researchers and organizations all over the world. So far, it remains the world’s leading fuel, with nearly one-third of global energy consumption. Crude oil price prediction has a scope larger than we can think of, the forecasting used is relevant for big and small industries along with the government benefiting from the predicted prices, but due to the evaporate nature of oil, it becomes very challenging to achieve accuracy. Moreover, crude oil volatility has a critical impact on macroeconomic parameters such as such as inflation, unemployment, exchange rate, economic growth of countries whose economy rely heavily on crude oil export or import. Thus, crude oil price prediction can help governments of countries of the world in economic policy making and make quick and operative economic decisions to hedge against probable risk in these economic parameters.

**1.1 PROJECT OVERVIEW**

In this, we have used LSTM based recurrent neural networks for the purpose of crude oil price prediction. Recurrent neural networks (RNN) have been proved to be one of the most powerful models for processing time-series based sequential data. LSTM is one of the most successful RNN architectures. LSTM introduces the memory cell, a unit of computation that replaces traditional artificial neurons in the hidden layer of the network. With these memory cells, networks are able to effectively associate memories and input remote in time, hence suit to grasp the structure of data dynamically over time with high prediction capacity.

Recurrent neural network is a type of Neural Network where the output from previous step is fed as input to the current step. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus, RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is Hidden state, which remembers some information about a sequence.

LSTM’s have chain like structure with the repeating module having a different structure. There are four neural network layers which are interacting to each other in a special way. The key to LSTM’s is the cell state, which is the horizontal line running through the top of the diagram. The cell state runs straight down the entire chain, with only some minor linear interactions. The information flows along it unchanged. The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through. They are composed out of a sigmoid neural network layer and a point-wise multiplication operation. An LSTM has three of these gates, to protect and control the cell state.

**1.2 PURPOSE**

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

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 EXISTING PROBLEM**

Crude oil is one of the major products which includes global measurements. The origin of crude oil prediction errors involves composite supply-demand structures. Scientists have come across unique modes for exploring and forecasting crude oil prices. Studies related to the prediction of prices play an important role in the economic crisis. One of the features of the imperfection of all the methodologies was that the upcoming movement of oil price was derived from the prior data. Machine learning strategies came into existence for oil price prediction. In recent times, many studies have given more focus on the Convolutional neural network which is a neural network based on deep learning concepts.

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

**2.2 REFERENCES**

[1] Mohammad Reza Mahdiani and Ehsan Khamehchi, “A modified neural network model for predicting the crude oil price”, Intellectual Economics, vol. 10, no. 2, pp. 71-77, Aug. 2016.

[2] Manel Hamdi and Chaker Aloui, “Forecasting Crude Oil Price Using Artificial Neural Networks: A Literature Survey,” Economics Bulletin, AccessEcon, vol. 35, no. 2, pp. 1339-1359, 2015.

[3] Yu Runfang, Du Jiangze and Liu Xiaotao, “Improved Forecast Ability of Oil Market Volatility Based on combined Markov Switching and GARCH-class Model, Procedia Computer Science, vol. 122, pp. 415-422, 2017.

[4] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink and J. Schmidhuber, “LSTM: A Search Space Odyssey,” IEEE Transactions on Neural Networks and Learning Systems, vol. 28, no. 10, pp. 2222-2232, Oct. 2017.

[5] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computing, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[6] Jammazi, R., Aloui, C.: Crude oil price forecasting: experimental evidence from wavelet decomposition and neural network modelling. Energy Econ. 34(3), 828–841 (2012).

[7] S. Moshiri, and F. Foroutan, “Forecasting nonlinear crude oil futures prices,” The Energy Journal vol. 27, pp. 81-95, 2005.

[8] Siddhi Vinayak Kulkarni and Imad Haidar, Forecasting Model for Crude Oil Price Using Artificial Neural Networks and Commodity Futures Prices. International Journal of Computer Science and Information Security, vol. 2, no.1, June 2009.

[9] Hamdi and Aloui, “Machine learning approach for crude oil price prediction with Artificial Neural Networks-Quantitative (ANN-Q) model,” The 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, pp. 1-8, 2010.

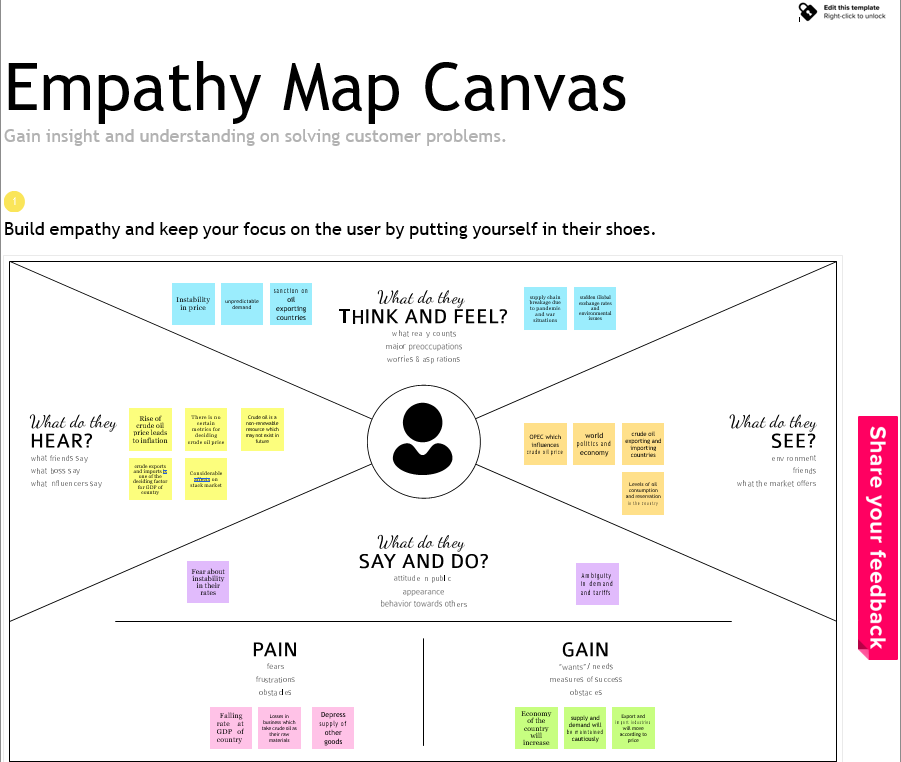
[10] Abdullah and Zeng.: Exploring the core factors and its dynamic effects on oil price: An application on path analysis and BVAR-TVP model. Energy Policy 39(12), 8022–8036 (2011).

**2.3 PROBLEM STATEMENT DEFINITION**

Nowadays, the increased oil prices worldwide are having a great impact on all economic activities. Over the years there has been a fluctuation in petroleum prices, and a close consideration of the demand and supply side effects that sparked these price changes shows there is high probability that these changes will continue in the outlook period and beyond. West African Monetary Agency (2008) concluded that increase in world oil prices have been shown to worsen fiscal deficit positions of oil importing countries like Ghana. For this reason, we believe that if the government can see ahead of monthly petroleum prices, our deficit would not be worsened. The ability to forecast these changes in oil prices allows economic participants such as firms to adapt to future market changes and provides decision makers with accurate information with which they can use to select the optimal decision for them. As such, it is vital that we develop a robust model that can forecast the prices of oil and these changes as accurately as possible

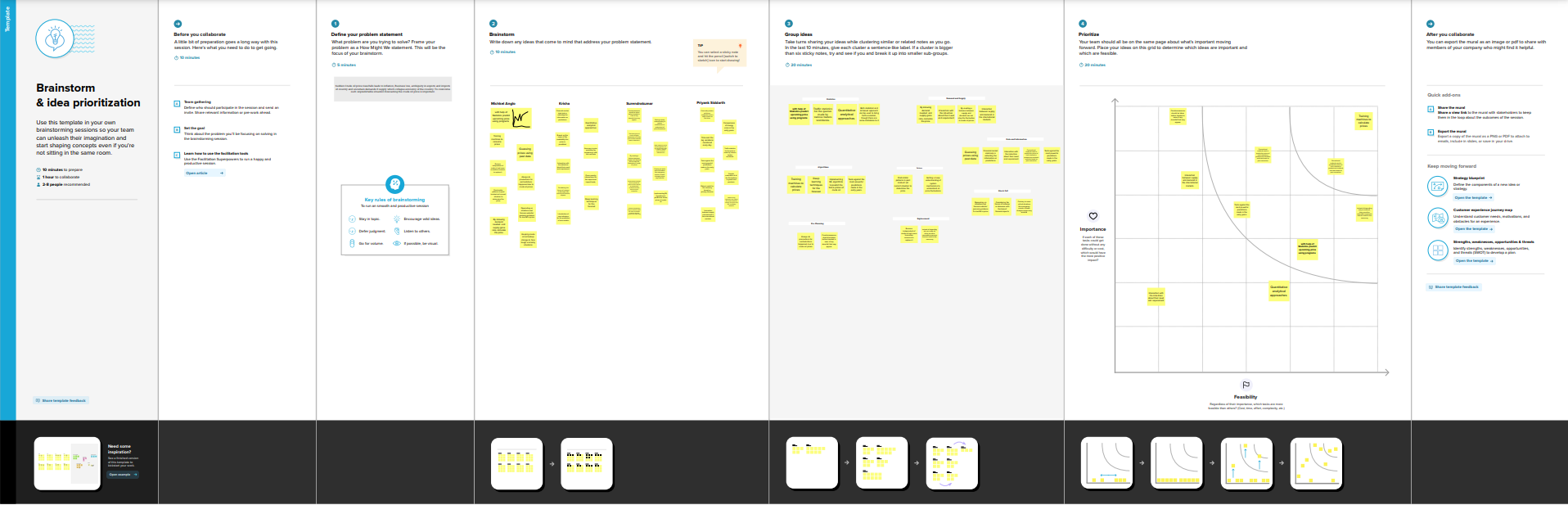
**CHAPTER 3**

**IDEATION & PROPOSED SOLUTION**

**3.1 EMPATHY MAP CANVAS**

**Figure 1.1. Empathy map canvas**

**3.2 IDEATION & BRAINSTORMING**

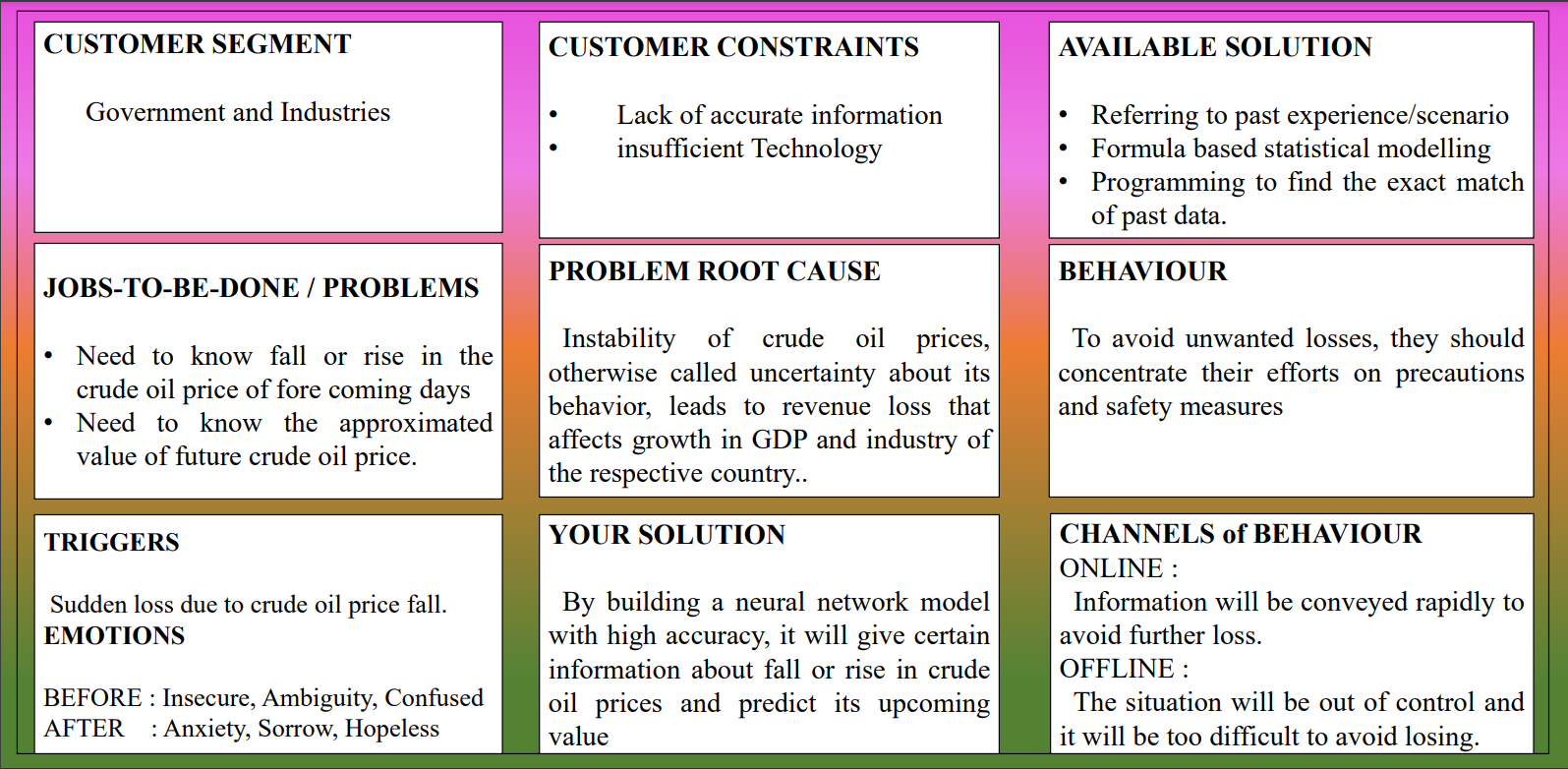
****

**Figure 1.2. Brainstorming & Ideation**

**3.3 PROPOSED SOLUTION**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Description** |
| 1. | Problem Statement (Problem to be solved) | Uncertainty in crude oil prices rises or falls, and ambiguity in this growth leads to serious effects in companies and the economy, which in turn affect the nation’s GDP. |
| 2. | Idea / Solution description | A model will be built by the combination of machine learning with deep learning techniques. It will function with an artificial neutral network like how the human brain  functions. |
| 3. | Novelty / Uniqueness | Nowadays, the world is moving more into a digital world, like automating everything. This deep learning approach will be revolutionary, giving solutions with the highest  accuracy ever achieved. |
| 4. | Social Impact / Customer Satisfaction | It prevents companies from losses in their revenue. The GDP of that nation  will gradually increase. |
| 5. | Business Model (Revenue Model) | Crude oil is one of the most traded oils, which contributes to export companies' marketing revenue and the country’s GDP. So, this model will become an undeniable product or source for almost all companies to  retain their positions. |
| 6. | Scalability of the Solution | Deep learning is a widely developed technique that comes with new innovations frequently. Due to its fast-growing nature, the model created by this will be used for many year. |

**3.4 PROBLEM SOLUTION FIT**

****

**Figure 1.3. Problem solution fit canvas**

**CHAPTER 4**

**REQUIRMENT ANALYSIS**

**4.1 FUNCTIONAL REQUIREMENT**

Solution Requirements (Functional & Non-functional)

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Registration through Gmail |
| FR-2 | User Confirmation | Confirmation via Email  Confirmation via OTP |
| FR-3 | User Solution | Model creation using python with necessary libraries needed |
| FR-4 | User Acknowledgement | Sending and receiving data will be made using flask  library |
| FR-5 | User Understanding | For better UI experience Angular, HTML and CSS |
| FR-6 | User Storage | Cloud data will be needed |
| FR-7 | User access | To access information a server needed |

**4.2 NON-FUNCTIONAL REQUIREMENTS**

# Non-functional Requirements:

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | Available in all Phone and laptop/computer systems. |
| NFR-2 | **Security** | High Level Technology supporting for Two level  authentication and Cryptography used. |
| NFR-3 | **Reliability** | Stable Internet connection |
| NFR-4 | **Performance** | High resolution screen for pictorial representation  available without lagging in view. |
| NFR-5 | **Availability** | Power needed for 24x7 |
| NFR-6 | **Scalability** | Online data will be feed into model for effective  prediction |

**CHAPTER 5**

**PROJECT DESIGN**

**5.1 Data Flow Diagram**

AUTHENTICATION

APPLICATION UI

USER

RESULT

DISPLAY

DATABASE

INPUT

VALUE

USER

ML MODEL

**Figure 1.4 Data flow map**

**USER STORIES**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| **Customer (Web**  **user)** | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | I can access my database  with this | High | Sprint-1 |
| **Customer (Cloud user)** | Access | USN-2 | As a user, I can access the model database | I can receive confirmation email & click confirm | High | Sprint-1 |
| **Customer (People)** | Linked-in registration | USN-3 | As a user, I can register for the application through Linked-in | I can register & access the database model with Linked- in Login | Low | Sprint-2 |
| **Customer Care Executive** | Gmail account | USN-4 | As a user, I can register for the application through Gmail | I can register and access the model | Medium | Sprint-1 |
| **Administrator** | Login | USN-5 | As a Admin, I can log into the application by entering email & password | I can access the model database directly | High | Sprint-1 |
| **Customer (User)** |  | USN-6 | As a user I can give input to the model through the website | I can get crude oil price | High | Sprint-3 |
| **Customer (User)** |  | USN-7 | As a user I can view the pictorial or graphical representation of crude price | I can insights on crude oil price | High | Sprint-4 |

**5.2 SOLUTION ARCHITECTURE**

# SOLUTION ARCHITECTURE

Crude oil price can be predicted using AI model. AI model will be built with help of deep learning technique.

# The Technique used for deep learning is RNN (recurrent neutral network). Python Programming will be used for building. In python, NumPy, pandas, TensorFlow and keras libraries will be used

Data splitting into train and test data in the ratio 8:2

Data Pre-processing for making model efficient

Data Visualisation for getting insights

Creating dataset/

Downloading from website (Kaggle)

Model Building using python with keras and tensorflow modules

Validate the model with help of test dataset and calculate error

Train the model using train dataset and save the model

Building artificial neural network by adding input, hidden and output layer

Integrate model into front end with help of flask library

Build HTML and CSS file

**Figure 1.5 Solution map**

**TECHNICAL ARCHITECTURE**

**Project Design Phase-II Technology Stack (Architecture & Stack)**

Technical Architecture:

User Login

User Authentication with cloud database

User Giving value of price to predict next value (optional)

Cloud Storage

Model predicting default value and user given value

App UI

**Figure 1.6 Technical flow map**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1 | User Interface | How user access solution. Web application | HTML,CSS |
| 2 | Application Logic-1 | Logic for a process in the application | Python (flask) |
| 3 | Database | Data Type access, Configurations | MySQL |
| 4 | Cloud Database | Database Service on Cloud | IBM Cloud |
| 5 | File Storage | File storage requirements | IBM Block Storage & Local Filesystem |
| 6 | External API-1 | For standalone server | Firebase |
| 7 | Machine Learning Model | To predict upcoming price of crude oil | Recurrent neural network & LSTM |
| 8 | Infrastructure | Application Deployment on Local Server and local host address | Local, Firebase. |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Framework -1 | Python | Pandas, flask, NumPy, TensorFlow |
| 2. | Open-Source Framework -2 | Angular js | App module, component module |
| 3. | Open-Source Framework -3 | HTML & CSS | <div> and flex model |
| 4. | Security Implementations | User data will be stored according to CIA model | End to end encryption (SHA- 256) |
| 5. | Scalable Architecture | IBM cloud and firebase both used for better performance in storage and authentication | IBM Watson, Firebase, MySQL |
| 6. | Availability | Handle huge requests, avoid DDOS and XSS attack | Effective coding and restrictive user access based on need |
| 7. | Performance | Handle 100 to 10000 users to use server at a time | Flask |

**CHAPTER 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-2 | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | 10 | High | Krisha S |
| Sprint-3 | Access | USN-2 | As a user, I can access the model database | 5 | High | Michkel Anglo J,  Surendrakumar B. |
| Sprint-1 | Mobile registration | USN-3 | As a user, I can register for the application  Through mobile number | 10 | Low | Surendrakumar B |
| Sprint-1 | Mail Account Access | USN-4 | As a user, I can register for the application  through mail | 10 | Low | Priyank siddarth M J |
| Sprint-2 | Login | USN-5 | As a Admin, I can log into the application by  entering email & password | 10 | High | Krisha S,  Priyank siddarth M J |
| Sprint -3 | Internet Facility | USN--6 | As a user I can give input to the model through  the website | 15 | High | Michkel Anglo J |
| Sprint-4 | Dashboard | USN--7 | As a user I can view the pictorial or graphical  representation of crude price | 10 | Medium | Krisha s,  Surendrakumar B |
| Sprint-4 | Laptop or computer or mobile | USN-8 | As a user I can access Application using Interactive UI | 10 | Medium | Michkel Anglo J |

**6.2 SPRINT DELIVERY SCHEDULE**

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

**6.3 REPORTS FROM JIRA**

****

**Figure 1.7 Jira Milestone**

**CHAPTER 7**

**CODING & SOLUTIONING**

**MODEL BUILDING**

Python code used for model development in IBM

Pandas was used to control dataframe.Numpy was used for array manipulations.Matplotlip and seaborn are used for visualization. Tensorflow used for deep learning where scikit learn used for

import os, types

import pandas as pd

from botocore.client import Config

import ibm\_boto3

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import tensorflow

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import Sequential,load\_model,Model

from tensorflow.keras.layers import LSTM,Dense,Input

data = pd.read\_excel(body.read()) # used to read the file

Batch = 64

epochs = 120

timesteps = 30

Dataset was split into 64 batch with 30 step size.

#test set will be 20% of entire data set

length = length - int(length \*0.20)

print (length)

def get\_train\_length(dataset, batch\_size, test\_percent):

    # substract test\_percent to be excluded from training, reserved for testset

    length = len(dataset)

    length = length - int(length \*test\_percent)

    train\_length\_values = []

    for x in range(int(length) - 100,int(length)):

        modulo=x%batch\_size

        if (modulo == 0):

            train\_length\_values.append(x)

    return (max(train\_length\_values))

length = get\_train\_length(data, batch\_size, 0.20)

#Adding timesteps \* 2

upper\_train = length + timesteps\*2

data\_train = data[0:upper\_train]

training\_set = data\_train.iloc[:,1:2].values

scale = MinMaxScaler(feature\_range = (0, 1)) #scaling the dataset

training\_set\_scaled = scale.fit\_transform(np.float64(training\_set))

X\_train = []

y\_train = []

# Creating a data structure with n timesteps

print (length + timesteps)

for i in range(timesteps, length + timesteps):

    X\_train.append(training\_set\_scaled[i-timesteps:i,0])

    y\_train.append(training\_set\_scaled[i:i+timesteps,0])

Model Creation

inputs\_1\_mae = Input(batch\_shape=(batch\_size,timesteps,1))

lstm\_1\_mae = LSTM(10, stateful=True, return\_sequences=True)(inputs\_1\_mae)

lstm\_2\_mae = LSTM(10, stateful=True, return\_sequences=True)(lstm\_1\_mae)

output\_1\_mae = Dense(units = 1)(lstm\_2\_mae)

regressor\_mae = Model(inputs=inputs\_1\_mae, outputs = output\_1\_mae)

regressor\_mae.compile(optimizer='adam', loss = 'mae')

regressor\_mae.summary()

for i in range(epochs):

    print("Epoch: " + str(i))

    regressor\_mae.fit(X\_train, y\_train, shuffle=False, epochs = 1, batch\_size = batch\_size)

regressor\_mae.reset\_states()

Input layer was selected with 10 and consequences layer also selected with 10 input.At last it wil yield one output. For correcting it erroe itself, ‘adam’ was used

After test data was cleaned and predicted using model, it will undergo error evaluation between original and predicted value.

print(np.any(np.isnan(test\_set)))

print(np.any(np.isnan(y\_test)))

np.nan\_to\_num(test\_set,copy=False)

np.nan\_to\_num(y\_test,copy=False)

from sklearn.metrics import mean\_absolute\_error

mae = float(mean\_absolute\_error(test\_set[timesteps:len(y\_test)], y\_test[0:len(y\_test) - timesteps]))

print(mae)

import math

from sklearn.metrics import mean\_squared\_error

rmse = math.sqrt(mean\_squared\_error(test\_set[timesteps:len(y\_test)], y\_test[0:len(y\_test) - timesteps]))

print(rmse)

regressor\_mae.save("Crude\_oil\_LSTM\_prediction.h5")

# compress the file

!tar -zcvf crudeoil\_prediction\_model.tgz Crude\_oil\_LSTM\_prediction.h5

Flask File Integration with model and html

app.py file code

import bcrypt

import numpy as np

from flask import Flask, redirect, render\_template, request, session, url\_for

from pymongo import MongoClient

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import load\_model

model = load\_model('prediction\_model.h5')

#set app as a Flask instance

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

#encryption relies on secret keys so they could be run

app.secret\_key = "GReece@2"

# #connect to your Mongo DB database

def MongoDB():

    con\_string ="mongodb+srv://micky:power0!6@cluster0.ziiirqp.mongodb.net/?retryWrites=true&w=majority"

    client = MongoClient(con\_string )

    db = client['Predict\_app'] ## database anme

    user\_collection = db['userdata'] ## table name

    return user\_collection

records = MongoDB()

#assign URLs to have a particular route

@app.route("/", methods=['post', 'get'])

def index():

    message = ''

    #if method post in index

    if "email" in session:

        return redirect(url\_for("logged\_in"))

    if request.method == "POST":

        user = request.form.get("fullname")

        email = request.form.get("email")

        password1 = request.form.get("password1")

        password2 = request.form.get("password2")

        #if found in database showcase that it's found

        user\_found = records.find\_one({"name": user})

        email\_found = records.find\_one({"email": email})

        if user\_found:

            message = 'There already is a user by that name'

            return render\_template('index.html', message=message)

        if email\_found:

            message = 'This email already exists in database'

            return render\_template('index.html', message=message)

        if password1 != password2:

            message = 'Passwords should match!'

            return render\_template('index.html', message=message)

        else:

            #hash the password and encode it

            hashed = bcrypt.hashpw(password2.encode('utf-8'), bcrypt.gensalt())

            #assing them in a dictionary in key value pairs

            user\_input = {'name': user, 'email': email, 'password': hashed}

            #insert it in the record collection

            records.insert\_one(user\_input)

           #find the new created account and its email

            user\_data = records.find\_one({"email": email})

            new\_email = user\_data['email']

            #if registered redirect to logged in as the registered user

            return render\_template('logged\_in.html', email=new\_email)

    return render\_template('index.html')

@app.route("/login", methods=["POST", "GET"])

def login():

    message = 'Please login to your account'

    if "email" in session:

        return redirect(url\_for("logged\_in"))

    if request.method == "POST":

        email = request.form.get("email")

        password = request.form.get("password")

        #check if email exists in database

        email\_found = records.find\_one({"email": email})

        if email\_found:

            email\_val = email\_found['email']

            passwordcheck = email\_found['password']

            #encode the password and check if it matches

            if bcrypt.checkpw(password.encode('utf-8'), passwordcheck):

                session["email"] = email\_val

                return redirect(url\_for('logged\_in'))

            else:

                if "email" in session:

                    return redirect(url\_for("logged\_in"))

                message = 'Wrong password'

                return render\_template('login.html', message=message)

        else:

            message = 'Email not found'

            return render\_template('login.html', message=message)

return render\_template('login.html', message=message)

@app.route('/logged\_in')

def logged\_in():

    if "email" in session:

        email = session["email"]

        return render\_template('logged\_in.html', email=email)

    else:

        return redirect(url\_for("login"))

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

def predict():

    if request.method == 'POST':

        value=request.form.get('list') ## get the value that come with list name

        b=value.split(",") ## split the data using comma

        values = list(map(float, b)) ## create a list from string

        List= np.array([values]) ## convert it to array

        List=np.reshape(List,(-1,1))

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

        List=scale.fit\_transform(np.float64(List))

        val= model.predict(List) ## predict the value for give data

        val=np.reshape(val,(30,1))

        val=scale.inverse\_transform(val)

        answer= float(val[29])

        answer=round(answer, 2)

    return render\_template('prediction.html', result=answer)

@app.route("/logout", methods=["POST", "GET"])

def logout():

    if "email" in session:

        session.pop("email", None)

        return render\_template("signout.html")

else:

return render\_template('index.html')

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True, host='0.0.0.0', port=5000) ## run in local host 5000 port and any change in code will reflected immediately.

**CHAPTER 8**

**TESTING**

**8.1 TEST CASES**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test case ID** | **Feature Type** | **Component** | **Test Scenario** | **Steps To Execute** | **Test Data** |
| RegisterPage\_TC\_OO1 | UI | Home Page | Verify the UI elements working properly | 1.Enter URL and click go 2.Click on registration dropdown button, if not in register page 3.Verify register page | [https://120.0.0.1:5000/register](https://120.0.0.1:5000/register" \o "https://120.0.0.1:5000/register) |
| RegisterPage\_TC\_OO2 | Functional | Register Page | Verify user is able to give valid details then only allowing them to register successfully with notification | 1.Enter URL and click go 2.Verify Register page popup with below UI elements: a.email text box  b.password text box c.Username d.register button | [https://120.0.0.1:5000/register](https://120.0.0.1:5000/register" \o "https://120.0.0.1:5000/register) |
| RegisterPage\_TC\_OO3 | Functional | Home page | Verify user data successfully added into DB | 1.Enter and click go 2.Enter Valid username/email in Email text box 3.Enter valid password in password text box and click register button 5.Check whether data successfully added into MongoDB | name:MichkelAnglo J email: anglo1199038@gmail.com password: 123456 |
| LoginPage\_TC\_OO1 | Functional | Login page | Verify the UI elements working properly | 1.Enter URL and click go 2.Click on email id text box 3.click on password text box 5.Click on login button and check | [https://120.0.0.1:5000/login](https://120.0.0.1:5000/login" \o "https://120.0.0.1:5000/login) |
| LoginPage\_TC\_OO2 | Functional | Login page | Verify user is not able to log into application with InValid credentials | 1.Enter URL(https://shopenzer.com/) and click go 2.Enter Valid username/email in Email text box 3.Enter Invalid password in password text box 4.Click on login button | email: rajkumar@gmail.com password: 453245 |
| LoginPage\_TC\_OO3 | Functional | Login page | Verify user is able to log into application with Valid credentials | 1.Enter URL(https://shopenzer.com/) and click go 2.Enter Valid username/email in Email text box 3.Enter valid password in password text box 4.Click on login button | email: anglo1199038@gmail.com password: 123456 |
| LoginPage\_TC\_OO4 | Functional | Login page | After giving valid credentials it must redirect to home page | 1.Enter URL and click go 2.Enter Valid username/email in Email text box 3.Enter valid password in password text box 4.Click on login button and check whether it redirect to home page | [https://120.0.0.1:5000/login](https://120.0.0.1:5000/login" \o "https://120.0.0.1:5000/login) |
| HomePage\_TC\_001 | UI | Home Page | Verify the UI elements working properly | 1.Enter URL and click go 2.click Input data(text box) field for model  3.Enter predict button 4.check whether it redirect to result page | [https://120.0.0.1:5000/home](https://120.0.0.1:5000/home" \o "https://120.0.0.1:5000/home) |
| HomePage\_TC\_002 | Functional | Home Page | Check whether input for crude oil price model passed properly | 1.Enter URL and click go 2.click Input data(text box) field for model  3.Enter predict button 4.check whether it redirect to result page | 23.34,34.67,65,78,98,95,93,98.7,98.6,98 |
| HomePage\_TC\_003 | Functional | Home Page | Check total number of input got  as much as needed | 1.Enter URL and click go 2.click Input data(text box) field for model  3.Enter predict button 4.check whether it redirect to result page | 23.34,34.67,65,78,98,95,93,98.7,98.6,98 |
| Result\_TC\_001 | UI | Result Page | Verify the UI elements working properly | 1.Enter URL and click go 2.check whether page loaded properly | [https://120.0.0.1:5000/prediction](https://120.0.0.1:5000/prediction" \o "https://120.0.0.1:5000/prediction) |
| Result\_TC\_002 | Functional | Result Page | check model result displayed properly from api | 1.Enter URL and click go 2.check whether it display price prediction properly | [https://120.0.0.1:5000/prediction](https://120.0.0.1:5000/prediction" \o "https://120.0.0.1:5000/prediction) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test case ID** | **Expected Result** | **Actual Result** | **Status** | **BUG ID** | **Executed By** |
| RegisterPage\_TC\_OO1 | Register popup want to display properly | Working as expected | Pass | No Bug | Michkel Anglo J |
| RegisterPage\_TC\_OO2 | Application should show below UI elements: a.email text box b.password text box c.Name text box d.Register button with blue color | Working as expected | Pass | No Bug | Michkel Anglo J |
| RegisterPage\_TC\_OO3 | check Mongo DB compass with data passed are stored properly | Working as expected | Pass | No Bug | Priyank Siddarth |
| LoginPage\_TC\_OO1 | Login/Signup popup should display | Working as expected | Pass | No Bug | Krisha S |
| LoginPage\_TC\_OO2 | Application should show 'Incorrect email or password ' validation message. | Working as expected | Pass | No Bug | Krisha S |
| LoginPage\_TC\_OO3 | Application should show 'Incorrect email or password ' validation message. | Working as expected | Pass | No Bug | Surendrakumar B |
| LoginPage\_TC\_OO4 | User should navigate to user account homepage | Working as expected | Pass | No Bug | Surendrakumar B |
| HomePage\_TC\_001 | Application should show below UI elements: a.Description about crude oil b.input text box c.Predict button with blue color | Working as expected | Pass | No Bug | Priyank Siddarth |
| HomePage\_TC\_002 | User should navigate to Result page | Working as expected | Pass | No Bug | Michkel Anglo J |
| HomePage\_TC\_003 | User should navigate to Result page | Working as expected | Pass | No Bug | Michkel Anglo J |
| Result\_TC\_001 | User should able to see the prediction result price | Working as expected | Pass | No Bug | Surendrakumar B |
| Result\_TC\_002 | User should able to see the prediction result price | Working as expected | Pass | No Bug | Krisha S |

**8.2 USER ACCEPTANCE TESTING**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **UAT Task** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Mobile registration, Mail Account Access,login | UAT Design | US01,US02,US03 | Preparing UAT test cases for stories planned for current sprint | 10 | Medium | Surendrakumar B  Priyank siddarth M J |
| Sprint-2 | Mobile registration,Mail Account Access,login | UAT Execution | US01,US02,US03 | Executing UAT test cases against the UAT Environment | 15 | Medium | Michkel Anglo J  Priyank siddarth M J |
| Sprint-3 | Model development& Access | UAT Design | US04,US05,US06 | Preparing UAT test cases for stories planned for current sprint | 20 | High | Krisha S  Surendrakumar B |
| Sprint-4 | Model development& Access | UAT Execution | US04,US06,US05 | Executing UAT test cases against the UAT Environment | 20 | High | Michkel Anglo J,  Surendrakumar B,  Krisha S |
| Sprint-5 | Prediction Dashboard | UAT Design | US08,US07 | Preparing UAT test cases for stories planned for current sprint  I | 15 | High | Michkel Anglo J,  Surendrakumar B,  Krisha S |
| Sprint-6 | Prediction Dashboard | UAT Execution | US08,US07 | Executing UAT test cases against the UAT Environment | 20 | High | Michkel Anglo J,  Surendrakumar B,  Krisha S |

| **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 | 1 day | 10 Nov 2022 | 10 Nov 2022 | 20 | 10 Nov 2022 |
| Sprint-2 | 15 | 1 day | 11 Nov 2022 | 11 Nov 2022 | 15 | 11 Nov 2022 |
| Sprint-3 | 20 | 2 days | 12 Nov 2022 | 13 Nov 2022 | 20 | 13 Nov 2022 |
| Sprint-4 | 20 | 1 day | 14 Nov 2022 | 14 Nov 2022 | 20 | 14 Nov 2022 |
| Sprint-5 | 15 | 2 day | 15 Nov 2022 | 16 Nov 2022 | 15 | 16 Nov 2022 |
| Sprint-6 | 20 | 1 days | 17 Nov 2022 | 17 Nov 2022 | 20 | 17 Nov 2022 |

**UAT Execution & Report Submission**

# Purpose of Document

The purpose of this document 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 (UAT).

# Defect Analysis

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Resolution** | **Severity 1** | **Severity 2** | **Severity 3** | **Severity 4** | **Subtotal** |
| By Design | 7 | 2 | 1 | 4 | 14 |
| Duplicate | 1 | 0 | 0 | 0 | 1 |
| External | 0 | 1 | 0 | 1 | 2 |
| Fixed | 8 | 1 | 1 | 2 | 10 |
| Not Reproduced | 0 | 0 | 0 | 0 | 0 |
| Skipped | 1 | 0 | 0 | 1 | 1 |
| Won't Fix | 0 | 1 | 0 | 2 | 3 |
| Totals | 16 | 5 | 2 | 10 | 33 |

# Test Case Analysis

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Section** | **Total Cases** | **Not Tested** | **Fail** | **Pass** |
| Print Engine | 3 | 0 | 0 | 3 |
| UI | 7 | 0 | 0 | 7 |
| Security | 5 | 0 | 0 | 5 |
| Outsource Shipping | 1 | 0 | 0 | 1 |
| Exception Reporting | 6 | 0 | 0 | 6 |
| Final Report Output | 2 | 0 | 0 | 2 |
| Version Control | 1 | 0 | 0 | 1 |

**CHAPTER 9**

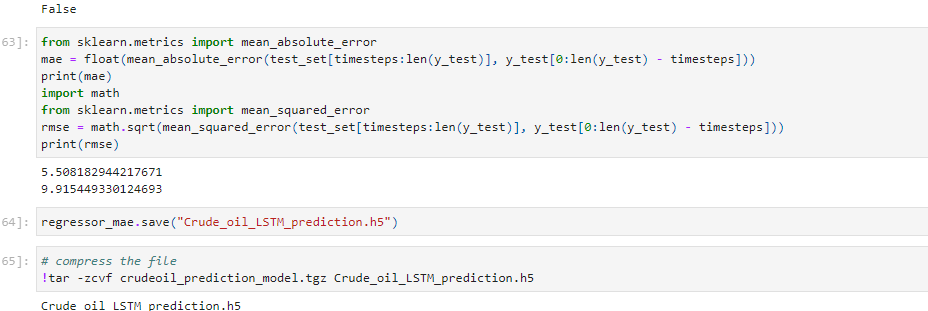
**RESULTS**

**Parameter:** Metrics

**Values:**

RNN with LSTM Model

1. MAE -5.56
2. MSE - 9.92



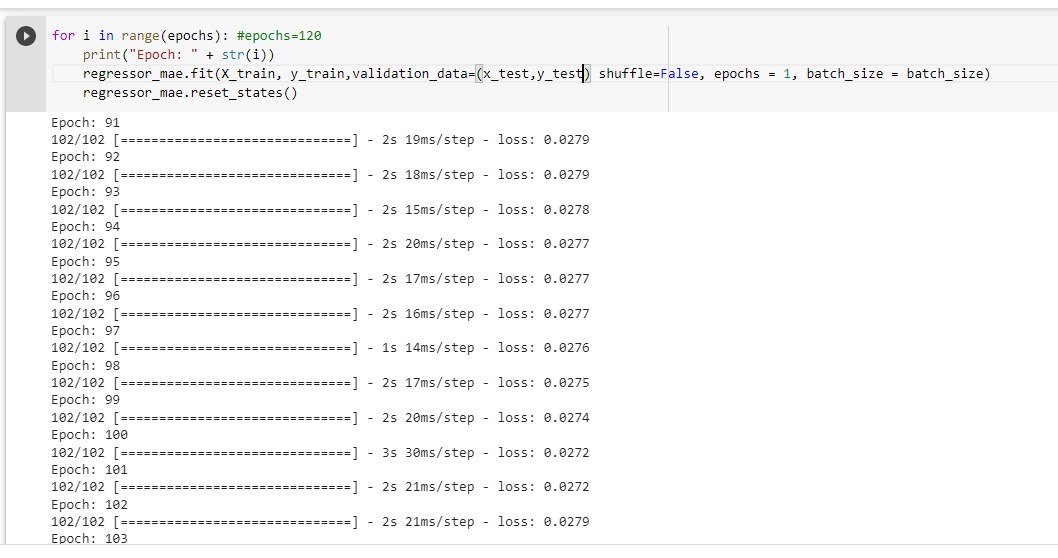
**Figure 1.8 Metrics**

**Parameter:** Tune the model

**Values:**

Hyper parameter Tuning - 120 epochs

Validation Method - Testing data

  
**Figure 1.9 Tuning**

**CHAPTER 10**

**ADVANTAGES AND DISADVANTAGES**

Advantages

* It will Provide stability for industries to make decisions with respect to crude oil
* Imports and exports of crude oil are made easier
* Due to its intelligence,predicting accuracy of crude oil price is high

Disadvantages

* Even though it trained with past prices datasets, time series will change for each period without restriction, so making model to be continuously learning was lacking.
* In time series, not all the time it follows seasonality or usual scenarios. So,predicting outlier data will be become challenging

**CHAPTER 11**

**CONCLUSION**

With help of RNN with LSTM layered network, future crude oil price was forecasted. Even though exact price of crude oil was unknown ,it will give intuition for industries to determine whether there is increment or decrement with high accuracy rate.

**CHAPTER 12**

**FUTURE SCOPE**

It can be developed to predict values very accurately by creating ensemble algorithms with this neutral network algorithm. But machine specifications needed to achieve this model is not yet easily available. Even though it is used it will consume more time or more cores to be processed at a time. Datasets used for this model will have plenty of features, to achieve that much multi dimensional is still difficult task.

**APPENDIX**

**SOURCE CODE**

Python code used for model development in IBM

import os, types

import pandas as pd

from botocore.client import Config

import ibm\_boto3

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import tensorflow

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import Sequential,load\_model,Model

from tensorflow.keras.layers import LSTM,Dense,Input

def \_\_iter\_\_(self): return 0

# @hidden\_cell

# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.

# You might want to remove those credentials before you share the notebook.

cos\_client = ibm\_boto3.client(service\_name='s3',

    ibm\_api\_key\_id='CksY-92OBcPex5E-MwaGfkfoTkQW6wen5HVtZ1LXGMOK',

    ibm\_auth\_endpoint="https://iam.cloud.ibm.com/oidc/token",

    config=Config(signature\_version='oauth'),

    endpoint\_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'crudeoilpriceprediction-donotdelete-pr-vtg7hvlasgnhuz'

object\_key = 'Crude Oil Prices Daily.xlsx'

body = cos\_client.get\_object(Bucket=bucket,Key=object\_key)['Body']

data = pd.read\_excel(body.read())

# @hidden\_cell

# The following code contains the credentials for a file in your IBM Cloud Object Storage.

# You might want to remove those credentials before you share your notebook.

metadata\_1 = {

    'IAM\_SERVICE\_ID': 'iam-ServiceId-5d623ebb-fdda-4e0c-baa0-104a5991e97f',

    'IBM\_API\_KEY\_ID': 'CksY-92OBcPex5E-MwaGfkfoTkQW6wen5HVtZ1LXGMOK',

    'ENDPOINT': 'https://s3.private.us.cloud-object-storage.appdomain.cloud',

    'IBM\_AUTH\_ENDPOINT': 'https://iam.cloud.ibm.com/oidc/token',

    'BUCKET': 'crudeoilpriceprediction-donotdelete-pr-vtg7hvlasgnhuz',

    'FILE': 'Crude Oil Prices Daily.xlsx'

}

batch\_size = 64

epochs = 120

timesteps = 30

length = len(data)

print (length)

#test set will be 20% of entire data set

length = length - int(length \*0.20)

print (length)

def get\_train\_length(dataset, batch\_size, test\_percent):

    # substract test\_percent to be excluded from training, reserved for testset

    length = len(dataset)

    length = length - int(length \*test\_percent)

    train\_length\_values = []

    for x in range(int(length) - 100,int(length)):

        modulo=x%batch\_size

        if (modulo == 0):

            train\_length\_values.append(x)

    return (max(train\_length\_values))

length = get\_train\_length(data, batch\_size, 0.20)

#Adding timesteps \* 2

upper\_train = length + timesteps\*2

data\_train = data[0:upper\_train]

training\_set = data\_train.iloc[:,1:2].values

scale = MinMaxScaler(feature\_range = (0, 1)) #scaling the dataset

training\_set\_scaled = scale.fit\_transform(np.float64(training\_set))

X\_train = []

y\_train = []

# Creating a data structure with n timesteps

print (length + timesteps)

for i in range(timesteps, length + timesteps):

    X\_train.append(training\_set\_scaled[i-timesteps:i,0])

    y\_train.append(training\_set\_scaled[i:i+timesteps,0])

print (len(X\_train))

print (len (y\_train))

print (np.array(X\_train).shape)

print (np.array(y\_train).shape)

# Creating a data structure with n timestep

for i in range(timesteps, length + timesteps):

    X\_train.append(training\_set\_scaled[i-timesteps:i,0])

    y\_train.append(training\_set\_scaled[i:i+timesteps,0])

# Reshaping

X\_train, y\_train = np.array(X\_train), np.array(y\_train)

X\_train = np.reshape(X\_train, (X\_train.shape[0], X\_train.shape[1], 1))

y\_train = np.reshape(y\_train, (y\_train.shape[0], y\_train.shape[1], 1))

Model Creation

inputs\_1\_mae = Input(batch\_shape=(batch\_size,timesteps,1))

lstm\_1\_mae = LSTM(10, stateful=True, return\_sequences=True)(inputs\_1\_mae)

lstm\_2\_mae = LSTM(10, stateful=True, return\_sequences=True)(lstm\_1\_mae)

output\_1\_mae = Dense(units = 1)(lstm\_2\_mae)

regressor\_mae = Model(inputs=inputs\_1\_mae, outputs = output\_1\_mae)

regressor\_mae.compile(optimizer='adam', loss = 'mae')

regressor\_mae.summary()

for i in range(epochs):

    print("Epoch: " + str(i))

    regressor\_mae.fit(X\_train, y\_train, shuffle=False, epochs = 1, batch\_size = batch\_size)

    regressor\_mae.reset\_states()

Testing

def get\_test\_length(dataset, batch\_size):

    test\_length\_values = []

    for x in range(len(dataset) - 200, len(dataset) - timesteps\*2):

        modulo=(x-upper\_train)%batch\_size

        if (modulo == 0):

            test\_length\_values.append(x)

            print (x)

    return (max(test\_length\_values))

test\_length = get\_test\_length(data, batch\_size)

print(test\_length)

upper\_test = test\_length + timesteps\*2

testset\_length = test\_length - upper\_train

print (testset\_length)

# construct test set

#subsetting

df\_data\_1\_test = data[upper\_train:upper\_test]

test\_set = df\_data\_1\_test.iloc[:,1:2].values

#scaling

scaled\_real\_bcg\_values\_test = scale.fit\_transform(np.float64(test\_set))

#creating input data

X\_test = []

for i in range(timesteps, testset\_length + timesteps):

    X\_test.append(scaled\_real\_bcg\_values\_test[i-timesteps:i, 0])

X\_test = np.array(X\_test)

#reshaping

X\_test = np.reshape(X\_test, (X\_test.shape[0], X\_test.shape[1], 1))

#prediction

predicted\_bcg\_values\_test\_mae = regressor\_mae.predict(X\_test, batch\_size=batch\_size)

regressor\_mae.reset\_states()

print (predicted\_bcg\_values\_test\_mae.shape)

#reshaping

predicted\_bcg\_values\_test\_mae = np.reshape(predicted\_bcg\_values\_test\_mae,

                                       (predicted\_bcg\_values\_test\_mae.shape[0],

                                        predicted\_bcg\_values\_test\_mae.shape[1]))

print (predicted\_bcg\_values\_test\_mae.shape)

#inverse transform

predicted\_bcg\_values\_test\_mae = scale.inverse\_transform(predicted\_bcg\_values\_test\_mae)

#creating y\_test data

y\_test = []

for j in range(0, testset\_length - timesteps):

    y\_test = np.append(y\_test, predicted\_bcg\_values\_test\_mae[j, timesteps-1])

# reshaping

y\_test = np.reshape(y\_test, (y\_test.shape[0], 1))

print (y\_test.shape)

plt.plot(test\_set[timesteps:len(y\_test)], color = 'red', label = 'Real Crude Oil Prices')

plt.plot(y\_test[0:len(y\_test) - timesteps], color = 'blue', label = 'Predicted Crude Oil Prices')

plt.title('Crude Oil Prices Prediction - MAE')

plt.xlabel('Time')

plt.ylabel('Crude Oil Prices')

plt.legend()

plt.show()

print(np.any(np.isnan(test\_set)))

print(np.any(np.isnan(y\_test)))

np.nan\_to\_num(test\_set,copy=False)

np.nan\_to\_num(y\_test,copy=False)

from sklearn.metrics import mean\_absolute\_error

mae = float(mean\_absolute\_error(test\_set[timesteps:len(y\_test)], y\_test[0:len(y\_test) - timesteps]))

print(mae)

import math

from sklearn.metrics import mean\_squared\_error

rmse = math.sqrt(mean\_squared\_error(test\_set[timesteps:len(y\_test)], y\_test[0:len(y\_test) - timesteps]))

print(rmse)

regressor\_mae.save("Crude\_oil\_LSTM\_prediction.h5")

# compress the file

!tar -zcvf crudeoil\_prediction\_model.tgz Crude\_oil\_LSTM\_prediction.h5

IBM Deployment

pip install ibm\_watson\_machine\_learning

from ibm\_watson\_machine\_learning import APIClient

wmi\_credentials={

    "url":"https://us-south.ml.cloud.ibm.com",

    "apikey":"9RsKJwBgAvBgaBdJb5NY4pFp2hP-z\_rV3jEVTEkRNdmm"

}

client = APIClient(wmi\_credentials)

space\_id="90c5aed6-96c9-4761-b1d2-eab8fbdd04db"

client.set.default\_space(space\_id)

software\_space\_uid=client.software\_specifications.get\_id\_by\_name("tensorflow\_rt22.1-py3.9")

model\_details=client.repository.store\_model(model='crudeoil\_prediction\_model.tgz',meta\_props={

    client.repository.ModelMetaNames.NAME:"LSTM Model",

    client.repository.ModelMetaNames.TYPE:"tensorflow\_2.7",

    client.repository.ModelMetaNames.SOFTWARE\_SPEC\_UID:software\_space\_uid

})

Flask File Integration with model and html

app.py file code

import bcrypt

import numpy as np

from flask import Flask, redirect, render\_template, request, session, url\_for

from pymongo import MongoClient

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import load\_model

model = load\_model('prediction\_model.h5')

#set app as a Flask instance

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

#encryption relies on secret keys so they could be run

app.secret\_key = "GReece@2"

# #connect to your Mongo DB database

def MongoDB():

    con\_string ="mongodb+srv://micky:power0!6@cluster0.ziiirqp.mongodb.net/?retryWrites=true&w=majority"

    client = MongoClient(con\_string )

    db = client['Predict\_app']

    user\_collection = db['userdata']

    return user\_collection

records = MongoDB()

##Connect with Docker Image###

#assign URLs to have a particular route

@app.route("/", methods=['post', 'get'])

def index():

    message = ''

    #if method post in index

    if "email" in session:

        return redirect(url\_for("logged\_in"))

    if request.method == "POST":

        user = request.form.get("fullname")

        email = request.form.get("email")

        password1 = request.form.get("password1")

        password2 = request.form.get("password2")

        #if found in database showcase that it's found

        user\_found = records.find\_one({"name": user})

        email\_found = records.find\_one({"email": email})

        if user\_found:

            message = 'There already is a user by that name'

            return render\_template('index.html', message=message)

        if email\_found:

            message = 'This email already exists in database'

            return render\_template('index.html', message=message)

        if password1 != password2:

            message = 'Passwords should match!'

            return render\_template('index.html', message=message)

        else:

            #hash the password and encode it

            hashed = bcrypt.hashpw(password2.encode('utf-8'), bcrypt.gensalt())

            #assing them in a dictionary in key value pairs

            user\_input = {'name': user, 'email': email, 'password': hashed}

            #insert it in the record collection

            records.insert\_one(user\_input)

            #find the new created account and its email

            user\_data = records.find\_one({"email": email})

            new\_email = user\_data['email']

            #if registered redirect to logged in as the registered user

            return render\_template('logged\_in.html', email=new\_email)

    return render\_template('index.html')

@app.route("/login", methods=["POST", "GET"])

def login():

    message = 'Please login to your account'

    if "email" in session:

        return redirect(url\_for("logged\_in"))

    if request.method == "POST":

        email = request.form.get("email")

        password = request.form.get("password")

        #check if email exists in database

        email\_found = records.find\_one({"email": email})

        if email\_found:

            email\_val = email\_found['email']

            passwordcheck = email\_found['password']

            #encode the password and check if it matches

            if bcrypt.checkpw(password.encode('utf-8'), passwordcheck):

                session["email"] = email\_val

                return redirect(url\_for('logged\_in'))

            else:

                if "email" in session:

                    return redirect(url\_for("logged\_in"))

                message = 'Wrong password'

                return render\_template('login.html', message=message)

        else:

            message = 'Email not found'

            return render\_template('login.html', message=message)

    return render\_template('login.html', message=message)

@app.route('/logged\_in')

def logged\_in():

    if "email" in session:

        email = session["email"]

        return render\_template('logged\_in.html', email=email)

    else:

        return redirect(url\_for("login"))

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

def predict():

    if request.method == 'POST':

        value=request.form.get('list')

        #print(type(value))

        b=value.split(",")

        values = list(map(float, b))

        List= np.array([values])

        List=np.reshape(List,(-1,1))

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

        List=scale.fit\_transform(np.float64(List))

        val= model.predict(List)

        val=np.reshape(val,(30,1))

        val=scale.inverse\_transform(val)

        answer= float(val[29])

        answer=round(answer, 2)

    return render\_template('prediction.html', result=answer)

@app.route("/logout", methods=["POST", "GET"])

def logout():

    if "email" in session:

        session.pop("email", None)

        return render\_template("signout.html")

    else:

        return render\_template('index.html')

if \_\_name\_\_ == "\_\_main\_\_":

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

HTML files

Base.html file

<!doctype html>

<html lang="en">

<head><meta charset="utf-8">

  <title>{% block title %}{%endblock %}</title>

  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" integrity="sha384-9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk" crossorigin="anonymous">

</head>

<body>

<nav class="navbar navbar-light" style="background-color: #7dafd3;">

  <a class="navbar-brand" href="#">Registration Form</a>

  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation">

    <span class="navbar-toggler-icon"></span>

  </button>

  <div class="collapse navbar-collapse" id="navbarNavAltMarkup">

    <div class="navbar-nav">

      <a class="nav-item nav-link active" href="/">Register <span class="sr-only">(current)</span></a>

      <a class="nav-item nav-link" href="/login">Login</a>

      <a class="nav-item nav-link" href="/logout">Logout</a>

    </div>

  </div>

</nav>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo" crossorigin="anonymous"></script>

    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js" integrity="sha384-OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI" crossorigin="anonymous"></script>

    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js" integrity="sha384-OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI" crossorigin="anonymous"></script>

{% block content %}{% endblock %}

</body>

</html>

Index.html file

{% extends "base.html" %}

{% block title %}Register System{% endblock %}

{% block content %}

{% if message  %}

    <div class="alert alert-secondary" role="alert">

        <p>{{ message }}</p>

    </div>

{% endif %}

<span class="test">

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

    <div class="form-group">

        <label for="Fullname"  class="fonts">Full name</label>

        <input name="fullname" class="form-control" id="inputFullName" aria-describedby="emailHelp" placeholder="Enter full name">

        <small id="fullName" class="form-text text-muted">Please enter your full name(First name and Last name)</small>

  </div>

  <div class="form-group">

        <label for="InputEmail"  class="fonts">Email address</label>

        <input name="email" class="form-control" id="InputEmail"  placeholder="Enter email">

        <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>

  </div>

  <div class="form-group">

    <label for="InputPassword"  class="fonts">Password</label>

    <input type="password" name="password1" class="form-control" id="InputPassword" placeholder="Password">

  </div>

    <div class="form-group">

    <label for="InputPassword2"  class="fonts">Repeat Password</label>

    <input type="password" name="password2" class="form-control" id="InputPassword2" placeholder="Repeat Password">

    </div>

  <button type="submit" class="btn btn-primary">Submit</button>

</form>

</span>

<style>

    html {

   height: 100%;

}

.test{

/\* Remember to use the other versions for IE 10 and older browsers! \*/

display: flex;

justify-content: center;

align-items: center;

min-height: 100%;

font-family: 'lato', sans-serif;

color: rgb(47, 206, 198);

 /\* W3C, IE 10+/ Edge, Firefox 16+, Chrome 26+, Opera 12+, Safari 7+ \*/

}

.fonts{

color: black;

}

</style>

{% endblock %}

Logged\_in.html file

{% extends "base.html" %}

{% block title %}Youv'e logged in {% endblock %}

{% block content %}

<p class="second">Crude oil is mainly used as a fuel and combustible, but is also indispensable as a chemical raw material. It is the foundation of modern life and in virtually every product around us – from smartphones and vehicle parts to wind turbines.<br>

</p>

<p class="second">Crude oil is one of the world's most important commodities and its price can have ripple effects through the broader economy. Rising oil prices mean higher gasoline prices at the pump, higher shipping costs, and increased input costs for producers.

</p>

<p class="second"><b>Here you can know about forecasting of crude oil price</b></p>

<form  class="todo" action="prediction" method="post">

    <div>

        Type last 30 days crude oil price<br>

        <input class="todo" type="text" name="list">

    </div>

<br>

<input type="submit" name="submit" value= "To predict click here!!!" href="/prediction" >

</form>

<style>

    .todo{

        padding-left: 10%;

        padding-right: 10%;

        align-items: center;

        justify-content: center;

    }

    .design{

        color: brown;

        font-size: medium;

        font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;

        font-display: solid;

    }

    .second{

        color: rgb(104, 26, 39);

        font-size: large;

        padding-top: 4%;

        padding-left: 2%;

        padding-right: 2%;

        font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;

        font-display: solid;

        align-items: center;

        justify-content: center;

    }

</style>

{% endblock %}

Login.html file

{% extends "base.html" %}

<span class="test">

    {% block title %}Login System{% endblock %}

{% block content %}</span>

{% if message  %}

    <div class="alert alert-secondary" role="alert">

        <p>{{ message }}</p>

    </div>

{% endif %}

<style>

    html {

   height: 100%;

}

.test{

/\* Remember to use the other versions for IE 10 and older browsers! \*/

display: flex;

justify-content: center;

align-items: center;

min-height: 100%;

font-family: 'lato', sans-serif;

color: #fff;

 /\* W3C, IE 10+/ Edge, Firefox 16+, Chrome 26+, Opera 12+, Safari 7+ \*/

}

.fonts{

color: black;

}

</style>

<span class="test">

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

  <div class="form-group">

      <label for="InputEmail" class="fonts">Email address</label>

      <input  name="email" class="form-control" id="InputEmail"  placeholder="Enter email/number">

      <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>

  </div>

    <div class="form-group">

      <label for="InputPassword" class="fonts">Password</label>

      <input type="password" name="password" class="form-control" id="InputPassword" placeholder="Password">

  </div>

  <button type="submit" class="btn btn-primary">Submit</button>

</form> </span>

{% endblock %}

Prediction.html file

<body class="todo">

    <b>

<p> From given input this is the crude oil price of next day</p>

<p>The future price of crude oil will be Rs.{{result}}.</p></b>

</body>

<style>

    .todo{

        background-color: rgb(101, 221, 221);

        padding-left: 34%;

        padding-right: 30%;

        padding-top: 18%;

        padding-bottom: 18%;

        font-size: xx-large;

    }

</style>

Signout.html file

{% extends "base.html" %}

{% block title %}Register System{% endblock %}

{% block content %}

<h1 class="design">You are signed out!</h1>

<style>

    .design{

        color: brown;

        font-size: x-large;

        font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;

        font-display: solid;}

        </style>

{% endblock %}

**GITHUB LINK**

<https://github.com/IBM-EPBL/IBM-Project-7228-1658850306>

**PROJECT DEMO LINK**

**<https://youtu.be/vh4QDrpWHjc>**

**--------------------- Thank you --------------------**