CRUDE OIL PRICE PREDICTION

Bonafede record of work done by

KAMALRAJ D	(19Z324)
MAHESH BOOPATHY M	(19Z330)
VINOTH SUBBIAH PL	(19Z362)
SRINIVASAN G	(20Z465)

Professional Readiness for Innovation, Employability, and Entrepreneurship

GUIDE: VANI K

BACHELOR OF ENGINEERING

BRANCH: COMPUTER SCIENCE AND ENGINEERING

NOVEMBER 2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PSG COLLEGE OF TECHNOLOGY
(Autonomous Institution)

COIMBATORE – 641 004

Table of Contents

CHAPTER 1

INTRODUCTION

1.1 Project Overview

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

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

1.2 Purpose

This project helps the People working in the investment of crude oil needs earlier crude oil price prediction system, which can help them to find the right time to buy crude oil so that they can increase profit from the purchase and reduce any substantial loss

CHAPTER 2

LITERATURE SURVEY

2.1 Existing Problem

Crude oil is one of the most significant sources of energy available today. With over a third of all energy consumed worldwide, it continues to be the most popular fuel. In the current environment, when technology is taking over our lives and efforts are being made to reduce the need for human labor, the Artificial Neural Network Technique has emerged as one of the most valuable techniques for data prediction. This paper offers a method for predicting oil prices that uses an artificial neural network (Sigmoid Function with the Learning Algorithm). Complex and non-linear interactions between input and output can be modeled using ANNs. The ability of ANN to generalize allows it to infer relationships even in the absence of data or input after learning from the inputs. A trustworthy method for creating predictions, ANN also learns from hidden relationships in the data without imposing any fixed relationships on the data. Many economists and analysts forecast the price of crude oil using data transformation and regression techniques like autoregressive moving average (ARMA) models and vector autoregressive (VAR) models, each time using a different input value. They then plot the graph with their forecasted prices while considering the main economic factors.

2.2 Problem Definition

The business people who invest or works in the crude oil field and petrol bunk owners requires a way to predict the crude oil price for the next day so that they can take some major investment decisions which can lead to business profit or reduce loss.

Persons working in the investment of crude oil needs a earlier crude oil price prediction system, which can help them to find the right time to buy crude oil so that they can increase profit from the purchase and reduce any substantial loss.

To develop a system that predicts the crude oil prices using LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) and to display the results using python-flask app.

2.3 References

1. “A comparison between AdaBoost-LSTM and AdaBoost-GRU for improving forecast prediction”. Ganiyu Adewale Busari, Dong Hoon Lim. (2021)
2. “Crude Oil Price Prediction Using LSTM Networks”. Varun Gupta, Ankit Pandey. (2018)
3. “Crude Oil Price Prediction using Artificial Neural Network”. Nalini Gupta, Shobhit Nigam
4. “Crude Oil Price Forecasting based on Support Vector Machines”. Wen Xie, Lean Yu, Shanying Xu, and Shouyang Wang.
5. LSTM. https://blog.mlreview.com/understanding-lstm-and-its-diagrams-37e2f46f1714 6. GRU. https://medium.com/geekculture/understanding-basic-architecture-of-lstm-gru-diagramm atically-6365befc64d

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

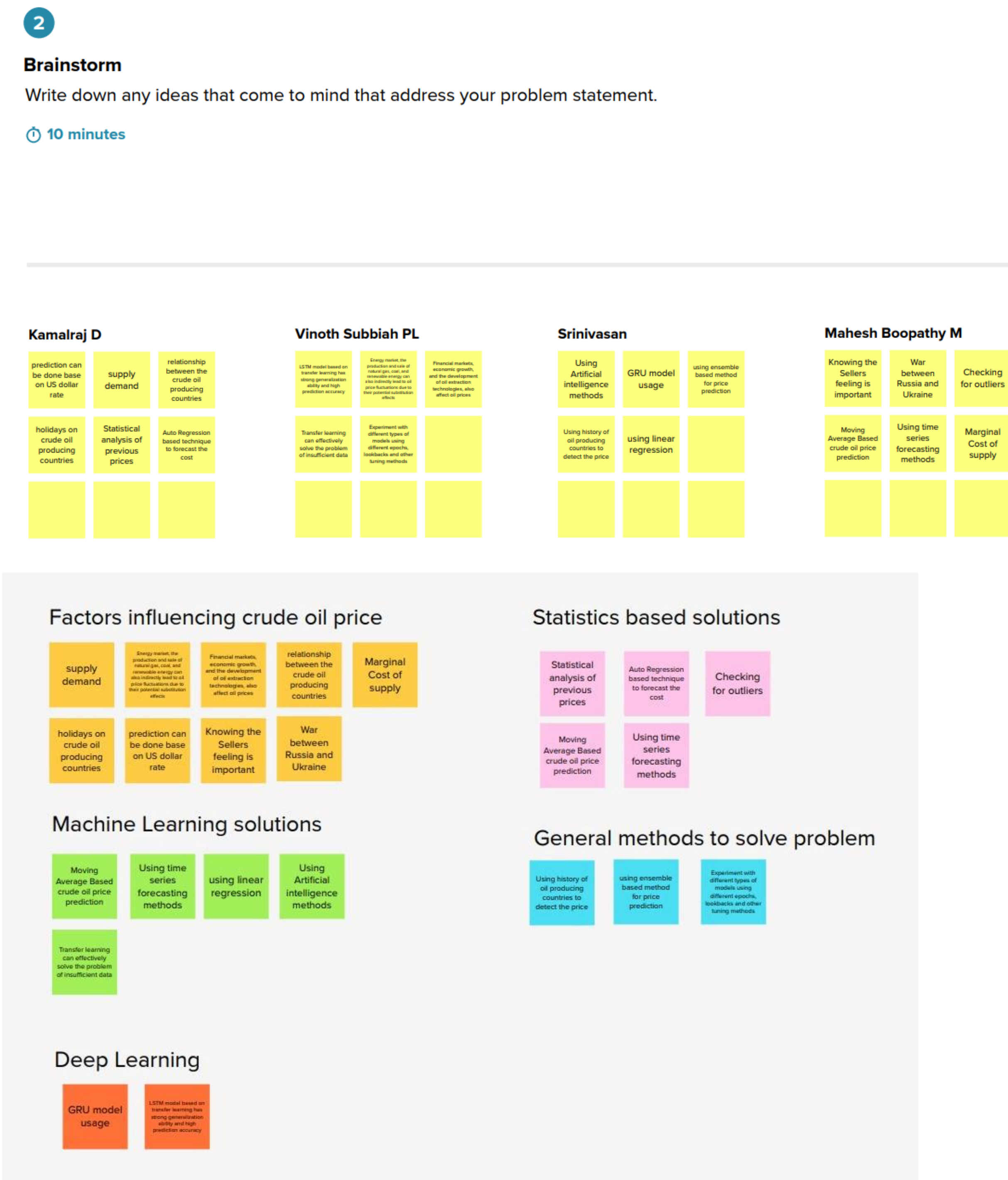
3.1 Empathy Map

The primary purpose of the empathy map is to bridge the understanding of the user and developer. Figure 3.1 represents the empathy map for the Crude oil Price Prediction System.

Figure 3.1 – Empathy Map

3.2 Ideation and Brainstorming

This is often the most exciting stage in a project, because during Ideation and brainstorming, the aim is to generate a large quantity of ideas that the team can then filter and cut down into the best, most practical, or most innovative ones to inspire new and better design solutions and products. Figure 3.2 shows the stages of ideation and brainstorming for the Crude oil Price Prediction System.



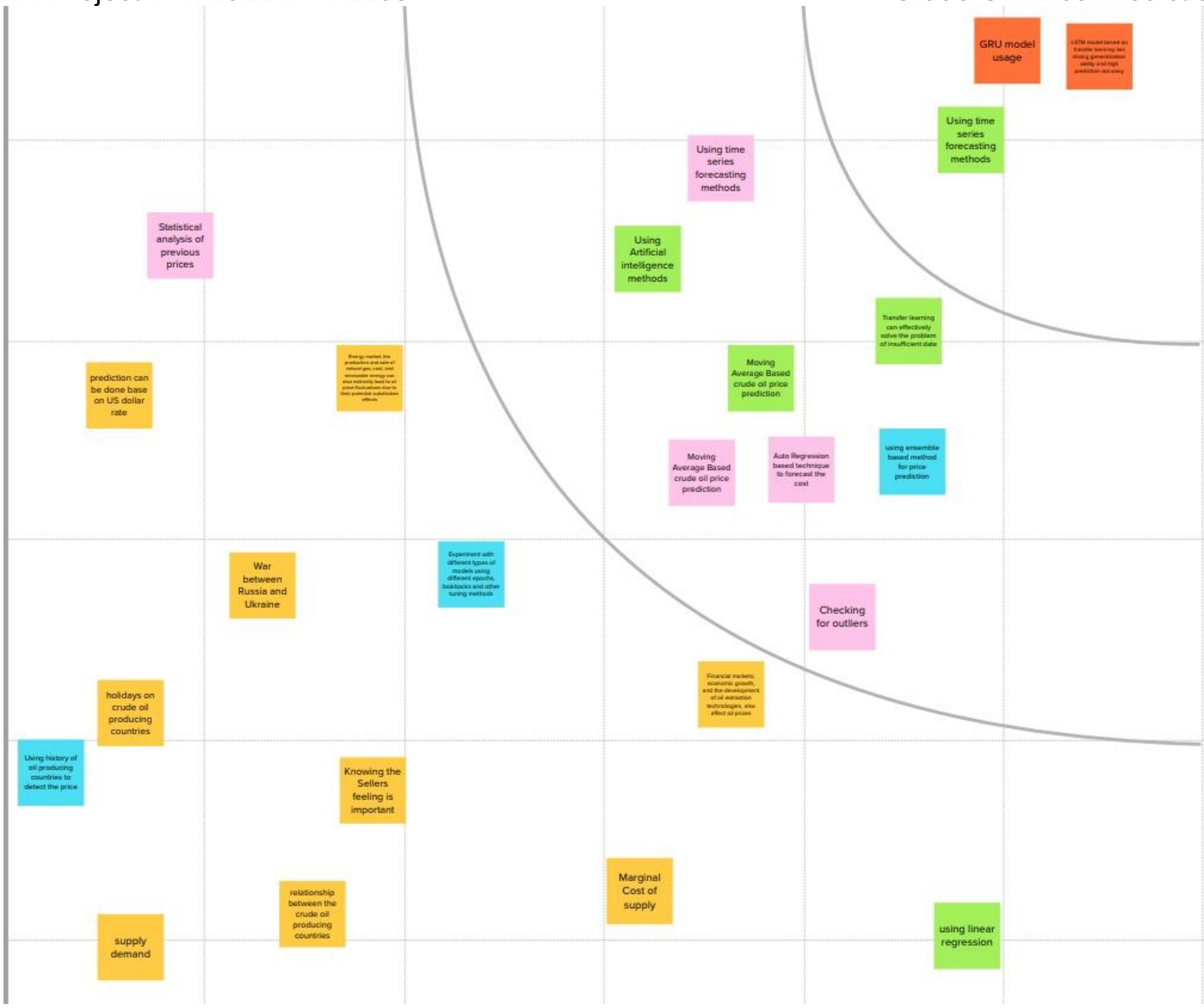


Figure 3.2 – Ideation & Brainstorming

3.3 Proposed Solution

S.No.	Parameter	Description
1	**Problem Statement (Problem to be solved)**	People working in the investment of crude oil needs an earlier crude oil price prediction system, which can help them to find the right time to buy crude oil so that they can increase profit from the purchase and reduce any substantial loss.
2	**Idea / Solution description**	This solution uses, deep learning models for the prediction of crude oil price. The deep learning models that are used for time series prediction are used in this solution. Two such models are LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Units). They can be used in combination to make the prediction.

3	**Novelty / Uniqueness**	The solution tries to combine the two best time series prediction models and predict the oil price with least error.
4	**Social Impact / Customer Satisfaction**	This application will help people working in the area of investment to take better decision regarding investing on the crude oil. This will help them to predict days to buy and sell crude oil, as the prices are often variable. The customer satisfaction depends on the accuracy of the solution. The better the closeness of the predicted prices of the crude oil to the original prices, more useful this solution becomes.
5	**Business Model (Revenue Model)**	The revenue model can be implemented as pay per month use model. The user can pay for the service for a month. Or it can also be provided by a yearly subscription
6	**Scalability of the Solution**	The solution frontend pages are built modularly. More pages can be added as we require. In backend flask can orchestrate the different functions, so in the future new functions and related pages can be added to the solution.

3.4 Problem Solution Fit

The problem solution fit is the solution one has found to address the problem of the customer. Figure 3.4 depicts the solution fit for the Crude oil Price Prediction System.

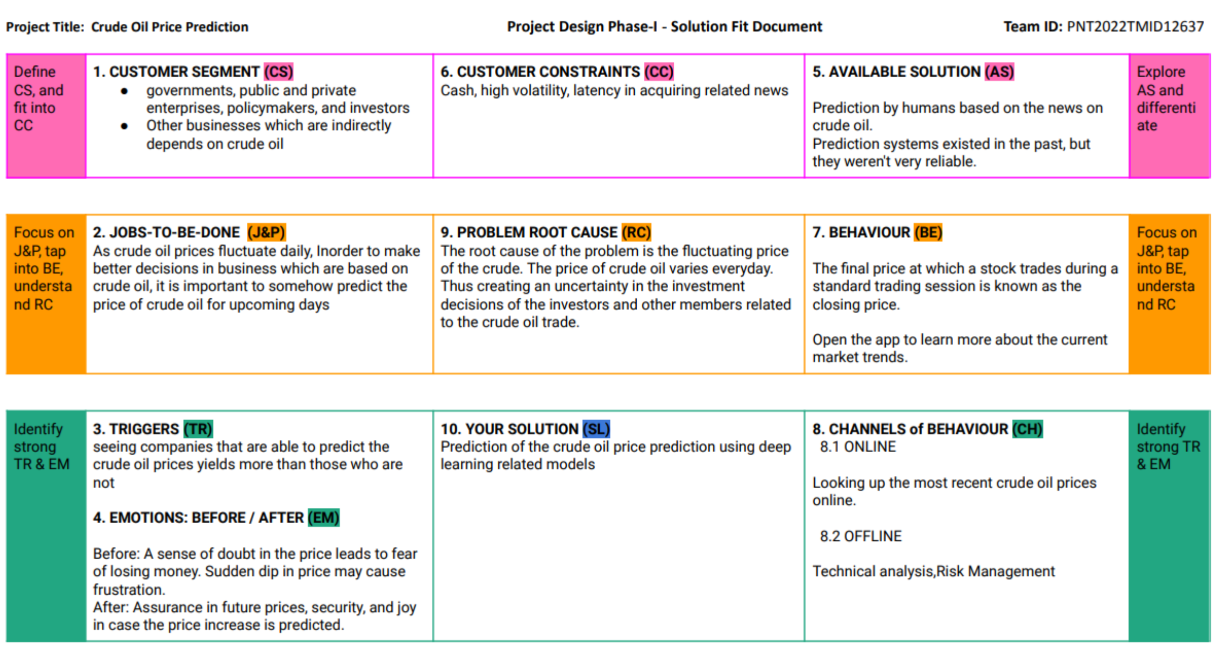


Figure 3.4 – Problem Solution Fit

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Functional Requirements

Table 4.1 are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	**Graph**	Showing Graph by obtaining the data from the Excel sheet.
FR-2	**News**	Information of all oil prices will be updated by the admin
FR-3	**Database**	Information of the crude oil price will be updated stored in excel sheet

Table 4.1 – Functional Requirements

4.2 Non-Functional Requirements

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	It can use by wide variety of client as it is very simple to learn and not complex to proceed.
NFR-2	**Security**	The information will be hashed so that it will be very secure to use.
NFR-3	**Reliability**	It will be reliable that it can update with very time period so that the accuracy will be good.
NFR-4	**Performance**	It will be performed fast and secure even at the lower bandwidth.
NFR-5	**Availability**	Prediction will be available for every user.
NFR-6	**Scalability**	we are going to use data in excel so it will be easily scalable.

Table 4.2 – Non-Functional Requirements

CHAPTER 5

PROJECT DESIGN

5.1 Dataflow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of how information flows within a system. A neat and clear DFD can thus depict the right amount of the system requirements graphically. It not only shows how data enters and leaves the system, but also what changes the information and where the data is stored. Figure 5.1 represents the DFD for the given project.

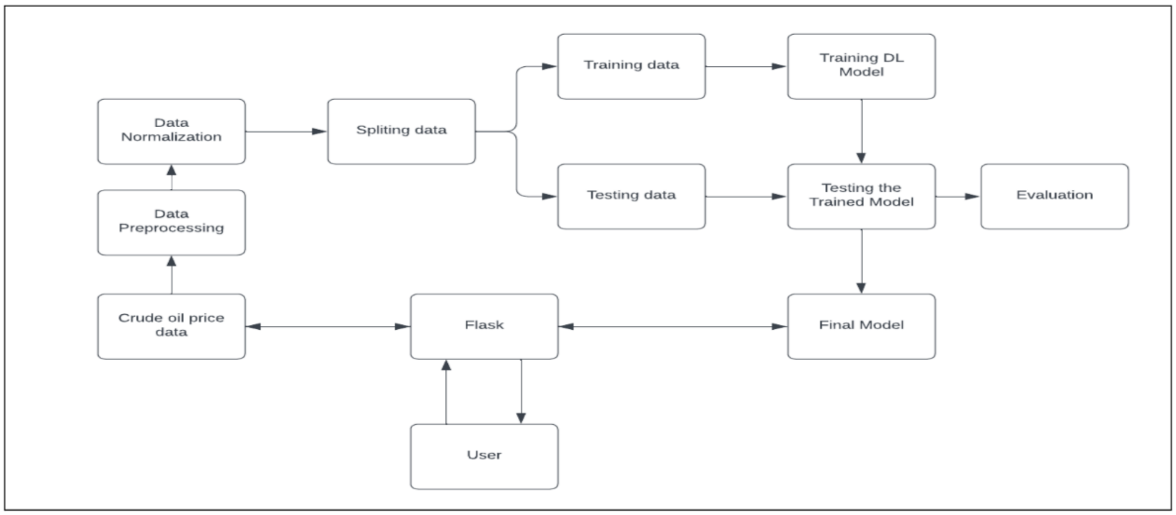


Figure 5.1 – Dataflow Diagram

5.2 Technical Architecture

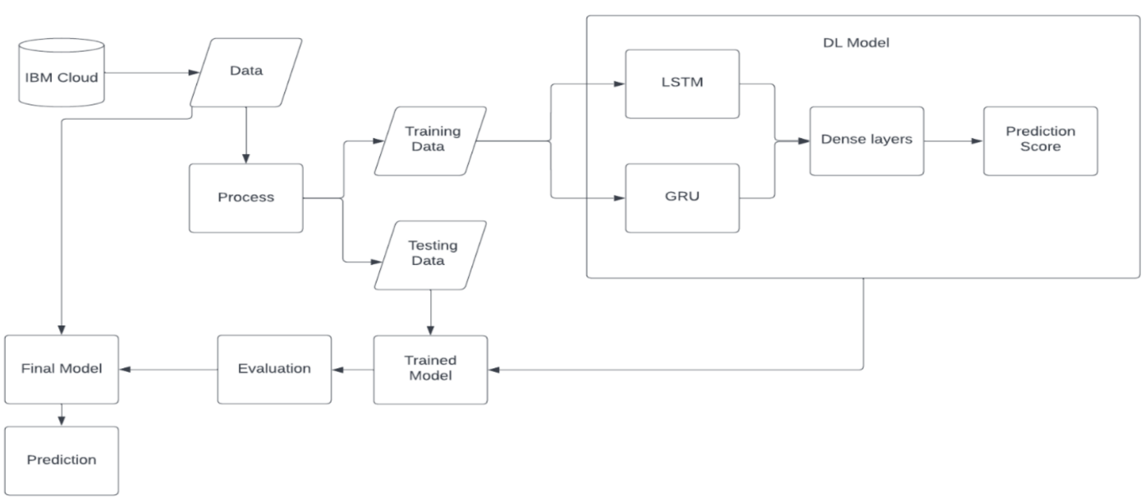


Figure 5.2 Technical Architecture

5.2.1 Component and Technologies

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Flask
2.	Loading data	Converting the csv file to python object	Python
3.	Pre-Processing of data	Pre-Processing and normalizing the data to getaccurate results	Python
4.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
5.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
6.	Machine Learning Model	Long short-term memory (LSTM) is an artificial neural network. Unlike standard feedforward neural networks, LSTM has feedback connections GRU Gated recurrent units is like a long short-termmemory (LSTM) with a forget gate, but has fewer parameters than LSTM, as it lacks an output gate.	Object Recognition Model, etc
7.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: 2.5Ghz processor, 8GB RAM Cloud Server Configuration: 4 GB GPU	Local, Cloud Foundry, Kubernetes, etc.

Table 5.2.1 – Components and Technologies

5.2.2 Application Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Flask
2.	Scalable Architecture	Justify the scalability of architecture (3 – tier,Micro-services)	Cloud Foundry, IBM Cloudant

3.	Availability	Justify the availability of application (e.g., use of load balancers, distributed servers etc.)	Cloud Foundry
4.	Performance	Design consideration for the performance of theapplication (number of requests per sec, use of Cache, use of CDN's) etc.	Cloud Foundry

Table 5.2.2 – Application Characteristics

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, and password, and confirming my password.	I can access my account/dashboard	High	Sprint-1
		USN-2	As a user, I will receive a confirmation email once I have registered for the application	I can receive a confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with FacebookLogin	Low	Sprint-2
		USN-4	As a user, I can note for the application through mail	I can register through already logged in mailaccount	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	After registration,I can login by only email & password	High	Sprint-1
	Dashboard	USN-1	As the web user,I can login simply by using account	Already created gmail can be used for Login.	Medium	Sprint-2
Customer (Web	Login	USN-1	As the web user,I can login simply	Already created gmail can	Medium	Sprint-2

user)			by using account.	be used for Login		
Customer Care Executive	Support		The Customer care service will provide solutions for any FAQ and also provide ChatBot.	I can solve the problemsarised by Support.	Low	Sprint-3
Administrator	News		Admin will give the recent news of Oil Prices	Provide the recent oil prices.	High	Sprint-4
	Notification		Admin will notify when the oil prices changes	Notification by Gmail	High	Sprint-4
	Access Control		Admin can control the access of users.	Access permission forUsers.	High	Sprint-4
	Database		Admin can store the details of users	Stores User details.	High	Sprint-4

Table 5.3 – User Stories

CHAPTER 6

PROJECT PLANNING AND 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	Download Crude Oil Price Dataset	2	Medium	All
Sprint-1	Data Preprocessing	USN-2	Importing The Dataset into Workspace	1	Low	All
Sprint-1		USN-3	Handling Missing Data	3	Medium	All
Sprint-1		USN-4	Feature Scaling	3	Low	All
Sprint-1		USN-5	Data Visualization	3	Medium	All
Sprint-1		USN-6	Splitting Data into Train and Test	4	High	All
Sprint-1		USN-7	Creating A Dataset with Sliding Windows	4	High	All
Sprint-2	Model Building	USN-8	Importing The Model Building Libraries	1	Medium	All
Sprint-2		USN-9	Initializing The Model	1	Medium	All
Sprint-2		USN-10	Adding LSTM Layers and GRU Layers	2	High	All
Sprint-2		USN-11	Adding Output Layers	3	Medium	All
Sprint-2		USN-12	Configure The Learning Process	4	High	All

Sprint	Functional Requirement (Epic)	User Story Number	User Story/Task	Story Points	Priority	Team Members
Sprint-2		USN-13	Train The Model	2	Medium	All
Sprint-2		USN-14	Model Evaluation	1	Medium	All
Sprint-2		USN-15	Save The Model	2	Medium	All
Sprint-2		USN-16	Test The Model	3	High	All
Sprint-3	Application Building	USN-17	Create An HTML File	4	Medium	All
Sprint-3		USN-18	Build Python Code	4	High	All

Sprint-3		USN-19	Run The App in Local Browser	4	Medium	All
Sprint-3		USN-20	Showcasing Prediction On UI	4	High	All
Sprint-4	Train The Model On IBM	USN-21	Register For IBM Cloud	4	Medium	All
Sprint-4		USN-22	Train The ML Model On IBM	8	High	All
Sprint-4		USN-23	Integrate Flask with Scoring End Point	8	High	All

Table 6.1 – Sprint Planning

6.2 Sprint Delivery Schedule

Sprint	Story Points	Duration (days)	Sprint Start Date	Story Points Completed	Sprint Release Date
Sprint 1	20	6	24 Oct 2022	20	29 Oct 2022
Sprint 2	20	6	31 Oct 2022	20	03 Nov 2022
Sprint 3	20	6	07 Nov 2022	20	10 Nov 2022
Sprint 4	20	6	14 Nov 2022	20	17 Nov 2022

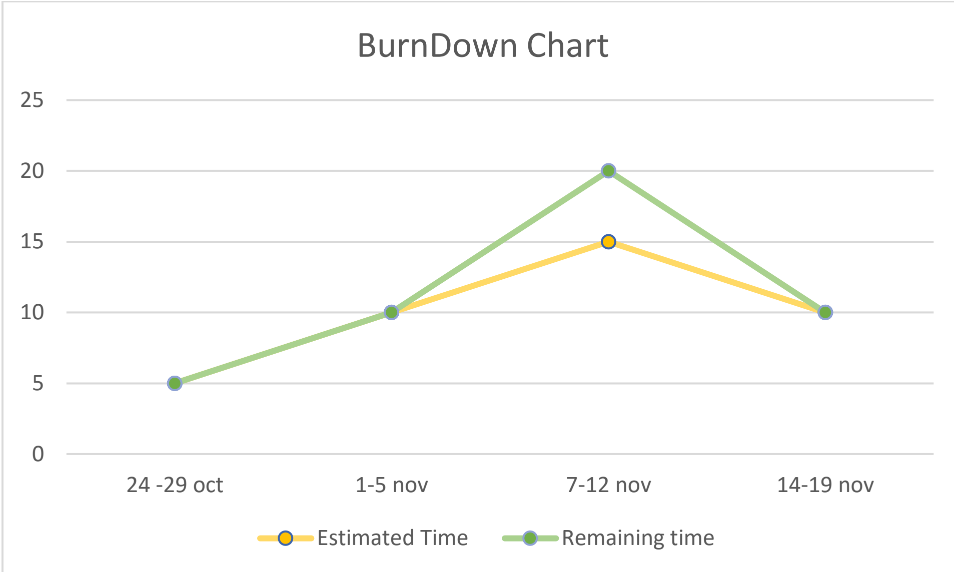
Table 6.2 – Sprint Delivery Schedule

6.3 Reports for JIRA

Velocity: Imagine we have a 10-day sprint duration, and the velocity of the team is20(points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (Story points per day)

$$AV = \frac{Sprint\ duration}{Velocity} = \frac{20}{6} = 3.33$$

Burndown Chart: A burndown chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



CHAPTER 7

CODING AND SOLUTION

7.1 Feature

FR No.	Feature	Description
FR-1	Crude oil Price Graph	Showing the price of crude oil for respective dates in a graph with dates in x axis and crude oil prices on y axis
FR-2	Current Price Prediction	Showing the last predicted price of the crude oil
FR-3	Prediction based on user provided values	When user provided with the three days prices of the crude oil the application will give predicted price for the next day

Table 7.1 – Description for Feature

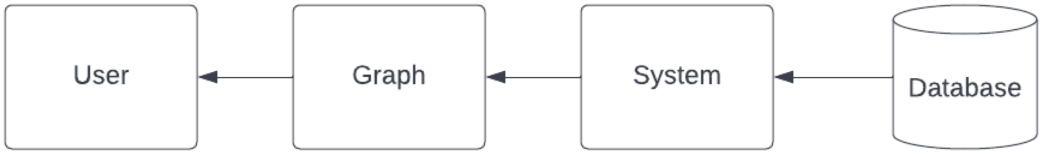


Figure 7.1 – Dataflow Diagram for Feature 1

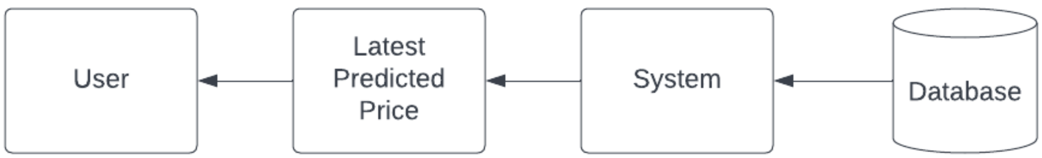


Figure 7.1 – Dataflow Diagram for Feature 2

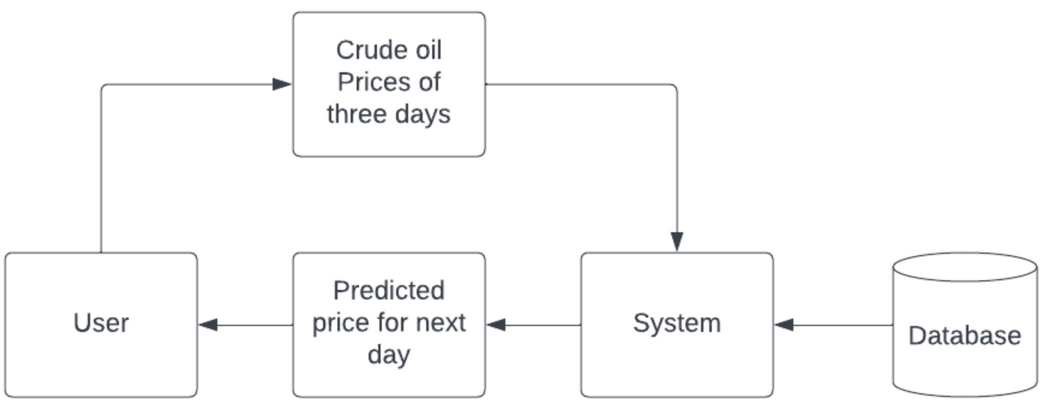


Figure 7.1 – Dataflow Diagram for Feature 3

CHAPTER 8

TESTING

8.1 Test Cases

The test cases are window of closing prices, where the window size is 3. The test cases are sent to the model and the prediction is compared with the original closing price. The loss metric is used to analyze the performance of the model. Figure 8.1 shows the result after the testing. The blue line in the bottom shows the true closing prices. The orange lines denote the prediction using the training data. The green line denotes the prediction based on testing data.

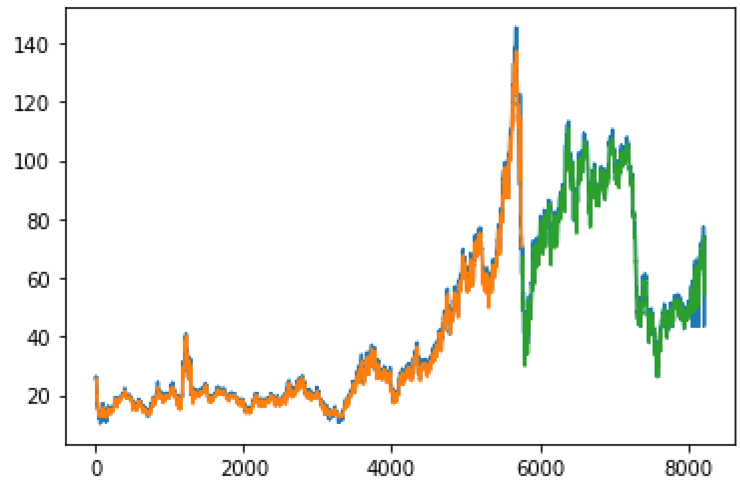


Figure 8.1 – Test Cases Run

CHAPTER 9

RESULTS

9.1 Performance Metrics

The performance metric used to measure the model is RMSE (Root Mean Square Error). RMSE is measured on both the prediction on training data and the testing data. Lower the RMSE score, better is the accuracy of the model. The results of the RMSE are given in the figure 9.1. From the figure 9.1 it can be observed that the RMSE are quite low. This indicates the model is working better and the predictions are quite accurate.

```
Train Score: 1.39 RMSE
Test Score: 2.37 RMSE
```

Figure 9.1 – Performance Metrics

CHAPTER 10

PROS AND CONS

10.1 Pros

- The application we have created is user friendly
- This application is flexible as user can choose the way they need to predict the price
- User can either give crude oil prices of any three continues dates or can get the latest predicted price of crude oil

10.2 Cons

As the data used in this project is not up to date the prices the model predicted will not be applicable for using in real world crude oil price prediction

CHAPTER 11

CONCLUSION

The prediction system works using the model that is built by combination of LSTM and GRU. The RMSE score for both the training and testing data is quite low. This shows that the accuracy of the model is good. A website is served using flask framework, which helps to enable the users to interact with the model. It helps the user to see the current predicted price the crude oil. And it helps to do prediction for manually entered crude oil closing price values.

CHAPTER 12

FUTURE WORKS

The model currently cannot update the prices to the current data automatically. Web automation can be enables to let the system update its database to current prices. And the model can be retrained on the updated data.

CHAPTER 13

APPENDIX

13.1 Source Code

App.py

```
from flask import Flask, render_template, url_for, flash, request, redirect
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
import requests

app = Flask(__name__)
app.config['SECRET KEY'] = "a5d6n3j4k5l6k7mn342nw3"

# NOTE: you must manually set API_KEY below using information retrieved from your
IBM Cloud account.
API_KEY = "AR9lZEewgN6dKPbjnLA46dB-sTUsO08rbIs_8BarVNXc"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' +
mltoken}

file = "dataset\\Crude Oil Prices Daily.xlsx"
df = pd.read_excel(file)
df["Closing Value"].fillna(df["Closing Value"].mean(), inplace=True)
x = df["Closing Value"].values.reshape(-1,1)

# normalising
scaler = MinMaxScaler(feature_range=(0,1))
x = scaler.fit_transform(x)
```

```
@app.route("/") #home route
def home():
    # dataset = [
    #     ('1',1),
    #     ('2',2),
    #     ('3',3),
    #     ('4',4),
    #     ('5',5)
    # ]
    # labels = [row[0] for row in dataset]
    # values = [row[1] for row in dataset]
    labels, values = getCrudeOilData(100)
    curr = getCrudeOilPriceCloud([values[-3],values[-2],values[-1]])
    return render_template("main_page.html", labels=labels, values=values,
current_price=curr)

@app.route("/predict", methods=["GET","POST"])
def predictPage():
    if request.method == "POST":
        day1 = request.form['day-1']
        day2 = request.form['day-2']
        day3 = request.form['day-3']
        if not day1 or not day2 or not day3:
            flash('Enter all the past 3 days value')
        else:
            day1, day2, day3 = float(day1), float(day2), float(day3)
            # price = getCurrentCrudeOilPrice([day1, day2, day3])
            price = getCrudeOilPriceCloud([day1, day2, day3])
            return render_template('prediction.html', price=price)
    return render_template('prediction.html')
    pass

def getCrudeOilData(n = 100):
    labels = list(df["Date"].astype(str))
    df["Closing Value"].fillna(df['Closing Value'].mean(), inplace=True)
    values = list(df["Closing Value"])
    return labels[len(labels)-n:], values[len(values)-n:] # returning only the
last n data

def getCrudeOilPriceCloud(prices=[]):
    data = [[prices]]

    payload_scoring = {"input_data": [{"fields": [["day-1","day-2","day-3"]],
"values": data}]}
```

```
    response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/8f848e93-fea8-40c6-a991-
43c14c6329e5/predictions?version=2022-11-16', json=payload_scoring,
     headers={'Authorization': 'Bearer ' + mltoken})
    response = response_scoring.json()['predictions'][0]["values"]
    res = scaler.inverse_transform(response)
    return round(res[0][0],4)


if __name__ == '__main__':
    app.run(debug=True)
```

13.2 GitHub & Project Demo Link

Github link: https://github.com/IBM-EPBL/IBM-Project-12679-1659457483

Project Demo Link: https://drive.google.com/file/d/14IRnVnVVUTtkxxm8JilmtiKw-i4Wow6g/view?usp=sharing