PROJECT NAME: **CRUDE OIL PRICE PREDICTION**

TEAM ID: **PNT2022TMID02708**

**Team:**

**Harish Sivaram – Team Leader**

**Abishek PL**

**Achyuth Pramod**

**Akash LP**

**ABSTRACT**

Oil prices are influenced by a variety of factors, particularly the decisions about output made by producers like the Organization of Petroleum Exporting Countries (OPEC), independent petro-states like Russia, and private oil-producing firms like ExxonMobil. Like any product, the laws of supply and demand influence prices.Natural disasters that could potentially disrupt production, and political unrest in oil-producing countries all impact pricing.Production costs influence prices, along with storage capacity. Although less impactful, the direction of interest rates can also influence the price of commodities. Economies are affected due to increased fluctuation in oil prices. Knowing the prices close to actual price might help tackling any crisis.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.**  **1** | **TITLE**  **ABSTRACT**  **LIST OF FIGURES**  **INTRODUCTION** | **PAGE NO**  **III**  **VI**  **1** |
|  | 1.1 Project Overview | 1 |
|  | 1.2 Purpose | 2 |
| **2** | **LITERATURE SURVEY** | **3** |
|  | 2.1 Existing problem | 3 |
|  | 2.2 References | 3 |
|  | 2.3 Problem Statement Definition | 5 |
| **3** | **IDEATION & PROPOSED SOLUTION** | **6** |
|  | 3.1 Empathy Map Canvas | 6 |
|  | 3.2 Ideation & Brainstorming | 7 |
|  | 3.3 Proposed Solution | 8 |
|  | 3.4 Problem Solution fit | 9 |
| **4** | **REQUIREMENT ANALYSIS** | **10** |
|  | 4.1 Functional requirement | 10 |
|  | 4.2 Non-Functional requirements | 11 |
| **5** | **PROJECT DESIGN** | **12** |
|  | 5.1 Data Flow Diagrams | 12 |
|  | 5.2 Solution & Technical Architecture | 13 |
|  | 5.3 User Stories | 14 |
| **6** | **PROJECT PLANNING & SCHEDULING** | **16** |
|  | 6.1 Sprint Planning & Estimation | 16 |
|  | 6.2 Sprint Delivery Schedule | 17 |
|  | 6.3 Reports from JIRA | 18 |
| **7** | **CODING & SOLUTIONING** | **19** |
|  | 7.1 Feature 1 | 19 |
|  | 7.2 Feature 2 | 26 |
|  | 7.3 Feature 3 | 32 |
| **8** | **TESTING** | **58** |
|  | 8.1 Test Cases | 58 |
|  | 8.2 User Acceptance Testing | 59 |
| **9** | **RESULTS** | **61** |
|  | 9.1 Performance Metrics | 61 |
| **10** | **ADVANTAGES & DISADVANTAGES** | **62** |
| **11** | **CONCLUSION** | **63** |
| **12** | **APPENDIX** | **65** |
|  | Source Code | 65 |
|  | GitHub & Project Demo Link | 105 |

**INTRODUCTION**

* 1. **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. Monitoring the crude oil prices and regulating is essential for the economy of a country. Take the case of Sri Lanka, we have all seen how the country’s economy is in shambles today and while crude oil is not a dominant factor for the same, it is nonetheless a contributor. The rising crude oil prices have a trickling effect on the prices of every other commodity in the market as the transport costs for the suppliers increase and therefore, it is a large contributor to the rising inflation.

Similarly for any economy, control of the crude oil prices is important to control [inflation](https://www.fisdom.com/how-does-inflation-impact-the-stock-market/). In India, when the fuel prices had gone beyond Rs. 100 a liter, it had created a huge impact on the daily lives of the citizens and was the cause of major protests and general discontent. The government to counter this reduced the indirect taxes which had created a huge dent in its revenue but was necessary to control the rising inflation that will ultimately hurt the pockets of the common citizen. 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.

* 1. **PURPOSE**

Since crude oil prices are constantly changing and are typically more volatile than stock or currency prices, it is crucial for successful investors and traders to have good information sources that report on the many factors that can influence oil prices.To an investor, crude oil can be a speculative asset, a portfolio diversifier, or a hedge against related positions.

As the linkage between the markets, the uncertainty of the world economy and energy, the influence factors of oil price have become complex. It is difficult to point out which factors have the dominant effect on the oil price. If all possible oil price factors are added into the existing forecast model, it may lead to over-fitting problems, which will affect the forecast results. How to forecast crude oil prices in a new and effective method is one problem that academics and practitioners are very concerned about all the time. It can provide reference and theoretical support for the formulation of national energy security strategy and enterprise avoidance of market risks. To better analyze the changing trend of the crude oil market, it is necessary to determine the price, determine the impact of each factor on price, and establish a forecasting model finally.

Systematic analysis of the characteristics of complex international oil markets and accurate capture of the new trend in international oil prices are critical.

**LITERATURE SURVEY**

**2.1 EXISTING PROBLEM**

Oil prices are heavily influenced by traders who bid on oil futures contracts in the commodities market based on their perceptions of the future supply and demand for oil. Futures contracts and oil derivatives are traded daily, which acts to influence the price of oil. This causes the price of oil to change daily because it all depends on how trading went that day.

Traders base their bids on their perceptions of supply and demand. Other entities, such as governments and the Organization of the Petroleum Exporting Countries (OPEC), can affect the traders' bidding decisions by influencing trade or adjusting the amount of oil produced and stored.

Oil is commonly referred to as the most volatile of commodities. If you are considering trading in oil or oil derivatives, it helps to understand what factors drive the price of oil and how traders, governments, and consumers influence it.

* Traders heavily influence oil prices through bids on futures contracts
* Bids are based on perceptions of current and future global supply and demand
* Human-made and natural crises have huge impacts on oil prices

## Traders are major oil price influencers where oil futures contracts are executed on the floor of a commodity exchange, where only commodities are traded. Due to fluctuations in the prices of oil, the effect of it in the economy is unavoidable, therefore knowing the oil prices before hand helps the government or private entities to prepare for any major crisis safeguarding the economy.

**2.2 REFERENCES**

[1]<https://energyinformatics.springeropen.com/articles/10.1186/s42162-021-00166-4>

1. <https://www.investopedia.com/terms/c/crude-oil.asp>
2. [https://www.digitalrefining.com/article/1002330/forecasting-crude-oil-prices#:~:text=Crude%20oil%20price%20fluctuations%20have,enterprises%2C%20policymakers%2C%20and%20investors](https://www.digitalrefining.com/article/1002330/forecasting-crude-oil-prices#:~:text=Crude oil price fluctuations have,enterprises, policymakers, and investors)
3. [https://www.fisdom.com/what-factors-determine-international-crude-oil-prices/#:~:text=Crude%20oil%20is%20one%20of,must%20be%20kept%20in%20check](https://www.fisdom.com/what-factors-determine-international-crude-oil-prices/#:~:text=Crude oil is one of,must be kept in check).
4. <https://www.thebalancemoney.com/how-are-oil-prices-determined-3305650>

[6]<https://www.eia.gov/finance/markets/crudeoil/spot_prices.php>

**2.3 PROBLEM STATEMENT DEFINITION**

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.

**IDEATION & PROPOSED SOLUTION**

**3.1 EMPATHY MAP CANVAS**

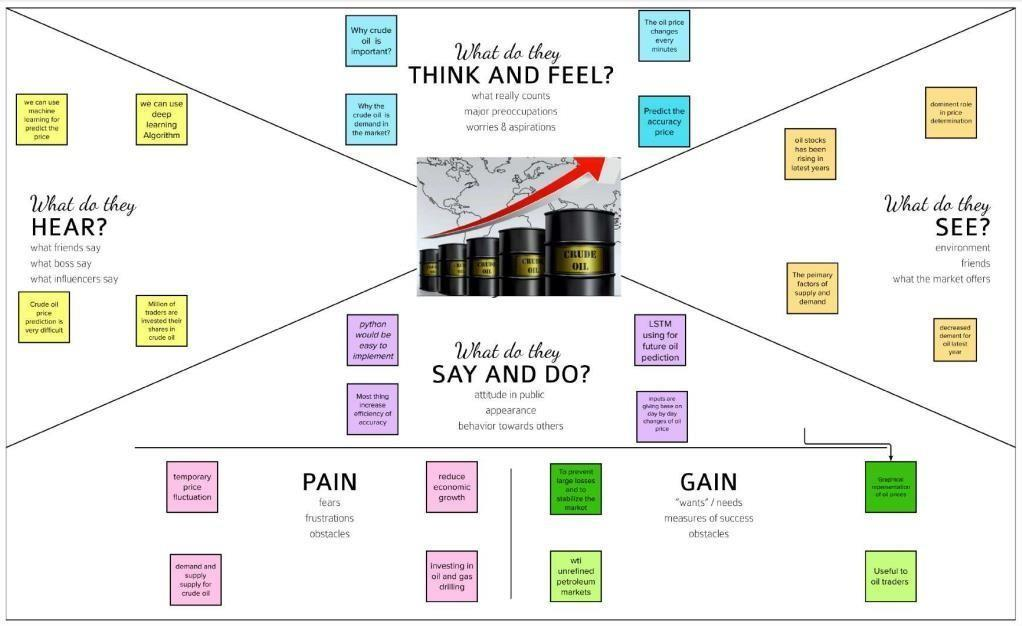
An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s behaviours and attitudes.

It is a useful tool to helps teams better understand their users.

Creating an effective solution requires understanding the true problem and the person who is experiencing it.

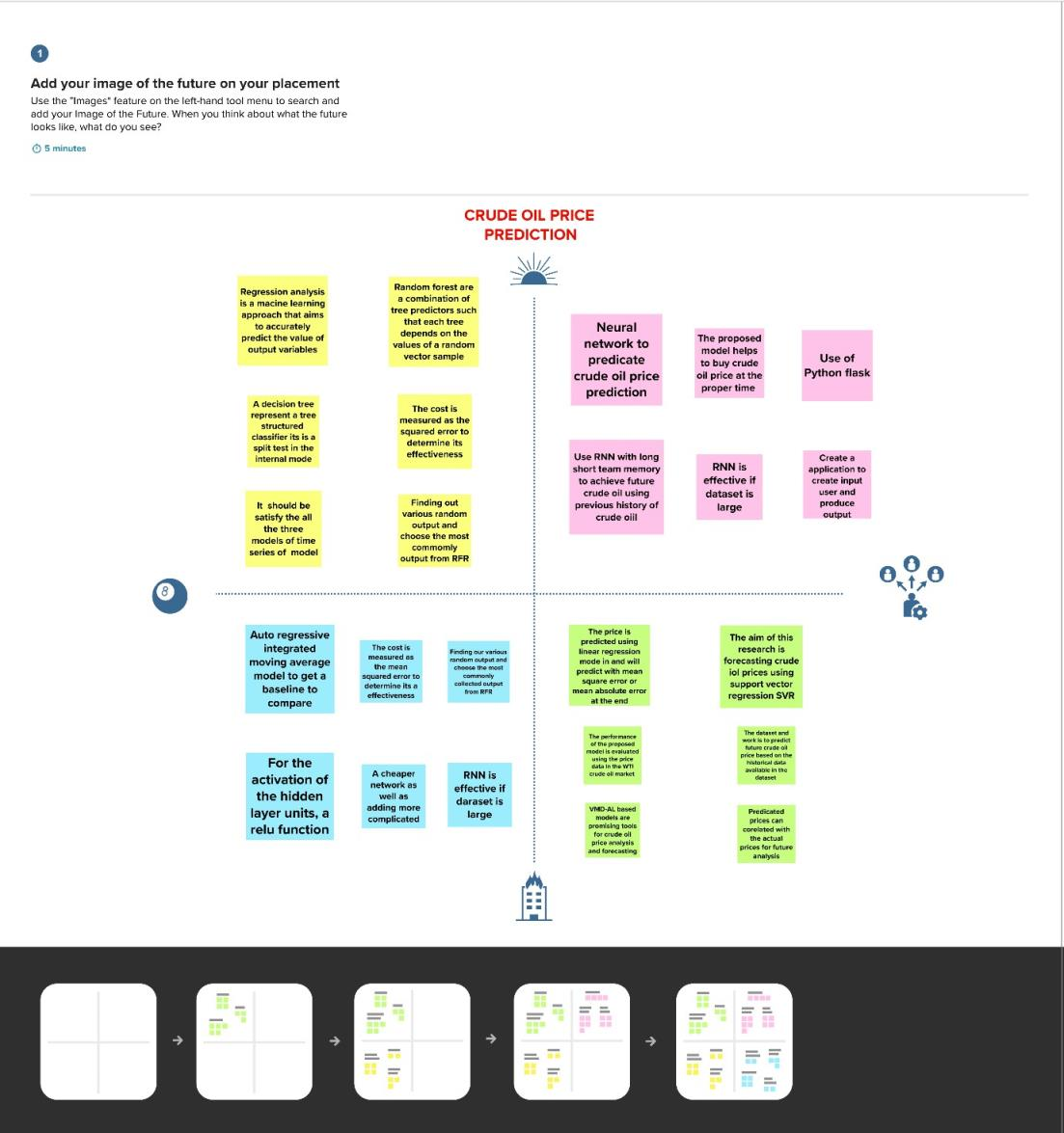
The exercise of creating the map helps participants consider things from the user’s perspective along with his or her goals and challenges.

Empathy Map for Crude Oil Price Prediction :

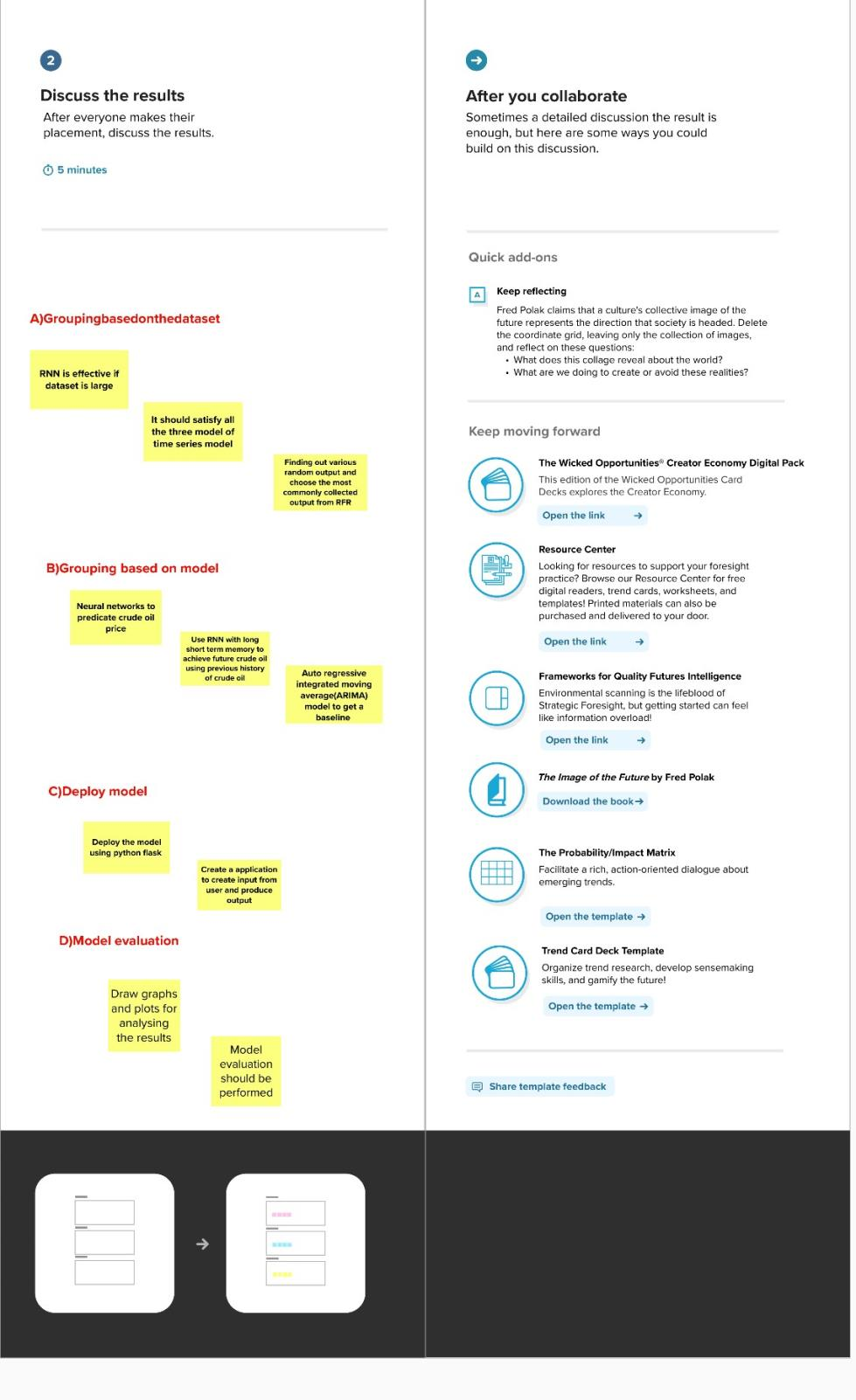


**Fig 1.1.Empathy Map**

**3.2 IDEATION & BRAINSTORMING**



**Fig 1.2.Problem Statement & Brainstorm**



**Fig 1.3.Group Ideas & Prioritize**

**3.3 PROPOSED SOLUTION**

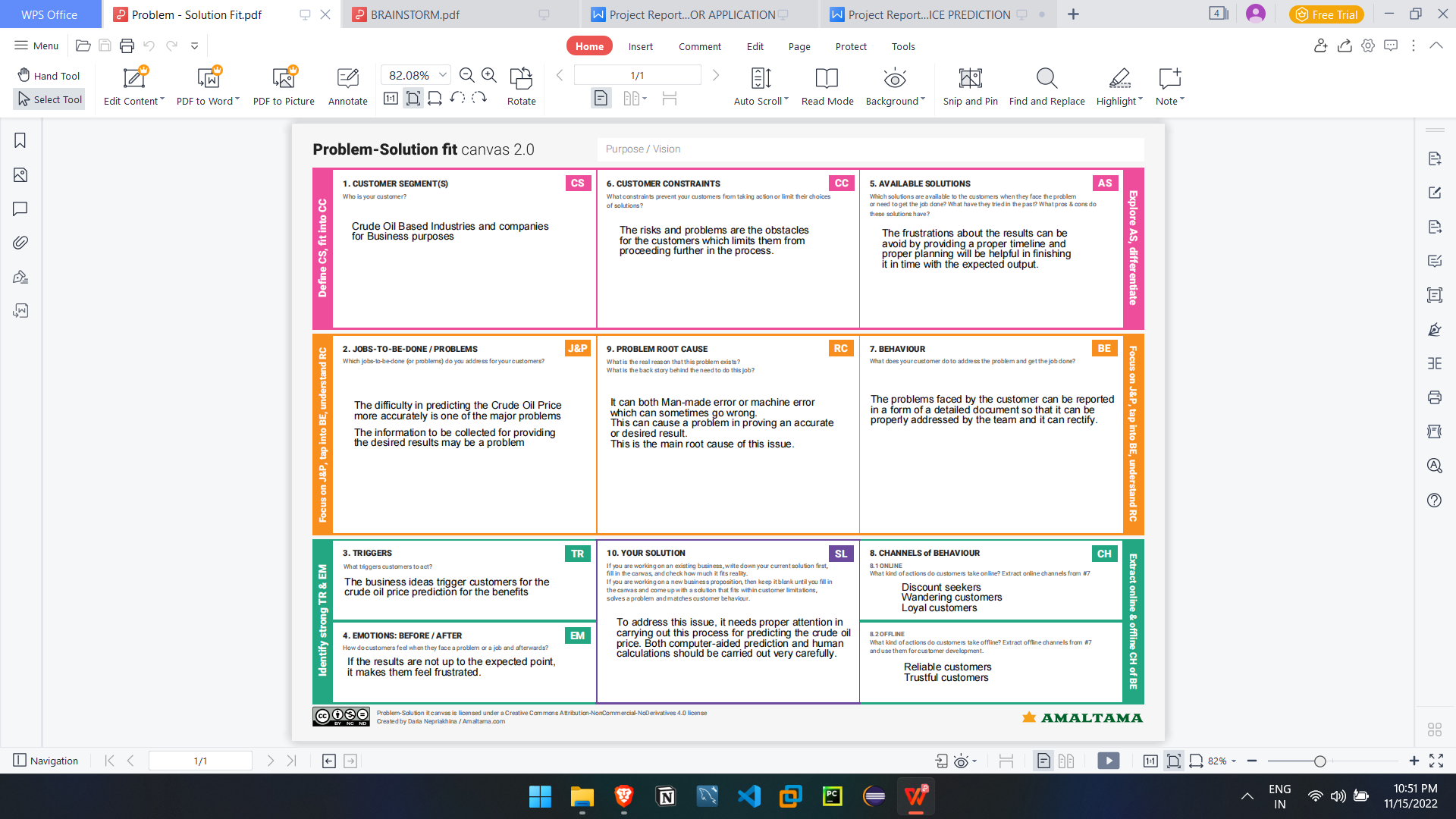
Collection of the dataset of the past oil prices with time (past 30 years)so that by feeding those to the model and training it and compiling it and

when it achieves  the optimal state it  can be implemented  in a web application.

This traditional idea will have a better effect with results due to implementation of periodic training. By using the web app, customers can gain knowledge of the crude oil price and get benefits financially. The proposed idea takes the input from the user and adjusts and trains through the datasets so that it will adapt to very different situations and predict prices.

The model is a Recurrent neural network built using Tensorflow Keras library with sequential method . Time series analysis is a key component in training the model with the dataset.

**3.4 PROBLEM SOLUTION FIT**

****

**Fig 1.4.Problem Solution fit**

**REQUIREMENT ANALYSIS**

**4.1 FUNCTIONAL REQUIREMENT**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR**  **No.** | **Functional**  **Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Registration through Form,Gmail, LinkedIn |
| FR-2 | User Confirmation | Confirmation via Email and OTP |
| FR-3 | Graph | Displaying graph from obtained dataset |
| FR-4 | Support | Providing answers for the queries asked by users |
| FR-5 | News | Information of the oil prices will be updated by admin |
| FR-6 | Notification | Notification will be sent for the users price alert |
| FR-7 | Database | Information of the User will be stored |

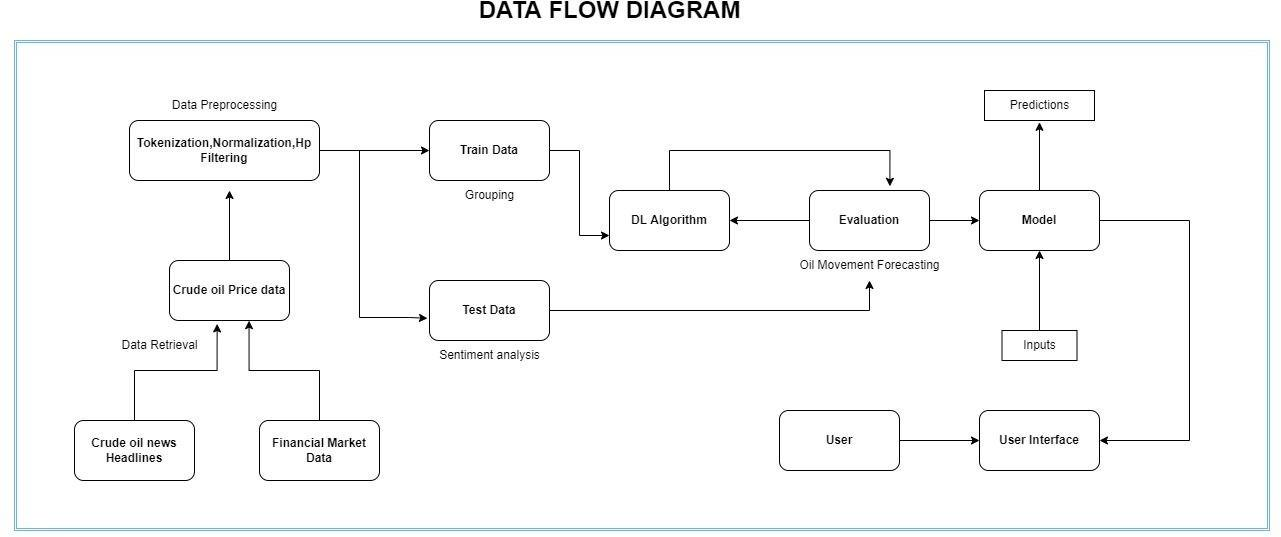
**4.2 NON-FUNCTIONAL REQUIREMENTS**

Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **NFR**  **No.** | **Non Functional**  **Requirement (Epic)** | **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 | We are using login for the user and 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 perform fast and secure even at the lower  bandwidth. |
| NFR-5 | Availability | Prediction will be available for every user but only  for premium user news,database and price alert  will be alert. |
| NFR-6 | Scalability | It is scalable that we are going to use data in kb so  that the quite amount of storage is satisfied |

**PROJECT DESIGN**

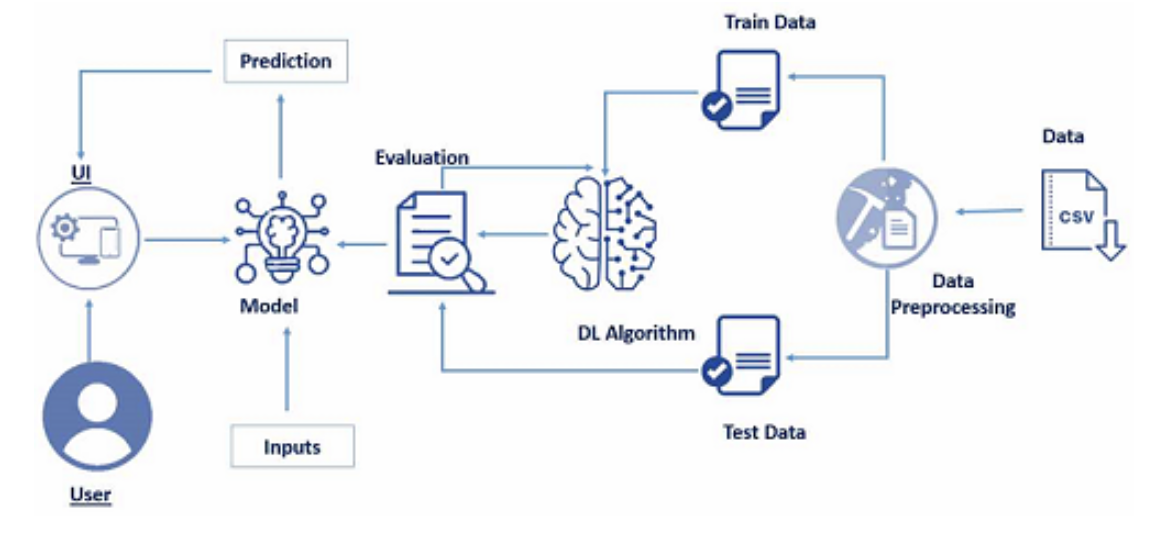
**5.1 DATA FLOW DIAGRAMS**



**Fig 2.1.Data Flow Diagram**

**5.2 SOLUTION & TECHNICAL ARCHITECTURE**

An application should be developed which would take the user credentials, store them for login purposes. A user friendly and responsive interface with a a simple input box. When the user requests for price, the application takes the date as input, the trained model will display the predicted price of the oil on that date.



**Fig 2.2. Architecture diagram**

**5.3 USER STORIES**

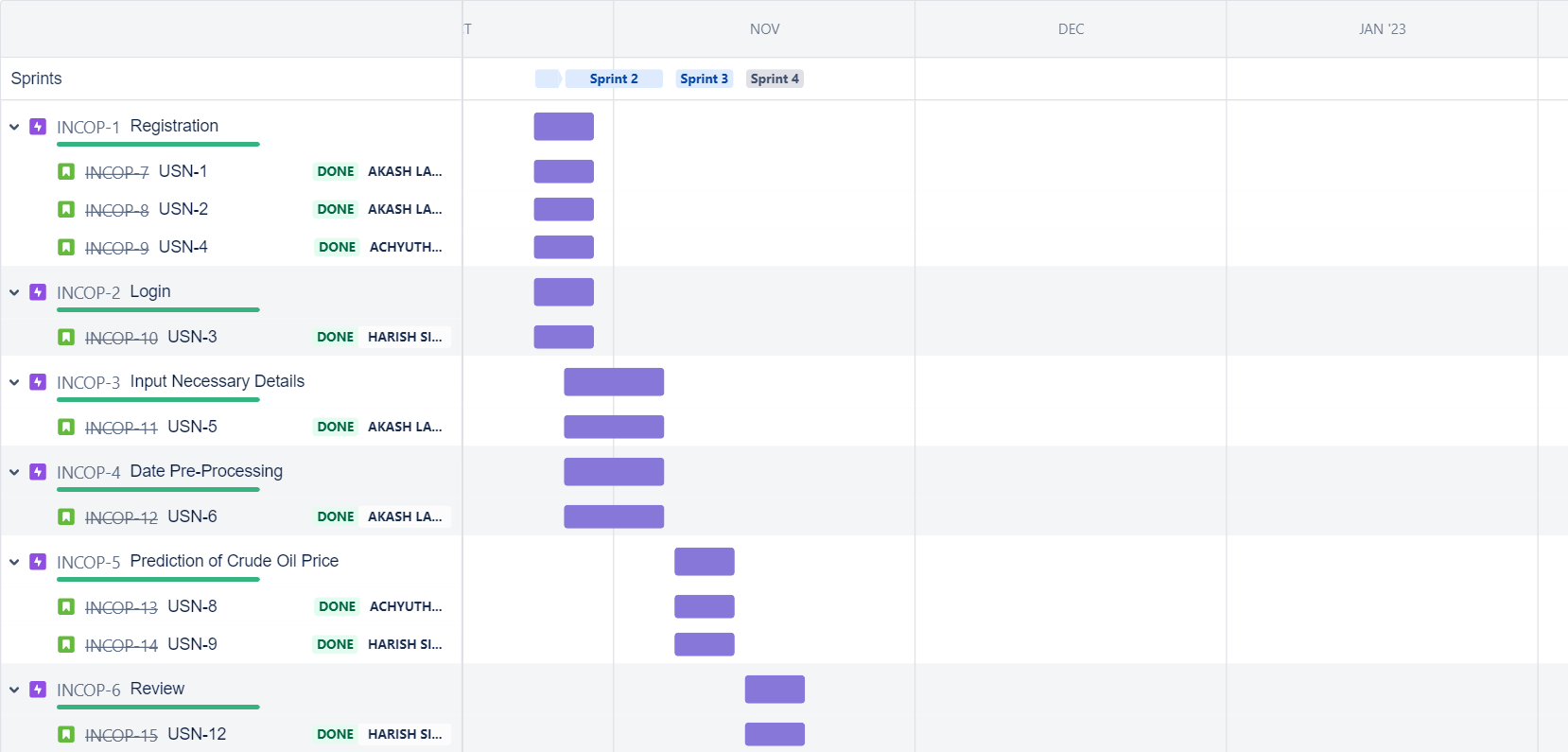
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| Customer (Mobile user) | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | I can access my account / dashboard | High | Sprint-1 |
|  |  | USN-2 | As a user, I will receive confirmation email once I have registered for the application | I can receive confirmation email & click confirm | High | Sprint-1 |
|  |  | USN-3 | As a user, I can register for the application through Gmail | I can receive confirmation notifications through Gmail | Medium | Sprint-1 |
|  |  | USN-4 | As a user, I can register for the application  through Gmail | I can register through  already  logged in gmail account. | High | Sprint-2 |
|  | Login | USN-5 | As a user, I can log into the application by  entering email & password | After registration,I can log in  by only email & password | High | Sprint-2 |
|  | Line\Bar Graph | USN-6 | After entering the inputs,the model will display  predictions in Line\Bar Graph Format | I can get the expected  prediction in various  formats. | High | Sprint-3 |
| Customer (Web  user) | Login | USN-7 | As the web user,I can login simply by using  Gmail or Facebook account | Already created gmail can  be used for Login. | High | Sprint-3 |
| Customer Care  Executive | Support | USN-8 | The Customer care service will provide solutions  for any FAQ and also provide ChatBot | I can solve the problems  arised by Support. | Low | Sprint-3 |
| Administrator | News | USN-9 | Admin will give the recent news of Oil Prices | Provide the recent oil prices  the appearance and navigation in a user friendly manner | High | Sprint-4 |
|  | Notification | USN-10 | Admin will notify when the oil prices changes | Notification by Gmail. | High | Sprint-4 |
|  | Access Control | USN-11 | Admin can control the access of users | Access permission for  Users | High | Sprint-4 |
|  | Database | USN-12 | Admin can store the details of users. | Stores User details | High | Sprint-4 |

**PROJECT PLANNING & SCHEDULING**

**6.1 SPRINT PLANNING & ESTIMATION**

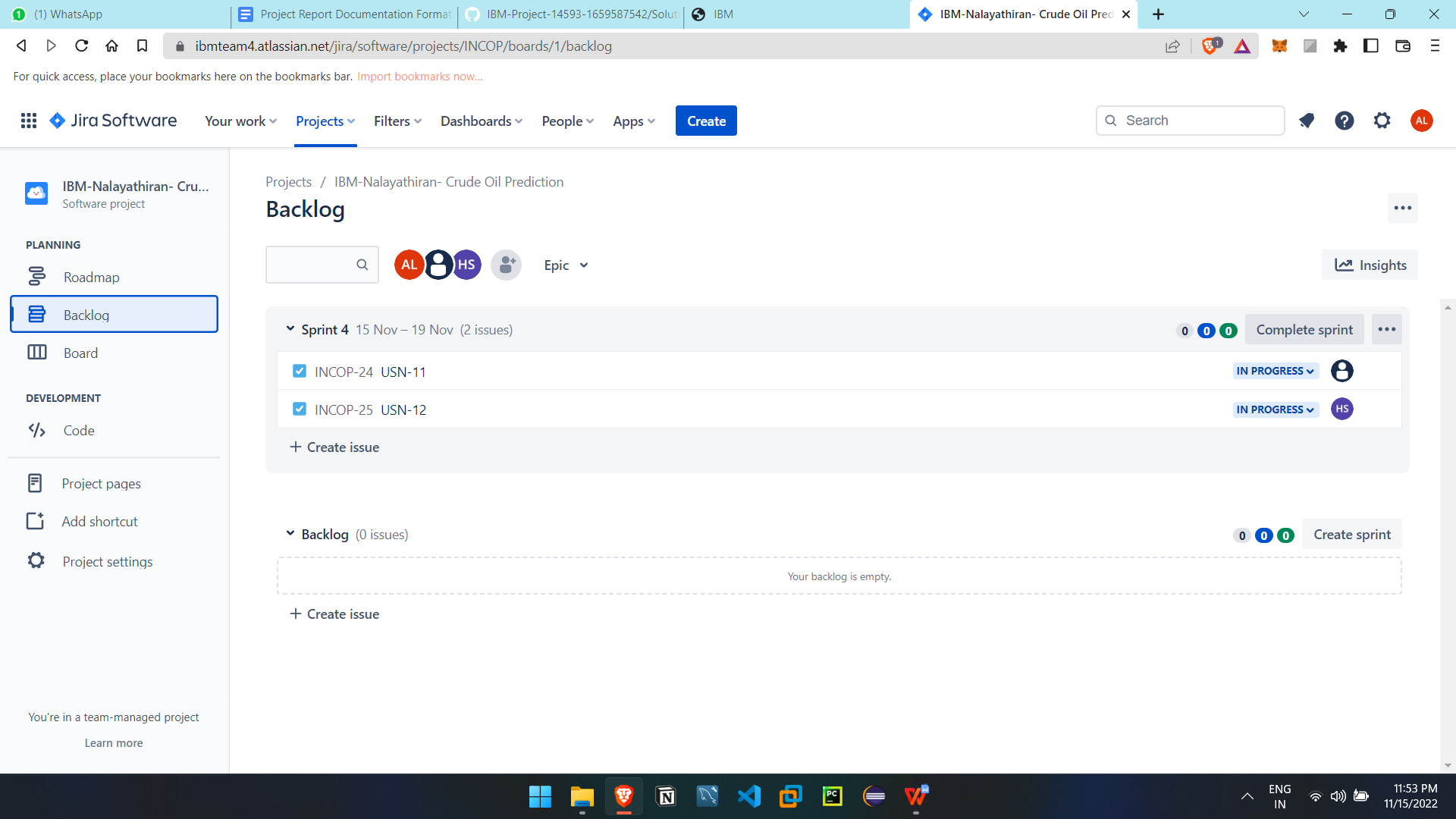
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Registration | USN-1 | As a user, I can register for the application by  entering my email, password, and confirming my password. | 2 | High | Akash Lakshmipathy |
| Sprint-1 |  | USN-2 | As a user, I will receive confirmation email once I have registered for the application | 2 | High | Akash Lakshmipathy |
| Sprint-1 | Login | USN-3 | As a user, I can log into the application by entering email & password | 2 | Medium | Harish Sivaram |
| Sprint- 2 |  | USN-4 | As a user, I can register for the application  through Gmail | 6 | High | Achyuth Pramod |
| Sprint- 2 | Login | USN-5 | As a user, I can log into the application by  entering email & password | 4 | High | Akash Lakshmipathy |
| Sprint-2 | Line/Bar | USN-6 | After entering the inputs,the model will display  predictions in Line\Bar Graph Format. | 2 | High | Akash Lakshmipathy |
| Sprint- 3 | Login | USN-7 | As the web user,I can login simply by using  Gmail or Facebook account. | 4 | High | Harish Sivaram |
| Sprint-3 | Support | USN-8 | The Customer care service will provide solutions  for any FAQ and also provide ChatBot. | 2 | Low | Achyuth Pramod |
| Sprint-3 | News | USN-9 | Admin will give the recent news of Oil Prices | 2 | High | Harish Sivaram |
| Sprint-3 | Notification | USN-10 | Admin will notify when the oil prices changes | 4 | High | Abishek P |
| Sprint-4 | Access Control | USN-11 | Admin can control the access of users | 2 | High | Abishek P |
| Sprint-4 | Database | USN-12 | Admin can store the details of users. | 2 | High | Harish Sivaram |

**6.2 SPRINT DELIVERY SCHEDULE**

****

**Fig 3.1. Sprint Delivery Schedule**

**6.3 REPORTS FROM JIRA**

****

**Fig 3.2. Sprint report**

**CODING & SOLUTIONING**

**7.1 FEATURE 1 – SIGN UP**

**Graphical user interface

Description automatically generated**

**7.2 FEATURE 2 – SIGN IN**

**Graphical user interface, website

Description automatically generated**

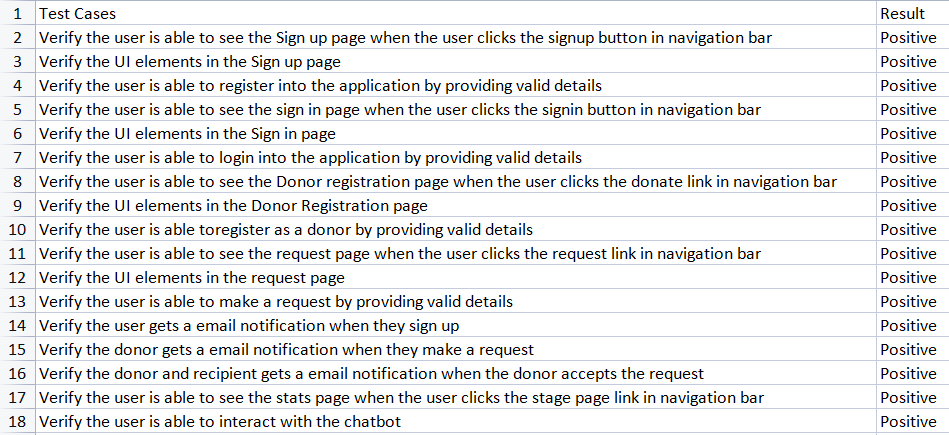
**7.2 FEATURE 3 – MODEL**

**Graphical user interface

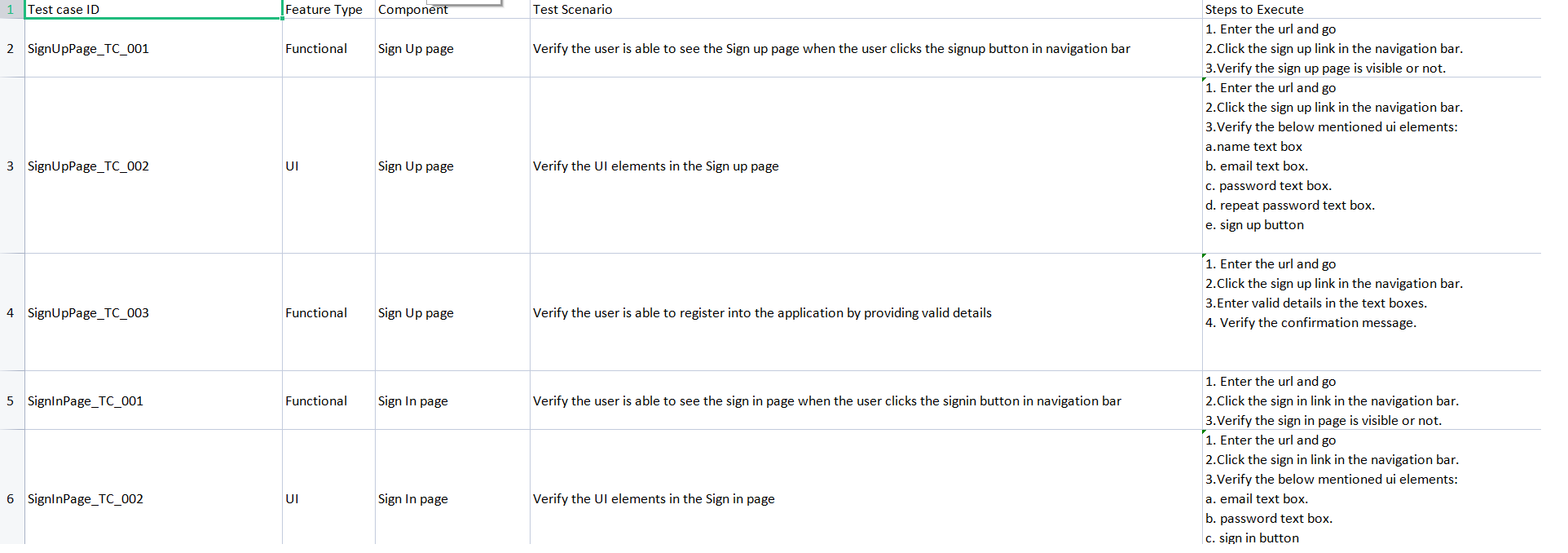
Description automatically generated**

**TESTING**

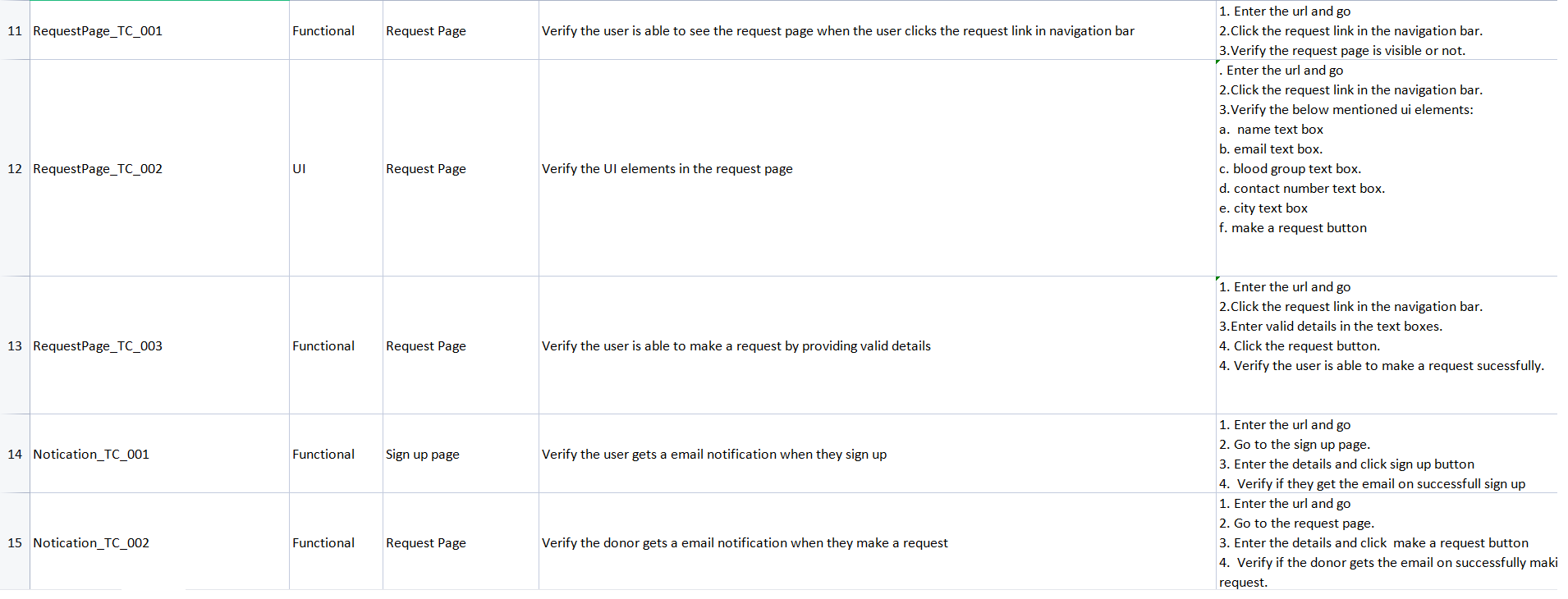
**8.1 TEST CASES**

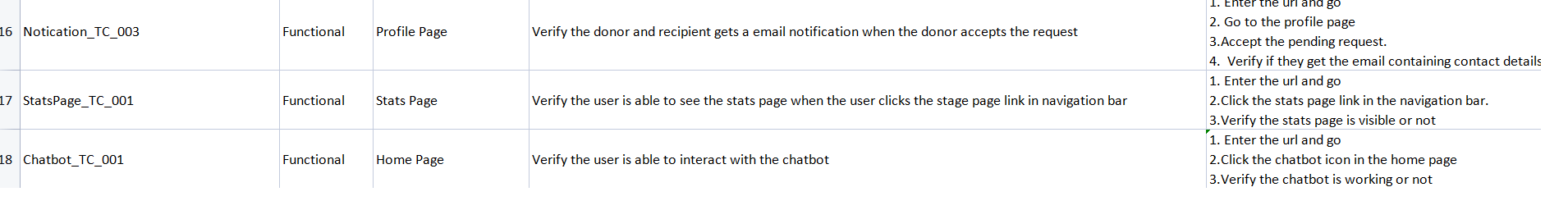
****

**8.2 USER ACCEPTANCE TESTING**

****

****

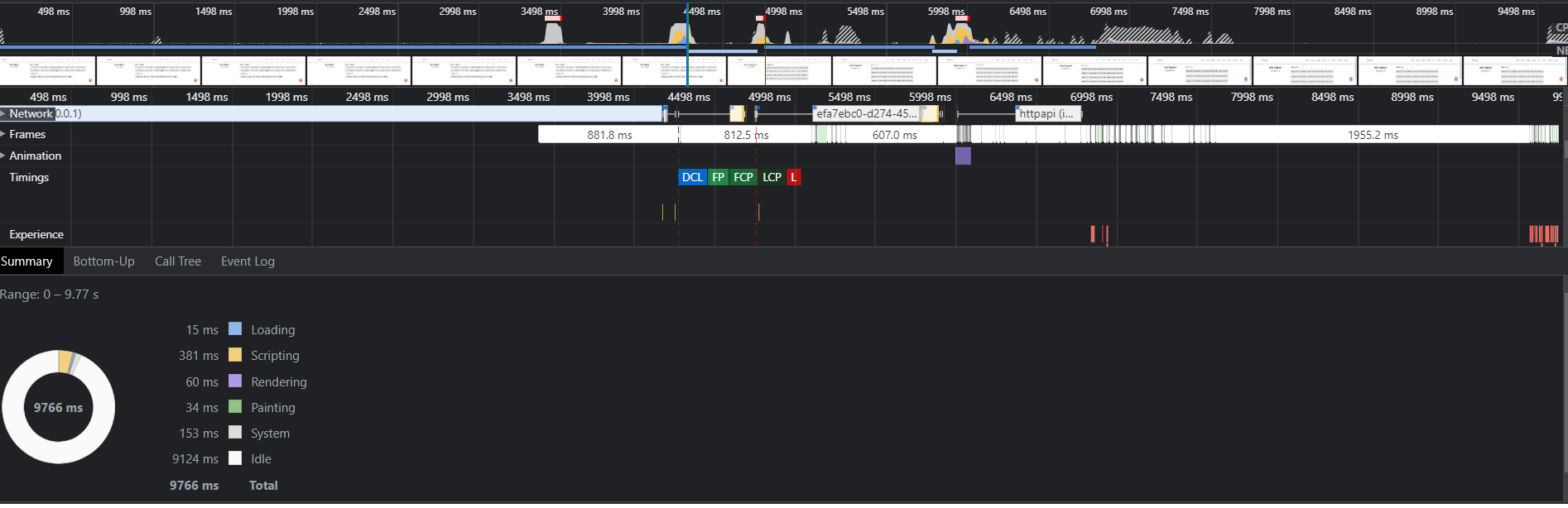
****

****

**RESULTS**

**9.1 PERFORMANCE METRICS**

Web application performance metrics help determine certain aspects that impact the performance of an application. There are eight key metrics, including: User Satisfaction—also known as Apdex Scores, uses a mathematical formula in order to determine user satisfaction.

****

**Fig 4.1. Performance Metrics**

**ADVANTAGES & DISADVANTAGES**

**ADVANTAGES**

* It is a user-friendly application.
* It will help user analyze the price of crude oil.
* Helps big entities and governments prepare for major crisis due to oil.
* Price fluctuations can be managed easily by traders and businessman.
* Helps stabilize economy.

**DISADVANTAGES**

* It requires an active internet connection.
* It does not consider global news as a factor in consideration.
* Uses only historical prices as predicting factor
* Prices may widely vary in case of sudden price changes due to natural or man- made disasters.

**CONCLUSION**

Crude oil is a major commodity traded all around the world. It is a commodity which involves a lot of Geo-politics. Lot of economies rely on the crude oil market.

We have seen a lot of downfall and uprise of powerful economies. The main driving factor of economy is the crude oil. Prices of crude oil vary everyday everyhour globally. Due to sudden spike or large fluctuations in price of crude oil, it increases the transportation costs and increasing the prices of all other commodities thus increasing inflation. Knowing the prices of crude oil before helps the governments to prepare for such crisis and avoid Srilanka like situatuins to happen to their respective countries. It also helps traders to book more profits. Thus crude oil drives the world, knowing the prices before hands enables us to drive the crude oil.

**APPENDIX**

**SOURCE CODE:**

import numpy as np # linear algebra

import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)

import datetime

from pylab import rcParams

import matplotlib.pyplot as plt

import warnings

import itertools

import statsmodels.api as sm

from keras.models import Sequential

from keras.layers import Dense

from keras.layers import LSTM

from keras.layers import Dropout

from sklearn.metrics import mean\_squared\_error

from keras.callbacks import ReduceLROnPlateau, EarlyStopping, ModelCheckpoint

from sklearn.metrics import mean\_squared\_error

from sklearn.metrics import mean\_absolute\_error

import seaborn as sns

sns.set\_context("paper", font\_scale=1.3)

sns.set\_style('white')

import math

from sklearn.preprocessing import MinMaxScaler

# Input data files are available in the "../input/" directory.

# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

warnings.filterwarnings("ignore")

plt.style.use('fivethirtyeight')

import os

for dirname, \_, filenames in os.walk('/kaggle/input'):

for filename in filenames:

print(os.path.join(dirname, filename))

dateparse = lambda x: pd.datetime.strptime(x, '%b %d, %Y')

#Read csv file

from google.colab import files

uploaded = files.upload()

import io

df = pd.read\_excel(io.BytesIO(uploaded['Crude Oil Prices Daily.xlsx']))

df.head()

df[:10]

|  | **Date** | **Closing Value** |
| --- | --- | --- |
| **0** | 1986-01-02 | 25.56 |
| **1** | 1986-01-03 | 26.00 |
| **2** | 1986-01-06 | 26.53 |
| **3** | 1986-01-07 | 25.85 |
| **4** | 1986-01-08 | 25.87 |
| **5** | 1986-01-09 | 26.03 |
| **6** | 1986-01-10 | 25.65 |
| **7** | 1986-01-13 | 25.08 |
| **8** | 1986-01-14 | 24.97 |
| **9** | 1986-01-15 | 25.18 |

#Sort dataset by column Date

df = df.sort\_values('Date')

df = df.groupby('Date')['Closing Value'].sum().reset\_index()

df.set\_index('Date', inplace=True)

df=df.loc[datetime.date(year=2000,month=1,day=1):]

def DfInfo(df\_initial):

# gives some infos on columns types and numer of null values

tab\_info = pd.DataFrame(df\_initial.dtypes).T.rename(index={0: 'column type'})

tab\_info = tab\_info.append(pd.DataFrame(df\_initial.isnull().sum()).T.rename(index={0: 'null values (nb)'}))

tab\_info = tab\_info.append(pd.DataFrame(df\_initial.isnull().sum() / df\_initial.shape[0] \* 100).T.

rename(index={0: 'null values (%)'}))

return tab\_info

|  | **Closing Value** |
| --- | --- |
| **column type** | float64 |
| **null values (nb)** | 0 |
| **null values (%)** | 0.0 |

DatetimeIndex(['2000-01-04', '2000-01-05', '2000-01-06', '2000-01-07',

'2000-01-10', '2000-01-11', '2000-01-12', '2000-01-13',

'2000-01-14', '2000-01-18',

...

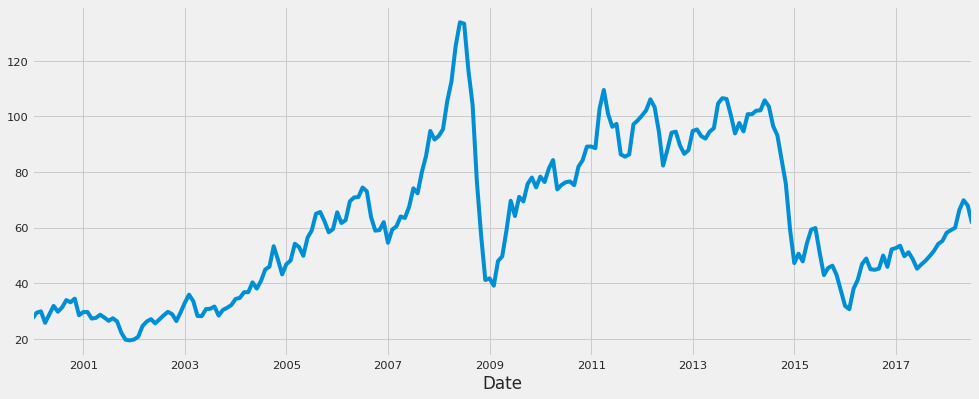
'2018-06-26', '2018-06-27', '2018-06-28', '2018-06-29',

'2018-07-02', '2018-07-03', '2018-07-04', '2018-07-05',

'2018-07-06', '2018-07-09'],

dtype='datetime64[ns]', name='Date', length=4673, freq=None)

y = df['Closing Value'].resample('MS').mean()



rcParams['figure.figsize'] = 18, 8

decomposition = sm.tsa.seasonal\_decompose(y, model='additive')

fig = decomposition.plot()

plt.show()

A picture containing histogram

Description automatically generated

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

df = sc.fit\_transform(df)

**Training and Testing**

train\_size = int(len(df) \* 0.70)

test\_size = len(df) - train\_size train\_size = int(len(df) \* 0.70)

test\_size = len(df) - train\_size

train, test = df[0:train\_size, :], df[train\_size:len(df), :]

def create\_data\_set(\_data\_set, \_look\_back=1):

data\_x, data\_y = [], []

for i in range(len(\_data\_set) - \_look\_back - 1):

a = \_data\_set[i:(i + \_look\_back), 0]

data\_x.append(a)

data\_y.append(\_data\_set[i + \_look\_back, 0])

return np.array(data\_x), np.array(data\_y)

regressor = Sequential()

regressor.add(LSTM(units = 60, return\_sequences = True, input\_shape = (X\_train.shape[1], 1)))

regressor.add(Dropout(0.1))

regressor.add(LSTM(units = 60, return\_sequences = True))

regressor.add(Dropout(0.1))

regressor.add(LSTM(units = 60))

regressor.add(Dropout(0.1))

regressor.add(Dense(units = 1))

regressor.compile(optimizer = 'adam', loss = 'mean\_squared\_error')

reduce\_lr = ReduceLROnPlateau(monitor='val\_loss',patience=5)

history =regressor.fit(X\_train, Y\_train, epochs = 20, batch\_size = 15,validation\_data=(X\_test, Y\_test), callbacks=[reduce\_lr],shuffle=False)

Epoch 1/20

212/212 [==============================] - 31s 122ms/step - loss: 0.0050 - val\_loss: 0.0231 - lr: 0.0010

Epoch 2/20

212/212 [==============================] - 24s 111ms/step - loss: 0.0117 - val\_loss: 0.0434 - lr: 0.0010

Epoch 3/20

212/212 [==============================] - 25s 116ms/step - loss: 0.0127 - val\_loss: 0.0520 - lr: 0.0010

Epoch 4/20

212/212 [==============================] - 23s 111ms/step - loss: 0.0166 - val\_loss: 0.0471 - lr: 0.0010

Epoch 5/20

212/212 [==============================] - 25s 117ms/step - loss: 0.0183 - val\_loss: 0.0527 - lr: 0.0010

Epoch 6/20

212/212 [==============================] - 24s 111ms/step - loss: 0.0161 - val\_loss: 0.0416 - lr: 0.0010

Epoch 7/20

212/212 [==============================] - 26s 122ms/step - loss: 0.0160 - val\_loss: 0.0029 - lr: 1.0000e-04

Epoch 8/20

212/212 [==============================] - 24s 115ms/step - loss: 0.0031 - val\_loss: 0.0021 - lr: 1.0000e-04

Epoch 9/20

212/212 [==============================] - 24s 115ms/step - loss: 0.0023 - val\_loss: 0.0018 - lr: 1.0000e-04

Epoch 10/20

212/212 [==============================] - 24s 111ms/step - loss: 0.0020 - val\_loss: 0.0016 - lr: 1.0000e-04

Epoch 11/20

212/212 [==============================] - 24s 115ms/step - loss: 0.0016 - val\_loss: 0.0015 - lr: 1.0000e-04

Epoch 12/20

212/212 [==============================] - 24s 113ms/step - loss: 0.0014 - val\_loss: 0.0014 - lr: 1.0000e-04

Epoch 13/20

212/212 [==============================] - 23s 111ms/step - loss: 0.0013 - val\_loss: 0.0014 - lr: 1.0000e-04

Epoch 14/20

212/212 [==============================] - 24s 115ms/step - loss: 0.0011 - val\_loss: 0.0013 - lr: 1.0000e-04

Epoch 15/20

212/212 [==============================] - 25s 116ms/step - loss: 0.0011 - val\_loss: 0.0013 - lr: 1.0000e-04

Epoch 16/20

212/212 [==============================] - 24s 115ms/step - loss: 0.0011 - val\_loss: 0.0013 - lr: 1.0000e-04

Epoch 17/20

212/212 [==============================] - 23s 109ms/step - loss: 0.0010 - val\_loss: 0.0014 - lr: 1.0000e-04

Epoch 18/20

212/212 [==============================] - 25s 116ms/step - loss: 0.0011 - val\_loss: 0.0013 - lr: 1.0000e-05

Epoch 19/20

212/212 [==============================] - 23s 110ms/step - loss: 9.6874e-04 - val\_loss: 0.0013 - lr: 1.0000e-05

Epoch 20/20

212/212 [==============================] - 24s 115ms/step - loss: 9.4882e-04 - val\_loss: 0.0013 - lr: 1.0000e-05

**Model Training**

train\_predict = regressor.predict(X\_train)

test\_predict = regressor.predict(X\_test)

train\_predict = sc.inverse\_transform(train\_predict)

Y\_train = sc.inverse\_transform([Y\_train])

test\_predict = sc.inverse\_transform(test\_predict)

Y\_test = sc.inverse\_transform([Y\_test])

print('Train Mean Absolute Error:', mean\_absolute\_error(Y\_train[0], train\_predict[:,0]))

print('Train Root Mean Squared Error:',np.sqrt(mean\_squared\_error(Y\_train[0], train\_predict[:,0])))

print('Test Mean Absolute Error:', mean\_absolute\_error(Y\_test[0], test\_predict[:,0]))

print('Test Root Mean Squared Error:',np.sqrt(mean\_squared\_error(Y\_test[0], test\_predict[:,0])))

plt.figure(figsize=(8,4))

plt.plot(history.history['loss'], label='Train Loss')

plt.plot(history.history['val\_loss'], label='Test Loss')

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epochs')

plt.legend(loc='upper right')

plt.show();

Train Mean Absolute Error: 2.4369837298603176

Train Root Mean Squared Error: 3.327537004178361

Test Mean Absolute Error: 2.383570259146614

Test Root Mean Squared Error: 5.270226533828571

**Chart, line chart

Description automatically generated**

aa=[x for x in range(180)]

plt.figure(figsize=(8,4))

plt.plot(aa, Y\_test[0][:180], marker='.', label="actual")

plt.plot(aa, test\_predict[:,0][:180], 'r', label="prediction")

plt.tight\_layout()

sns.despine(top=True)

plt.subplots\_adjust(left=0.07)

plt.ylabel('Price', size=15)

plt.xlabel('Time step', size=15)

plt.legend(fontsize=15)

plt.show();

**Chart, line chart

Description automatically generated**

**GITHUB & PROJECT DEMO LINK**

**GitHub Link:** [**https://github.com/IBM-EPBL/IBM-Project-14593-1659587542.git**](https://github.com/IBM-EPBL/IBM-Project-14593-1659587542.git)

**Demo Link:** [**https://drive.google.com/file/d/1hclB1Lu3uP5N9qOdK3jhS5zQDD69buCv/view?usp=sharing**](https://drive.google.com/file/d/1hclB1Lu3uP5N9qOdK3jhS5zQDD69buCv/view?usp=sharing)