IBM-Project-47385-1660798812

PROJECT DOCUMENTATION REPORT

CRUDE-OIL PRICE PREDICTION

TECHNOLOGY: ARTIFICIAL INTELLIGENCE

Team ID: PNT2022TMID46457

TEAM MEMBERS

Team Size: 4

Team Leader: SIVASAKTHI A

Team member: PRIYADHARSHINI B

Team member: SWETHA S

Team member: AFREEN YUSUFA A

1 INTRODUCTION:

PROJECT OVERVIEW:

This document is provided as a report for the project **Crude Oil Price Prediction.** Crude oil is amongst the most important resources in today's world, it is the chief fuel and its cost has a direct effect on the global habitat, our economy and oil exploration, exploitation and other activities. Prediction of oil prices has become theneed of the hour, it is a boon to many large and small industries, individuals, the government. The evaporative nature of crude oil, its price prediction becomes extremely difficult and it is hard to be precise with the same.Several different factors that affect crude oil prices.

PURPOSE:

The purpose of this document is to provide a clear-cut view of the project undertaken and produce a neat and greater understanding of the project.

2 LITERATURE SURVEY:

EXISTING PROBLEM:

One of the most significant commodities in the world, crude oil is responsible for one-third of the world's energy use. It serves as the foundation for the majority of the items we use on a daily basis, ranging from plastics to transportation fuels. Since changes in the price of crude oil have a significant impact on national economies around the world, price forecasting can help reduce the risks brought on by oil price volatility. For a variety of stakeholders, including governments, public and private organisations, policymakers, and investors, price projections are crucial.

REFERENCES:

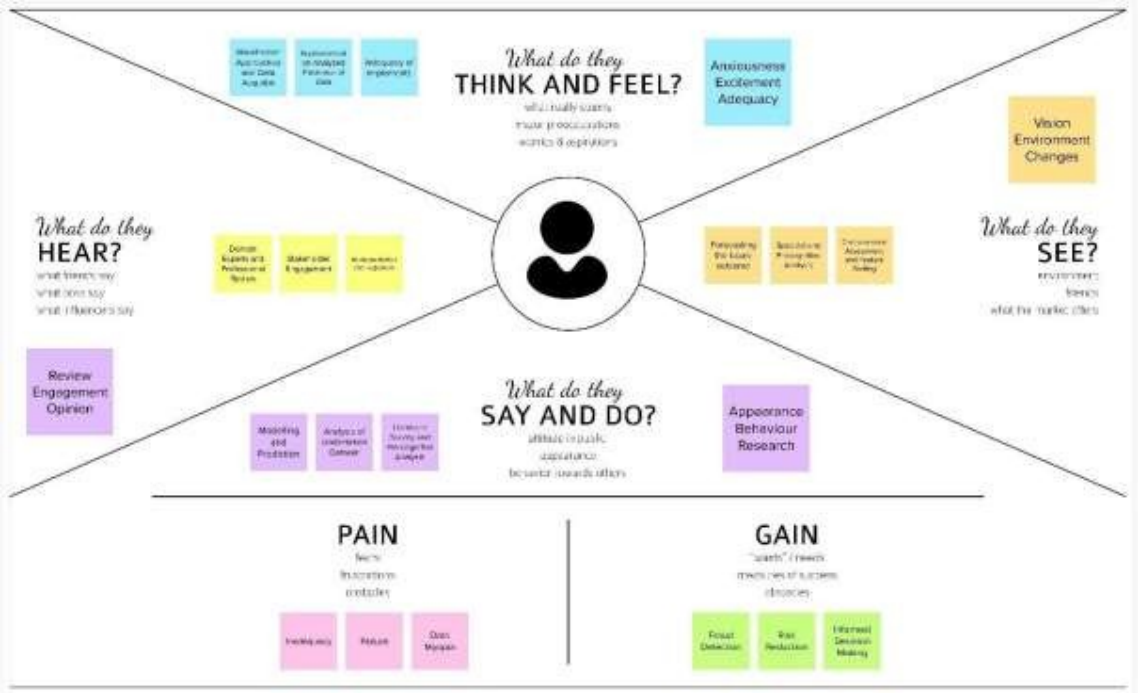
https://drive.google.com/drive/folders/1yq9UqoGpyAQFKR6ARNFwpVMofYtOHdCm?usp=sharing

PROBELM STATEMENT DEFINITION:

It is required to forecast CRUDE OIL PRICE in international market. The input and output should also be shown as charts and/or dashboards in various formats (like day, week, work-week, month, quarter, year, etc.). The models should be built with comprehensive explanation of data(using EDA), trend analysis, assumptions, data cleaning and validation, data augmentation (if required). Performance of various models need to be clearly evaluated and best model needs tobe recommended based on some robust evaluation criteria e.g., AIC (Akaike information criterion), Accuracy, RMSE, MSE etc.

3 IDEATION & PROPOSED SOLUTION:

EMPATHY MAPS:



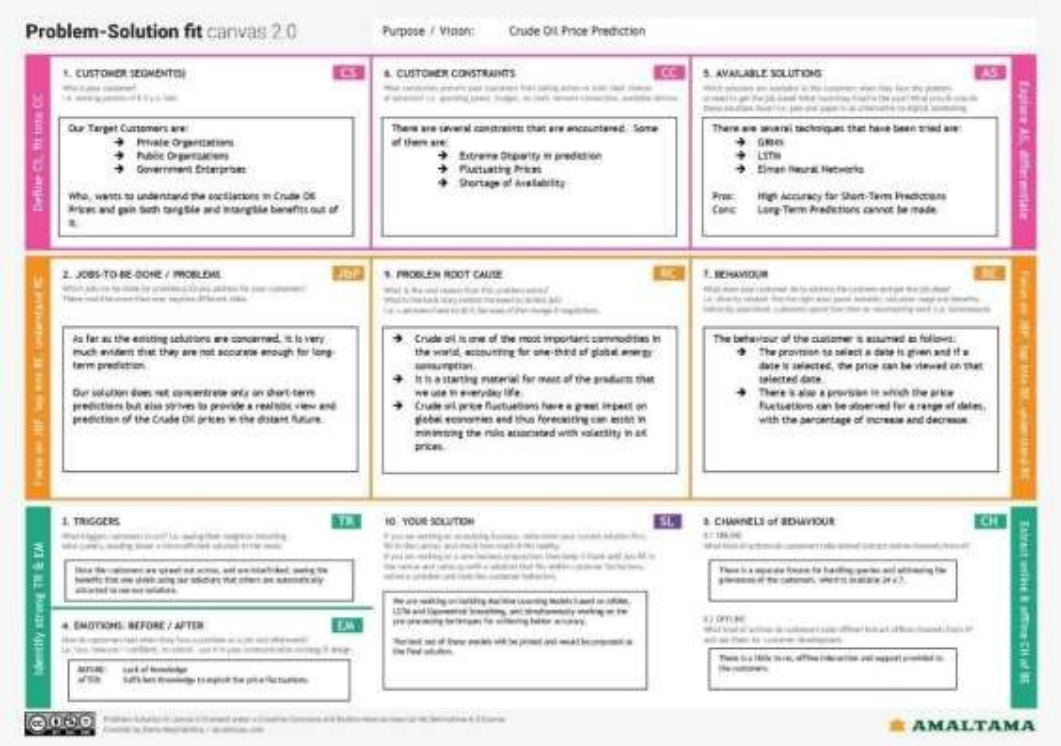
IDEATION AND BRAINSTORMING:

PROPOSED SOLUTION:

S. No.	Parameter	Description
1	Problem Statement (Problem to be Solved)	The existing patrimony model of oil price prediction is not capable enough to deliver the accurate predicted prices as expected. Few factors can be described as the conjectural buying and selling , geopolitical, OPEC output, increased demand from important role in the prediction of the oil prices. Now problem arising with the current ANN and CNN models that are used as prediction model's are that they can't provide accurate results when the data is too big.
2	Idea / Solution description	1) LSTM clears about keeping the previous data and prediction which might be encouraging and more accurate. The possible results are comparatively inspiring. 2) The LSTM model will be updated whenever new oil price data are available, and provided to model, so the model continuously evolves over time, and can capture the changing pattern of oil prices.
3	Novelty / Uniqueness	1) Price forecasting can assist in minimising the risks associated with volatility in oil prices. 2) Price forecasts are very important to various stakeholders: governments, public and private enterprises, policymakers, and investors.
4	Social Impact / Customer Satisfaction	1) Brand activation 2) Innovative and schemes 3) Instant reward schemes 4) Personalized consumer purchase exchanges 5) Capability building of sales personnel.

5	Business Model (Revenue Model)	1) The price of crude oil should be easily predictable from the equilibrium between demand and supply. 2) Traders analyze demand and supply factors and take calculated positions. If their prediction comes true, traders close their position to book profits way before expiry. 3) price of crude oil are changeable based from time to time.
6	Scalability of the Solution	1) hydrodynamic conditions in oilfield operations is suggested. 2) Modern refineries typically use a high number of sensors that generate an enormous amount of data. 3) Sustainable Solution for Crude Oil using Concentrated Solar Power Technology.

PROBLEM SOLUTION FIT:



4 REQUIREMENT ANALYSIS:

FUNCTIONAL REQUIREMENT:

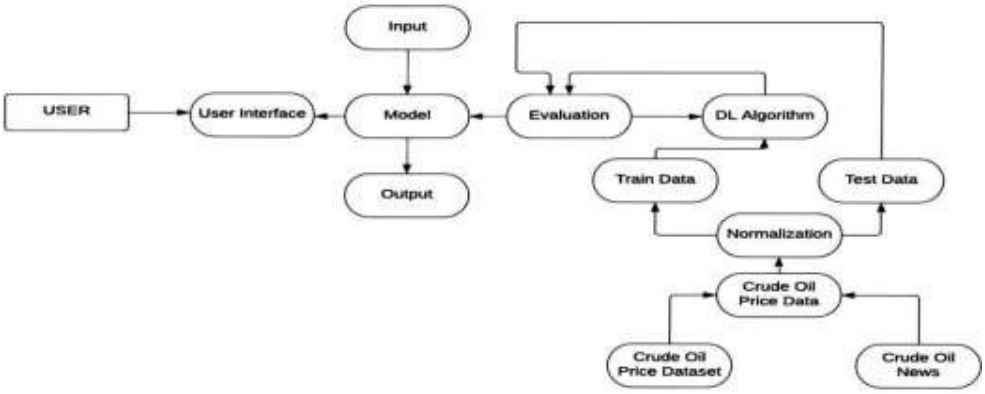
FRNO.	FUNCTIONAL REQUIREMENT	SUB REQUIREMENT
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Enquiry	Enter the date or range of dates
FR-4	User Visualization	Visualize the trend Enquire the prices Analyse the results
FR-5	User Endowment	See the results Gain Knowledge
FR-6	User Utilization	Use it in your idea Close the portal

NON-FUNCTIONAL REQUIREMENT:

FRNO.	NON-FUNCTIONAL REQUIREMENT	DESCRIPTION
FR-1	**Usability**	The application interface is easy to use and implement.
FR-2	**Security**	The credentials are secured and the result is encrypted.
FR-3	**Reliability**	The accuracy and reliability quotient is quoted to be high.
FR-4	**Performance**	The performance is uninterrupted and undeterred
FR-5	**Availability**	The data is freely available and the trend can be manually analysed
FR-6	**Scalability**	The predictions are scalable and reliable.

PROJECT DESIGN

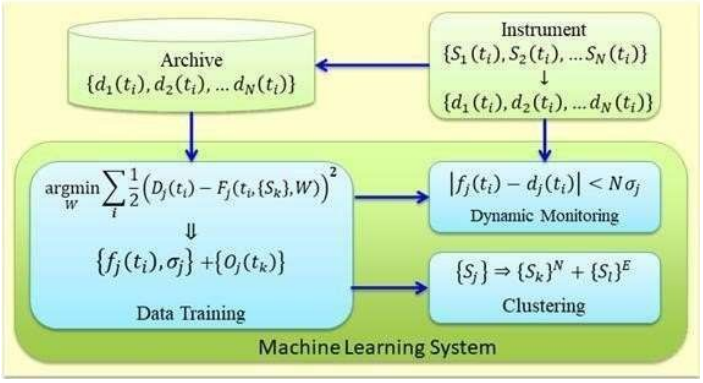
DATA FLOW DIAGRAM:



SOLUTION AND TECHNICAL ARCHITECTURE:

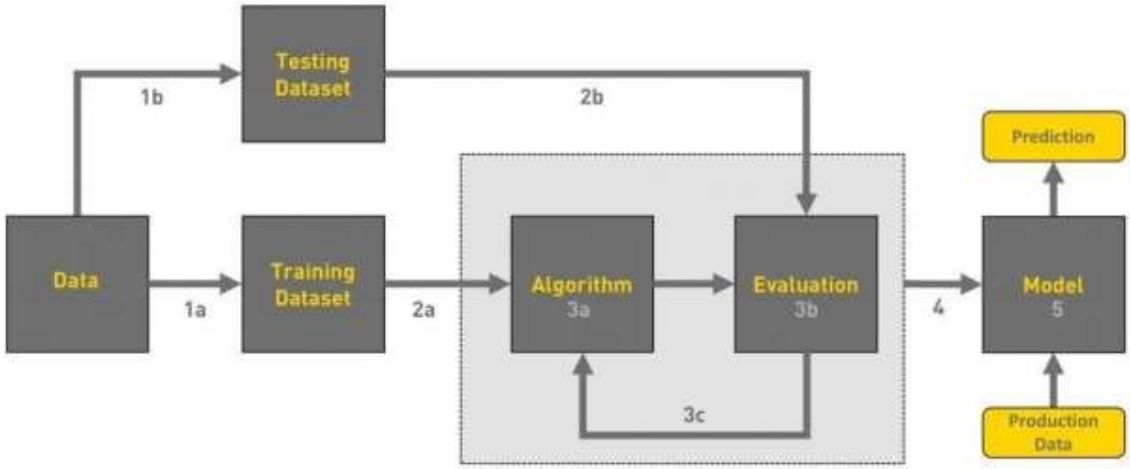
Context View

This view gives a high level representation of the system, the different user types and interactions with external entities. It describes the boundaries of the solution.



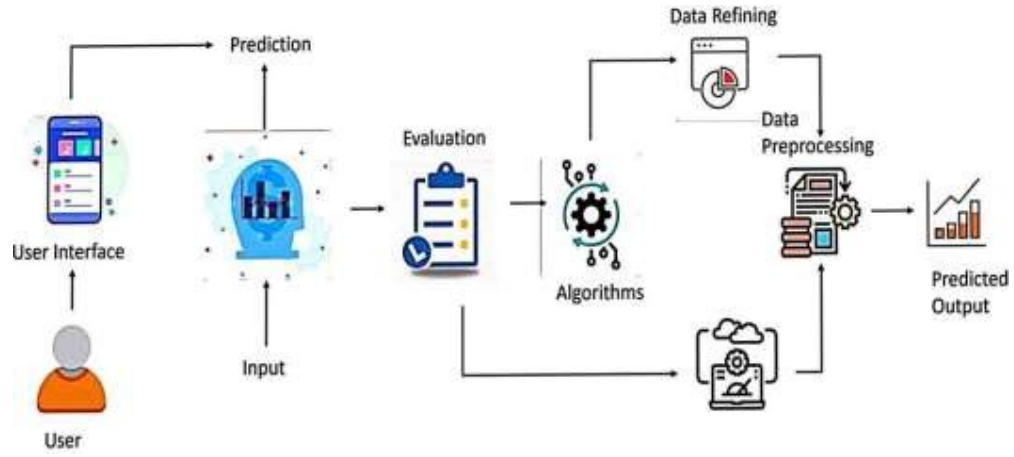
Project View

This section shows how key functionality relevant to the solution architecture maps to releases and milestones.



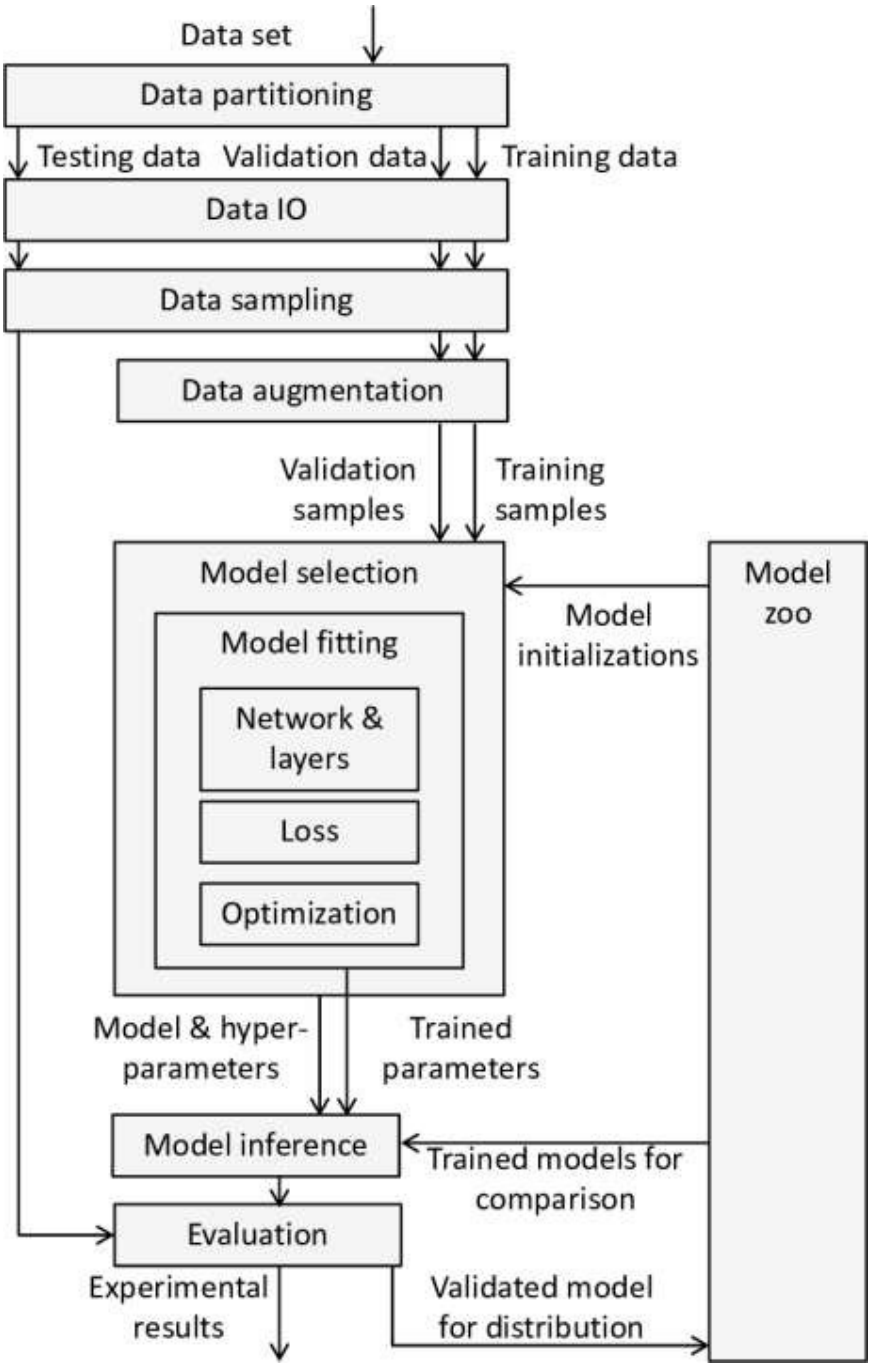
Functional View

This section describes the key functional areas of the project. The goal is to provide context around the architecture – all software performs some functionality and the definition of this functional scope is a very important factor to define the architecture.



Process View

The intent of the process view is to show how the various processing steps within the system fit together to implement the overall functional requirements. This is necessary if the system relies on workflow processes, forked or parallel processing mechanisms. The following processes are significant:



Non-Functional View

This section describes architecturally significant changes that enable the solution to achieve the agreed non-functional requirements (NFRs). Each change is mapped to the corresponding NFR category, which is based on the ISO/IEC 25010-2011 product quality model.

NFRs are documented and maintained in the Non-Functional Requirements Definition and will not be repeated here. In case of duplication, the Non-Functional Requirements Definition takes precedence.

Performance Easy tracking of records and updating can be done. All the requirements relating to performance characteristics of the system are specified in the section below. There are two types of requirements.

1. **Static Requirements:**

   These requirements do not impose any constraints on the execution characteristics of the system. They are:

   A) Number of Terminals: The software makes use of an underlying database that will reside at the same system, while the front end will be available to the administrative computer.

   B) Number of Users: The number of users can be administrator only, but this software can be extended to applications for almost all staff members of the organization.

2. **Dynamic Requirements:**

   These specify constraints on the execution characteristics of the system. They typically include response time and throughout of the system. Since these factors are not applicable to the proposed software, it will suffice if the response tine is high and the transactions are carried out precisely and quickly. Reliability: The software will not be able to connect to the database in the event of the server being down due to a hardware or software failure.

3. **Availability:**

   The software will be available only to administrator of the organization and the product as well as customer details will be recorded by him. He can add customers, update and delete them as well as add new products and manage them.

4. **Security:**

The security requirements deal with the primary security. The software should be handled only by the administrator and authorized users. Only the administrator has right to create new accounts and generating inventory. Only authorized users can access the system with username and password of administrator

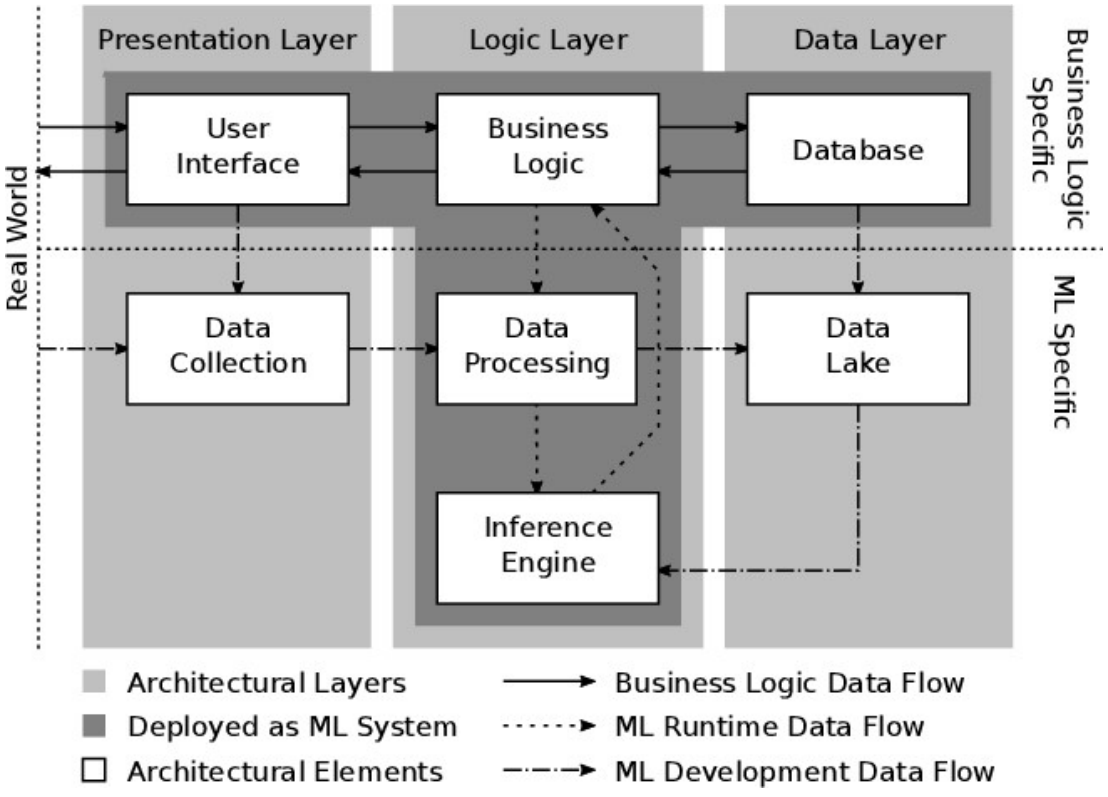
5. **Maintainability:**

Backups for database are available.

6. **Portability:**

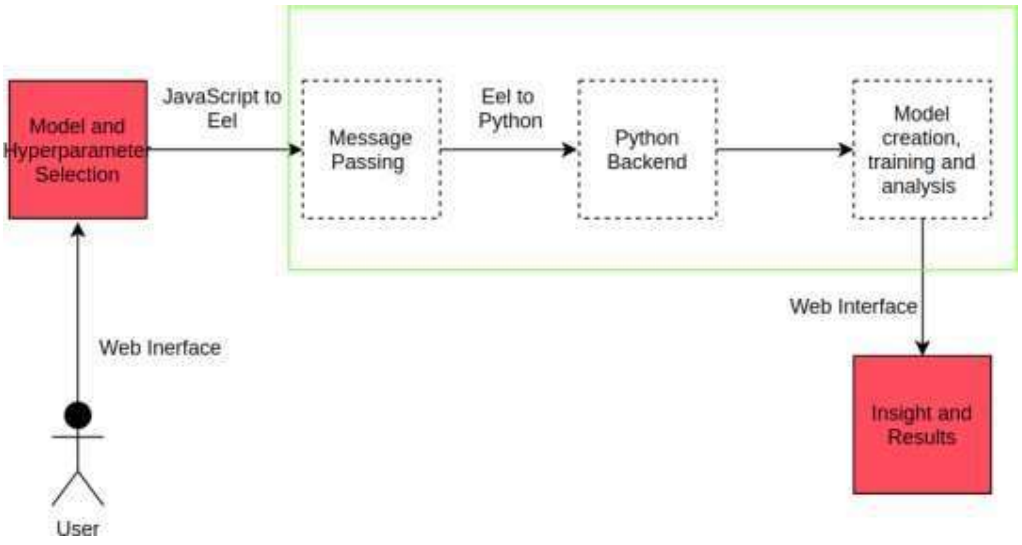
The Software is a web-based application and is built in Python and Mysql so it is platform independent and is independent of operating system.

Logical View



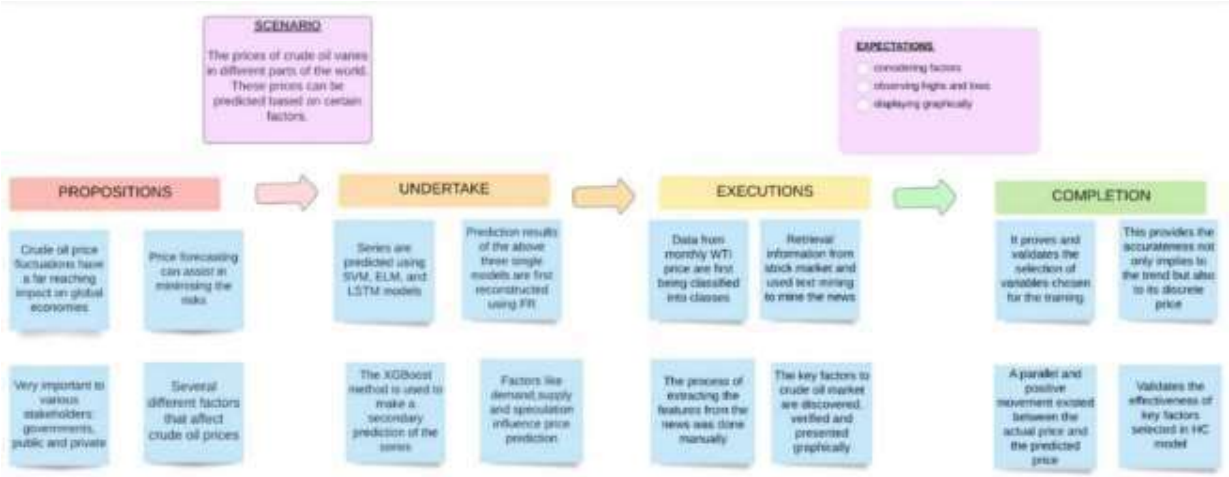
1 Interface View

This section describes the interfaces that will be required to the external systemintegration touch points

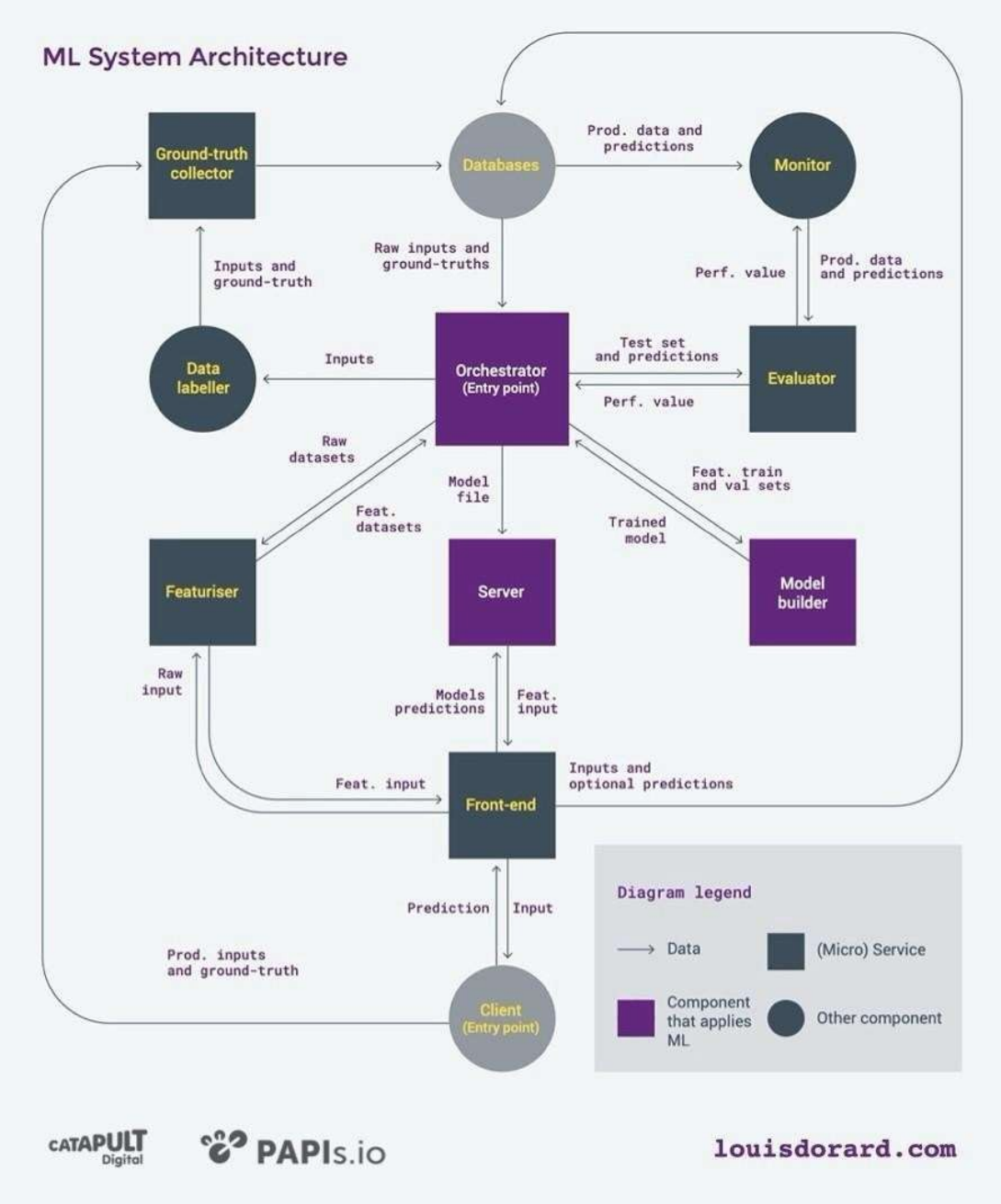


Design View

This section describes and explains any lower-level design concepts arising from thesolution if required.

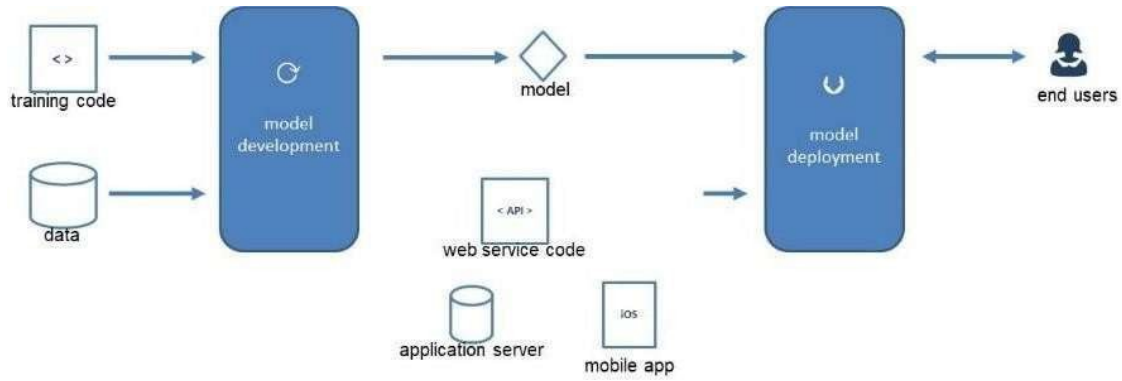


Physical View



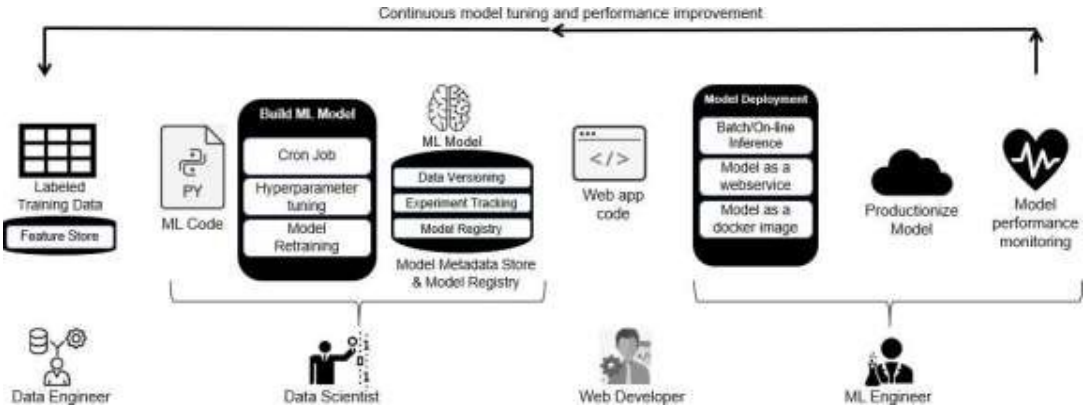
Deployment View

This section describes how code will be deployed in test environments and key considerations for the more complex Production go-live deployment.



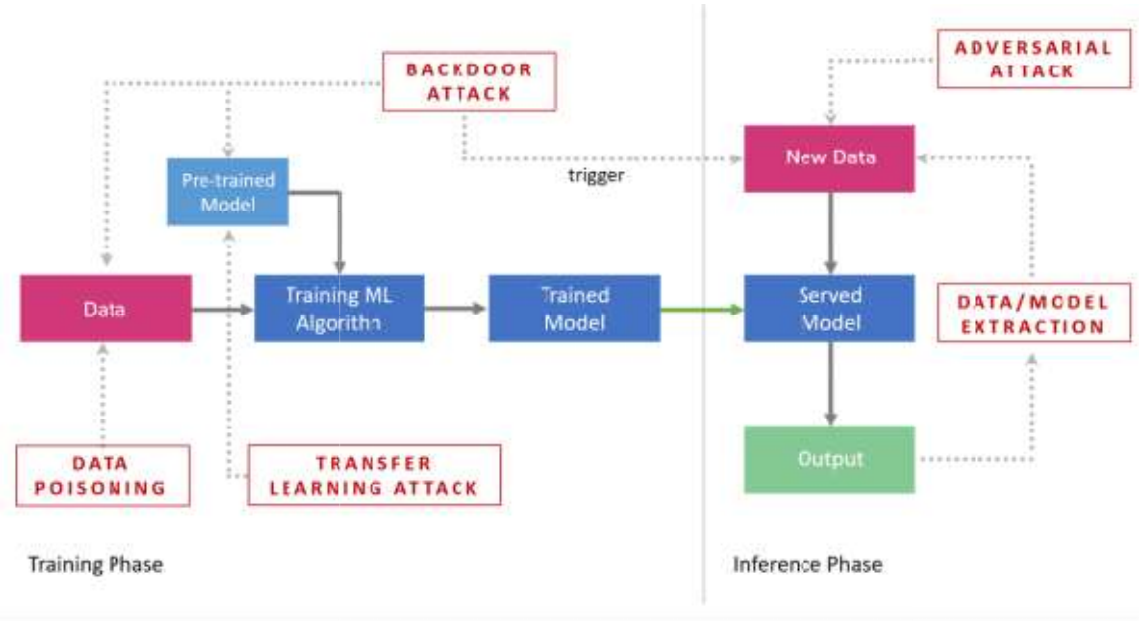
Operational View

This section describes how the architecture will support operational processes and activities.



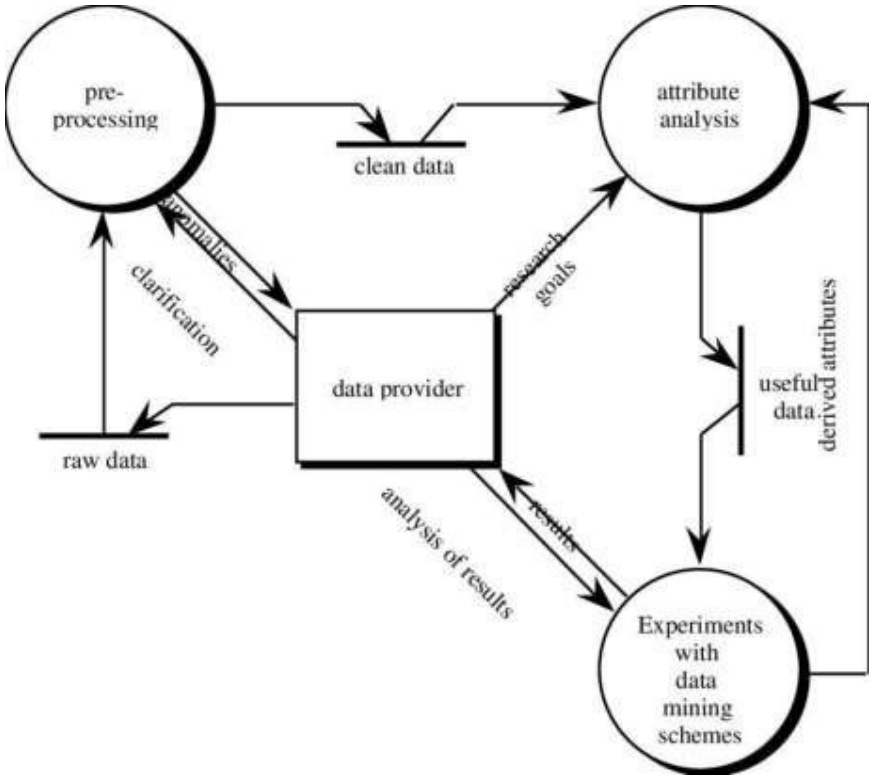
Security View

This section describes how the architecture addresses the different security aspects.



Data View

This section describes the important data model changes required to fulfill the requirements and the associated data flows.



USER STORIES:

USER TYPE	FUNCTIONAL REQUIREMENT	USER STORY	USER STORY / TASK	ACCEPTANCE CRITERIA	PRIORITY	RELEASE
Customer	Registration	1	Register for the Application through different vendors.	I can access my account / dashboard	High	Sprint-3
	Confirmation	2	Receiving Confirmation Mail	I can receive confirmation email & click confirm	High	Sprint-4
	Login	3	Log in into the application	Access to the account	High	Sprint-2
	Enquiry	4	Enter the range of dates	Plausible Range	High	Sprint-1
	Visualize	5	Visualize the Trend	Accuracy Check	High	Sprint-3
	Endowment	6	See the result	Prediction Check	High	Sprint-1
	Utilization	7	Log Out	Confirmation and Session Closure	High	Sprint-2
Administrator	Authority	1	Verify the imbalances	Session Dryness	High	Sprint-3

PROJECT PLANNING AND SCHEDULING:

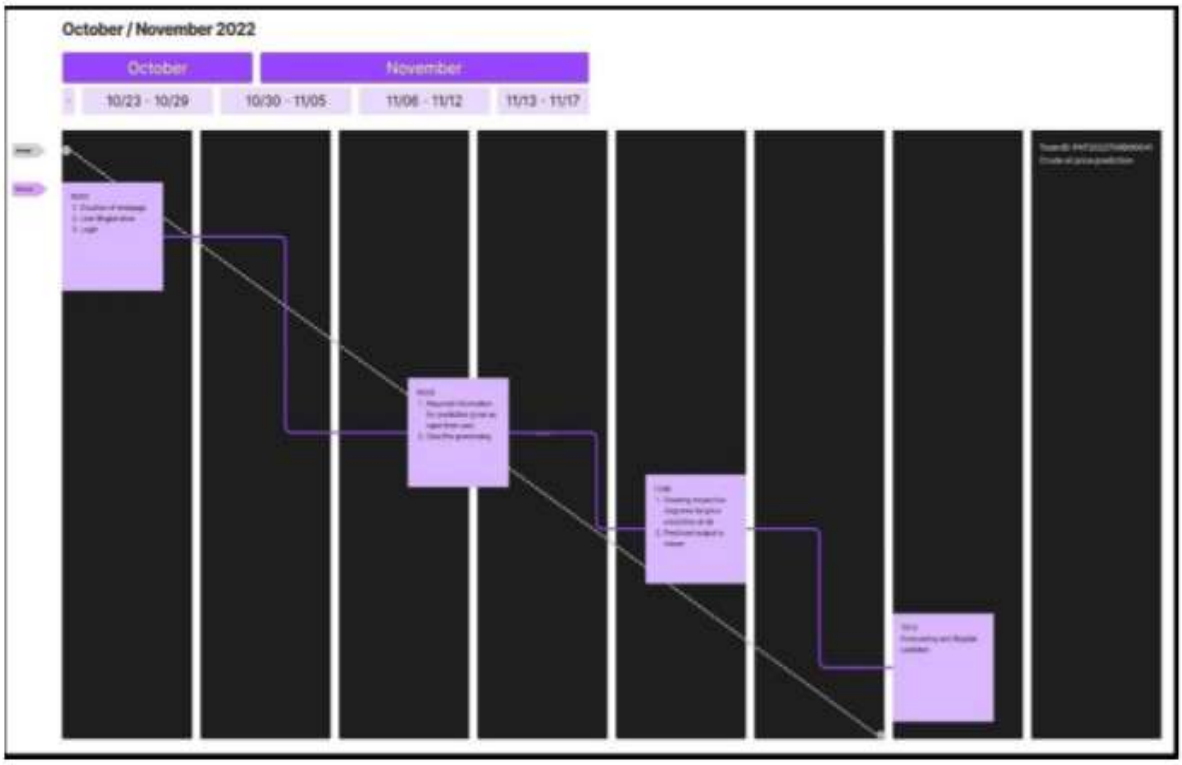
SPRINT PLANNING AND ESTIMATION:

SPRINT	FUNCTIONAL REQUIREMENT	USER STORY	USER STORY / TASK	STORY POINTS	PRIORITY	MEMBERS
1	Registration	1	Register for the Application	2	High	2
2	Confirmation	2	Receiving Confirmation Mail	1	Medium	2
2	Login	3	Log in into the application	2	High	2
3	Enquiry	4	Enter the range of dates	2	Medium	2
4	Visualize	5	Visualize the Trend	2	High	2
3	Endowment	6	See the result	2	High	2
4	Utilization	7	Log Out	1	Medium	2

SPRINT DELIVERY SCHEDULE:

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

REPORT FROM JIRA



CODING AND SOLUTION

FEATURE

Login.html

```
<!DOCTYPE html>
<html lan="en" and dir="Itr">
<head>
<meta charset="utf-8">
<title>login form</title>
<link rel="stylesheet" href="style.css">
<script src ="login.js"></script>
</head>
<body>
<form class="box" action="login.html" method="POST">
<h1>CRUDE OIL PRICE PREDICTION</h1>
<h2>
LOGIN
</h2>
<input type="text" name="" placeholder="Enter Username" id="username">
<input type="password" name="" placeholder="Enter Password" id="password">
<input type="submit" name="" value="Login" onclick="validate()">
<h3><a href="register.html"> New User ? Register </a></h3>
</form>

</body>
</html>
```

Register.html

```
<!DOCTYPE html>
<html lan="en" and dir="Itr">
<head>
<meta charset="utf-8">
<title>login form</title>
<link rel="stylesheet" href="register.css">
```

```
<script src ="login.js"></script>
</head>
<body>
<form class="box" action="login.html" method="POST">
<h1>CRUDE OIL PRICE PREDICTION</h1>
<h2>
Register
</h2>
<input type="text" name="" placeholder="Enter Username" id="username">
<input type="email" name="" placeholder="Enter Your Email Id" id="Email">
<input type="number" name="" placeholder="Enter Your Number" id="Number">
<input type="password" name="" placeholder="Enter Password" id="password">
<input type="submit" name="" value="Register" onclick="validate()">
<h3><a href="login.html"> Login </a></h3>
</form>

</body>
</html>
```

Style.css

```
body{
margin: 0;
padding: 0;
font-family: sans-serif;
background: url(p2.jpg);
background-size: cover;
}
.box{
width: 300px;
padding: 30px;
position: absolute;
top: 50%;
left: 50%;
transform: translate(-50%,-50%);
background: rgb(14, 14, 14);
```

```
text-align: center;
}
.box h1
{
color: rgb(253, 249, 251);
text-transform: uppercase;
font-weight: 700;
}
.box h2
{
color: rgb(253, 249, 251);
text-transform: uppercase;
font-weight: 700;
}
.box input[type="text"],.box input[type="password"] ,.box input[type="date"],.box
input[type="Number"],.box input[type="Email"]
{
border: 0;
background: white;
display: block;
margin: 28px auto;
text-align: center;
border: 3px solid #2af003;
padding: 14px 10px;
width: 220px;
outline: none;
color: #fff6ff(18, 18, 179);
border-radius: 24px;
transition: 0.25px;
}
.box input[type="text"]:focus,.box input[type="password"]:focus{
width: 270px;
border-color: rgb(238, 26, 203);
}
.box input[type="submit"]{
```

```
border: 0;
background: none;
display: block;
margin: 28px auto;
text-align: center;
border: 3px solid rgb(211, 15, 152);
padding: 14px 10px;
width: 220px;
outline: none;
color: rgb(73, 31, 224);
border-radius: 24px;
transition: 0.25px;
cursor: pointer;

}
.box input[type="submit"]:hover{
background: rgb(100, 182, 53);
}
h3{
color: wheat;
}
```

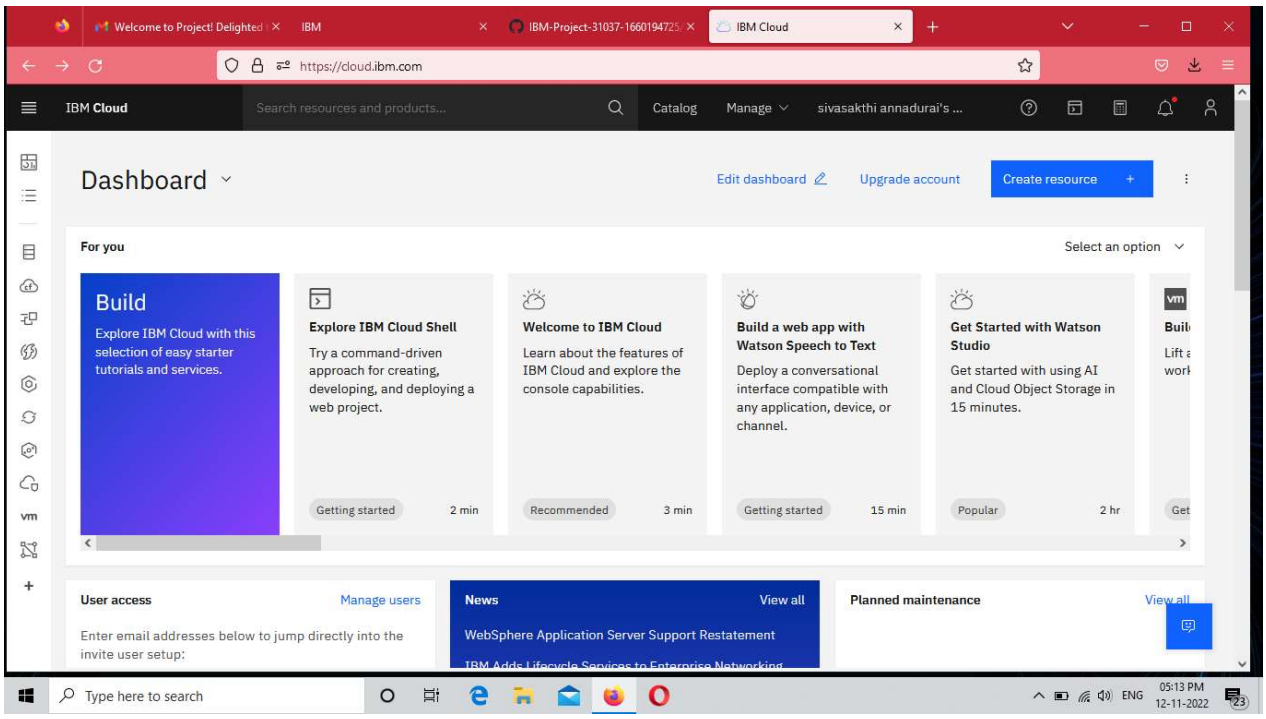
Register.css

```
body{
margin: 0;
padding: 0;
font-family: sans-serif;
background: url(ppp.jpg);
background-size: cover;
}
.box{
width: 300px;
padding: 30px;
position: absolute;
top: 50%;
```

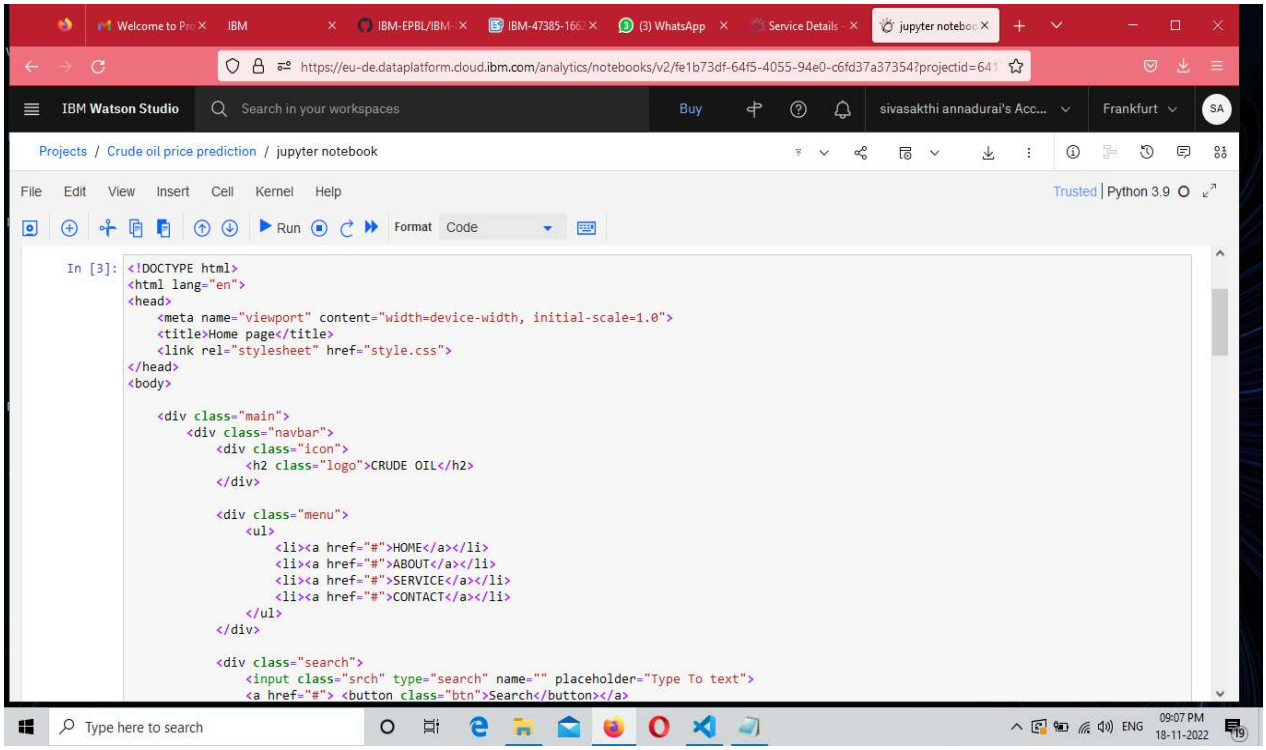
```
left: 50%;
transform: translate(-50%,-50%);
background: rgb(14, 14, 14);
text-align: center;
}
.box h1
{
color: rgb(253, 249, 251);
text-transform: uppercase;
font-weight: 700;
}
.box h2
{
color: rgb(253, 249, 251);
text-transform: uppercase;
font-weight: 700;
}
.box input[type="text"],.box input[type="password"] ,.box input[type="date"],.box
input[type="Number"],.box input[type="Email"]
{
border: 0;
background: white;
display: block;
margin: 28px auto;
text-align: center;
border: 3px solid #2af003;
padding: 14px 10px;
width: 220px;
outline: none;
color: #fff6ff(18, 18, 179);
border-radius: 24px;
transition: 0.25px;
}
.box input[type="text"]:focus,.box input[type="password"]:focus{
width: 270px;
```

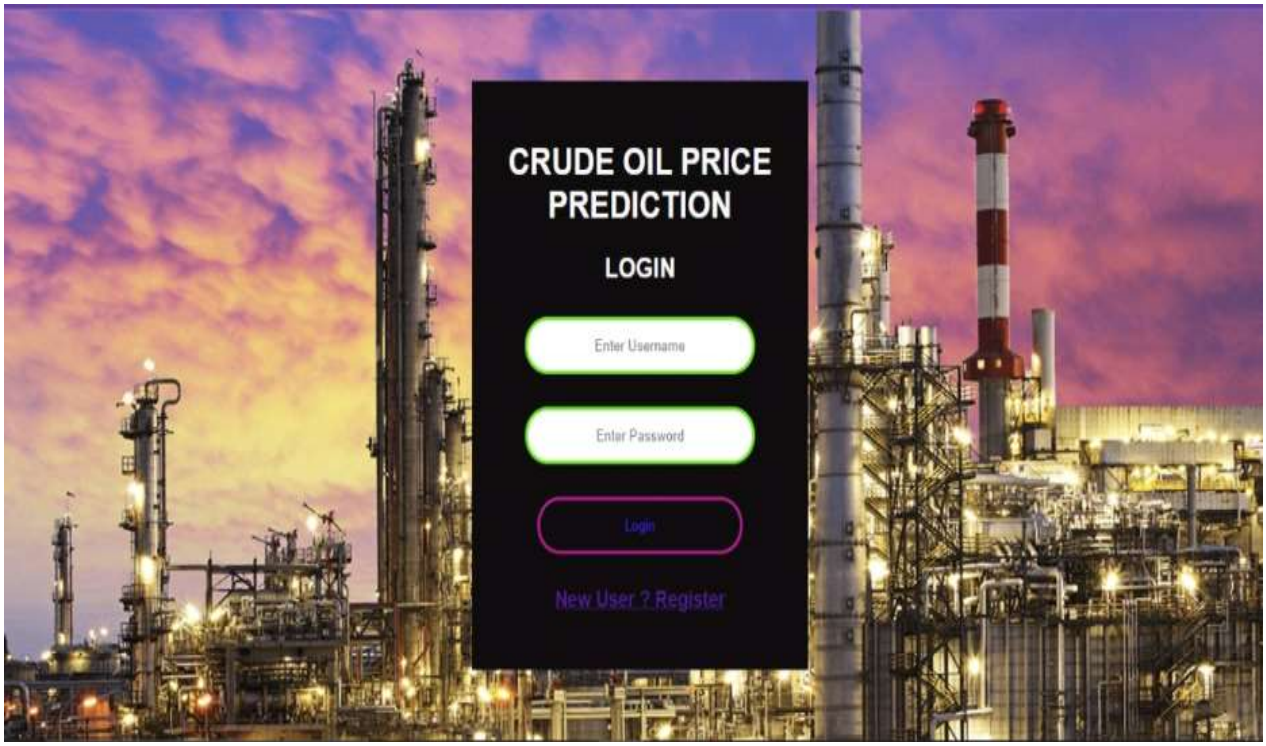
```
border-color: rgb(238, 26, 203);

}
.box input[type="submit"]{
border: 0;
background: none;
display: block;
margin: 28px auto;
text-align: center;
border: 3px solid rgb(211, 15, 152);
padding: 14px 10px;
width: 220px;
outline: none;
color: rgb(73, 31, 224);
border-radius: 24px;
transition: 0.25px;
cursor: pointer;

}
.box input[type="submit"]:hover{
background: rgb(100, 182, 53);
}
h3{
color: wheat;
}
```

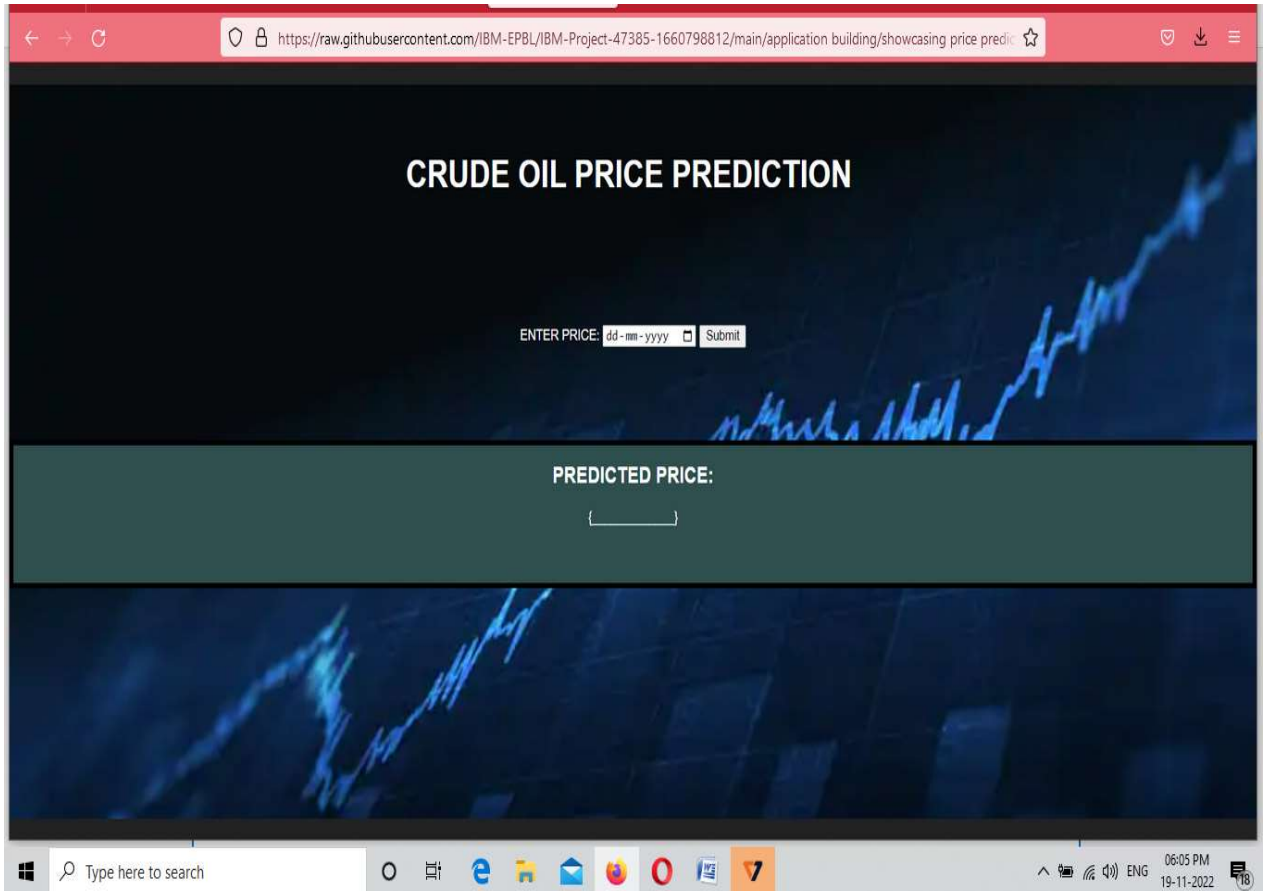
SCREENSHOTS:



CLOUD ACCOUNT CREATION




CRUDE OIL PRICE PREDICTION
LOGIN
Enter Username
Enter Password
Login
New User ? Register


https://raw.githubusercontent.com/IBM-EPBL/IBM-Project-47385-1660798812/main/application building/showcasing price predic
CRUDE OIL PRICE PREDICTION
ENTER PRICE: dd-mm-yyyy
Submit
PREDICTED PRICE:
{_____}
Type here to search
ENG
06:05 PM
19-11-2022

TESTING

TEST CASES

A test case has components that describe input, action and an expected response, in order to determine if a feature of an application is working correctly. A test case is a set of instructions on “HOW” to validate a particular test objective/target, which when followed will tell us if the expected behavior of the system is satisfied or not. Characteristics of a good test case:

- Accurate: Exacts the purpose.
- Economical: No unnecessary steps or words.
- Traceable: Capable of being traced to requirements.
- Repeatable: Can be used to perform the test over and over.
- Reusable: Can be reused if necessary.

USER ACCEPTANCE TESTING

This sort of testing is carried out by users, clients, or other authorised bodies to identify the requirements and operational procedures of an application or piece of software. The most crucial stage of testing is acceptance testing since it determines whether or not the customer will accept the application or programme. It could entail the application's U.I., performance, usability, and usefulness. It is also referred to as end-user testing, operational acceptance testing, and user acceptance testing (UAT).

ADVANTAGES & DISADVANTAGES

ADVANTAGE

• Give accurate result

• Easy to access and get the price

• Effective with large datasets

DISADVANTAGE

• Hard to find oil price

• Inefficient in accuracy

• Poor Customer support

CONCLUSION

Predicting Crude Oil prices is a very challenging problem due to the high volatility of oil prices. In this paper, we developed a new oil price prediction approach using ideas and tools from stream learning, a machine learning paradigm for analysis and inference of continuous flow of non-stationary data. Our stream learning model will be updated whenever new oil price data are available, and provided to model, so the model continuously evolves over time, and can capture the changing pattern of oil prices. In addition, updating the model requires only a small constant time per new data example, the experiment results show that our stream learning model outperformed four other popular oil price prediction models over a variety of forecast time horizons. This process is used to Predict the oil Prices. The prediction model predicts continuous valued functions.

FUTURE SCOPE

Future research may extend our work by considering a richer set of market variables, such as political or commercial factors and phases of economic instability, which are often determinants of crude oil price. Moreover, another direction for future research is the application of the proposed model to forecast the price of other commodities. Moreover, it is a worthwhile direction to explore the consideration of one or more computational cost factors when comparing different forecasting models. Therefore, calculations based on operational research methods might be a good direction.

APPENDIX

SOURCE CODE:

```
<!DOCTYPE html>
<html lan="en" and dir="Itr">
<head>
<meta charset="utf-8">
<title>login form</title>
<link rel="stylesheet" href="style.css">
<script src ="login.js"></script>
</head>
<body>
<form class="box" action="login.html" method="POST">
<h1>CRUDE OIL PRICE PREDICTION</h1>
<h2>
LOGIN
</h2>
<input type="text" name="" placeholder="Enter Username" id="username">
<input type="password" name="" placeholder="Enter Password" id="password">
<input type="submit" name="" value="Login" onclick="validate()">
<h3><a href="register.html"> New User ? Register </a></h3>
</form>

</body>
</html>

<!DOCTYPE html>
<html lan="en" and dir="Itr">
<head>
<meta charset="utf-8">
<title>login form</title>
<link rel="stylesheet" href="register.css">
<script src ="login.js"></script>
</head>
<body>
<form class="box" action="login.html" method="POST">
```

```
<h1>CRUDE OIL PRICE PREDICTION</h1>
<h2>
Register
</h2>
<input type="text" name="" placeholder="Enter Username" id="username">
<input type="email" name="" placeholder="Enter Your Email Id" id="Email">
<input type="number" name="" placeholder="Enter Your Number" id="Number">
<input type="password" name="" placeholder="Enter Password" id="password">
<input type="submit" name="" value="Register" onclick="validate()">
<h3><a href="login.html"> Login </a></h3>
</form>

</body>
</html>
```

Index.css

```
h1 {
text-align: center;
color: floralwhite;
font-size: 50px;
font-family: roboto;
}

p {
font-family: roboto;
color: ghostwhite;
margin-right: 30px;
margin-left: 30px;
text-align: center;
font-size: 20px;
font-weight: bold;
}

body {
background: url(index.png);
background-repeat: no-repeat;
background-size: cover;
```

```
}

.button {
display: inline-block;
border-radius: 4px;
background-color: black;
border: none;
color: #FFFFFF;
text-align: center;
font-size: 20px;
padding: 12px;
width: 100px;
transition: all 0.5s;
cursor: pointer;
margin: 5px;
}

a {
font-size: 20px;
font-family: roboto;
color: ghostwhite;
margin-right: 30px;
margin-left: 30px;
text-align: center;
font-size: 20px;
font-weight: bold;
}

table {
background: slateblue;
opacity: 0.8;
margin-left:auto;
margin-right:auto;
margin-bottom: 0px;
}

th,
td {
```

text-align: left;

color: black;

font-size: 30px;

font-family: roboto;

}

Predict.css

body{

background: url(index.png);

background-repeat: no-repeat;

background-size: cover;

}

Building python

IMPORTING LIBRARIES

```
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))
```

IMPORTING DATA

```
dateparse = lambda x: pd.datetime.strptime(x, '%b %d, %Y')
#Read csv file
from google.colab import files
uploaded = files.upload()
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Crude Oil Prices Daily.xlsx to Crude Oil Prices Daily.xlsx
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):]
df.head()
```

	Closing Value
Date	
2000-01-04	25.56
2000-01-05	24.65
2000-01-06	24.79
2000-01-07	24.79
2000-01-10	24.71

DATA PRE-PROCESSING

```
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
DfInfo(df)
```

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

```
df.index
```

```
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()
y.plot(figsize=(15, 6))
plt.show()

rcParams['figure.figsize'] = 18, 8
decomposition = sm.tsa.seasonal_decompose(y, model='additive')
fig = decomposition.plot()
plt.show()

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, 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)
look_back =90
X_train,Y_train,X_test,Ytest = [],[],[],[]
X_train,Y_train=create_data_set(train,look_back)
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
X_test,Y_test=create_data_set(test,look_back)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
```

LSTM LAYER

```
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 [==============================] - 23s 88ms/step - loss: 0.0047 - val_loss:
0.0251 - lr: 0.0010
Epoch 2/20
```

```
212/212 [==============================] - 17s 82ms/step - loss: 0.0122 - val_loss:
0.0478 - lr: 0.0010
Epoch 3/20
212/212 [==============================] - 17s 82ms/step - loss: 0.0115 - val_loss:
0.0505 - lr: 0.0010
Epoch 4/20
212/212 [==============================] - 17s 81ms/step - loss: 0.0163 - val_loss:
0.0461 - lr: 0.0010
Epoch 5/20
212/212 [==============================] - 19s 91ms/step - loss: 0.0193 - val_loss:
0.0461 - lr: 0.0010
Epoch 6/20
212/212 [==============================] - 17s 82ms/step - loss: 0.0174 - val_loss:
0.0605 - lr: 0.0010
Epoch 7/20
212/212 [==============================] - 18s 83ms/step - loss: 0.0275 - val_loss:
0.0047 - lr: 1.0000e-04
Epoch 8/20
212/212 [==============================] - 18s 83ms/step - loss: 0.0040 - val_loss:
0.0032 - lr: 1.0000e-04
Epoch 9/20
212/212 [==============================] - 17s 82ms/step - loss: 0.0029 - val_loss:
0.0021 - lr: 1.0000e-04
Epoch 10/20
212/212 [==============================] - 17s 81ms/step - loss: 0.0023 - val_loss:
0.0017 - lr: 1.0000e-04
Epoch 11/20
212/212 [==============================] - 17s 83ms/step - loss: 0.0020 - val_loss:
0.0016 - lr: 1.0000e-04
Epoch 12/20
212/212 [==============================] - 17s 82ms/step - loss: 0.0016 - val_loss:
0.0015 - lr: 1.0000e-04
Epoch 13/20
212/212 [==============================] - 17s 83ms/step - loss: 0.0014 - val_loss:
0.0014 - lr: 1.0000e-04
Epoch 14/20
212/212 [==============================] - 18s 83ms/step - loss: 0.0013 - val_loss:
0.0014 - lr: 1.0000e-04
Epoch 15/20
212/212 [==============================] - 18s 83ms/step - loss: 0.0012 - val_loss:
0.0013 - lr: 1.0000e-04
Epoch 16/20
212/212 [==============================] - 18s 84ms/step - loss: 0.0011 - val_loss:
0.0014 - lr: 1.0000e-04
Epoch 17/20
212/212 [==============================] - 18s 86ms/step - loss: 0.0011 - val_loss:
0.0014 - lr: 1.0000e-04
Epoch 18/20
212/212 [==============================] - 19s 87ms/step - loss: 0.0011 - val_loss:
0.0015 - lr: 1.0000e-04
Epoch 19/20
212/212 [==============================] - 17s 82ms/step - loss: 0.0011 - val_loss:
0.0013 - lr: 1.0000e-05
Epoch 20/20
212/212 [==============================] - 18s 83ms/step - loss: 0.0010 - val_loss:
0.0013 - lr: 1.0000e-05
```

MODEL TRAINING

```
train_predict = regressor.predict(X_train)
test_predict = regressor.predict(X_test)
100/100 [==============================] - 4s 27ms/step
41/41 [==============================] - 1s 28ms/step
train_predict = sc.inverse_transform(train_predict)
Y_train = sc.inverse_transform([Y_train])
test_predict = sc.inverse_transform(test_predict)
```

PREDICTION

```
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: 2.3165036988408305
Train Root Mean Squared Error: 3.285617879896689
Test Mean Absolute Error: 2.3989636110004624
Test Root Mean Squared Error: 5.289593391043789

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();
```

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-47385-1660798812