**EFFICIENT WATER QUALITY ANALYSIS AND PREDICTION USING MACHINELEARNING**

**Team ID : PNT2022TMID45992**

**Team Size : 4**

**Team Leader : SANTHOSH A**

**Team member : JONATHAN PAUL E**

**Team member : MOHAMED ARSHAD M M**

**Team member : SRIVIKRAM S**

1. **INTRODUCTION**

**1.1 PROJECT OVERVIEW**

Water is considered as a vital resource that affects various aspects of human health and lives. The quality of water is a major concern for people living in urban areas. The quality of water serves as a powerful environmental determinant and a foundation for the prevention and control of waterborne diseases. However predicting the urban water quality is a challenging task since the water quality varies in urban spaces non-linearly and depends on multiple factors, such as meteorology, water usage patterns, and land uses, so this project aims at building a Machine Learning (ML) model to Predict Water Quality by considering all water quality standard indicators.

**1.2 PURPOSE**

The quality of water has a direct influence on both human health and the environment. Water is utilized for a variety of purposes, including drinking, agriculture, and industrial use. The water quality index (WQI) is a critical indication for proper water management. The quality of water has a direct influence on both human health and the environment. Water is utilized for a variety of purposes, including drinking, agriculture, and industrial use. The water quality index (WQI) is a critical indication for proper water management.

In this project, we aim to design a web-based monitoring system for water quality and efficiency to be used for decision-making involving wastewater treatment plants. The current and present water quality data will be visualized by our web-based system. With our project, the quality of water will be predictable using machine learning algorithms. There are two parts to this project. Analyzing the water quality data for rivers, lakes, seas all around Turkey and analyzing the data for water treatment plants. The data for water treatment plants include samples taken from both the inlets and outlets of these plants.

**2. LITERATURE SURVEY**

**2.1 EXISTING PROBLEM**

In the Detection of quality from water has become one of the active research themes in water quality analysis and in applications . This research conducts an experimental study on water quality analysis and its predictions. These include detection of water quality by analyzing according to its components present in it. Our system focuses on analyzing of water quality. The aim of this research is to develop an system to predict the quality of water , whether it is safe to drink for human being.

**2.2 REERENCES**

1. APEC. 'The History of Clean Drinking Water', 2018. [Online]. Available: <https://www.freedrinkingwater.com/resource-history-of-clean-drinking-water.htm> [Accessed: 2020/11/01]
2. Minnesota Department of Health, 'Bacteria, Viruses, and Parasites in Drinking Water', 2019. [Online PDF]. Available: <https://www.health.state.mn.us/communities/environment/water/docs/contaminants/parasitesfactsht.pdf> [Accessed: 2020/11/01]
3. A. N. Ahmed, F. B. Othman, H. A. Afan, R. K. Ibrahim, C. M. Fai, M. S. Hossain, M. Ehteram, and A. Elshafie, “Machine learning methods for better water quality prediction,” Journal of Hydrology, vol. 578, p. 124084, Aug. 2019.
4. J.-T. Kuo, M.-H. Hsieh, W.-S. Lung, and N. She, “Using artificial neural networks for reservoir eutrophication prediction,” Ecological Modelling, vol. 200, no. 1-2, pp. 171–177, 2007. Retrieved from: <https://www.sciencedirect.com/science/article/abs/pii/S0304380006002985?via%3Dihub>
5. A. Zaqoot, A. K. Ansari, M. A. Unar, and S. H. Khan, “Prediction of dissolved oxygen in the Mediterranean Sea along Gaza, Palestine – an artificial neural network approach,” Water Science and Technology, vol. 60, no. 12, pp. 3051–3059, 2009. Retrieved from: <https://iwaponline.com/wst/article-abstract/60/12/3051/13774/Prediction-of-dissolved-oxygen-in-the?redirectedFrom=fulltext>

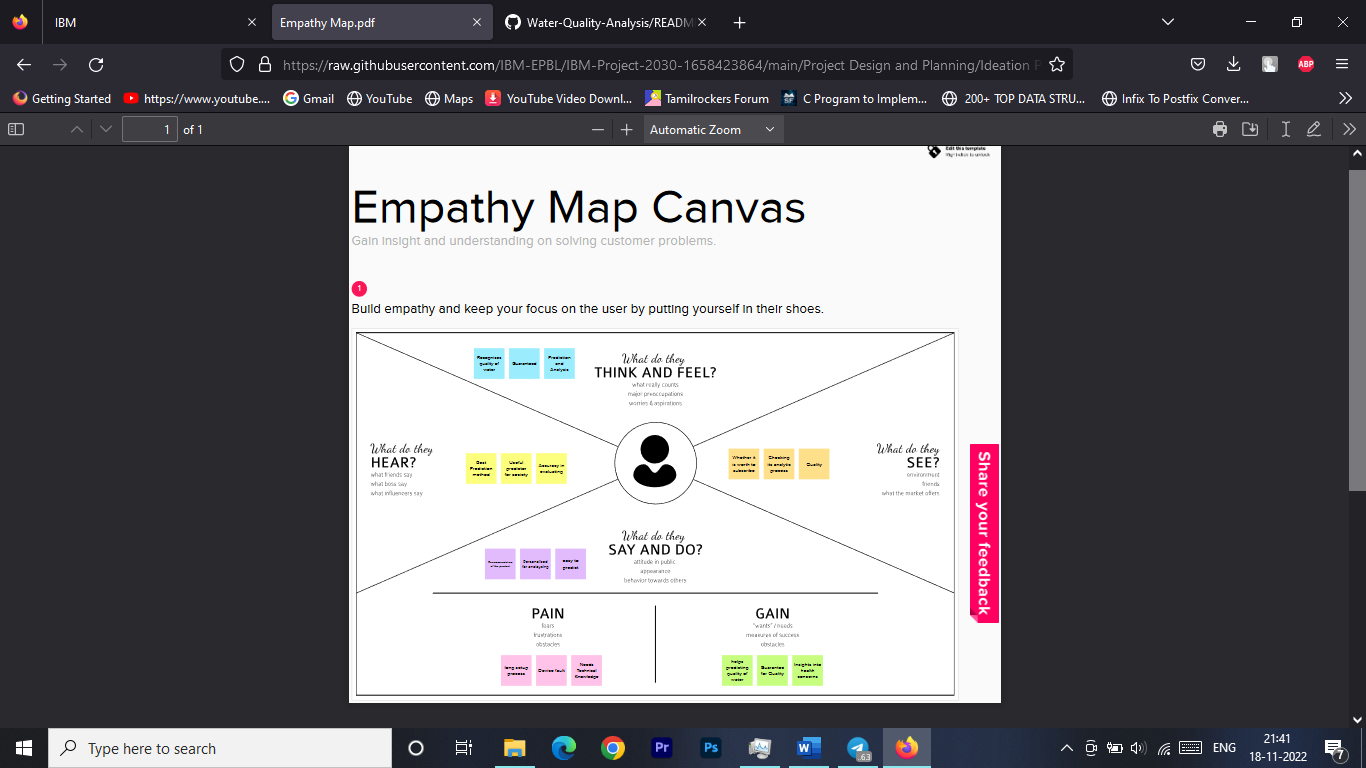
**2.3 PROBLEM STATEMENT DEFINITION**

Safe and readily available water is important for public health, whether it is used for drinking, domestic use, food production or recreational purposes. Better water supplies and sanitation, as well as better management of water resources, can contribute greatly to poverty reduction and economic growth. It is known that contaminated water and inadequate sanitation facilitate the transmission of diseases such as cholera, diarrhoea, dysentery, hepatitis A, typhoid, and polio. Those without access to clean water and sanitation face preventable health risks.

**3. IDEATION AND PROPOSED SOLUTION**

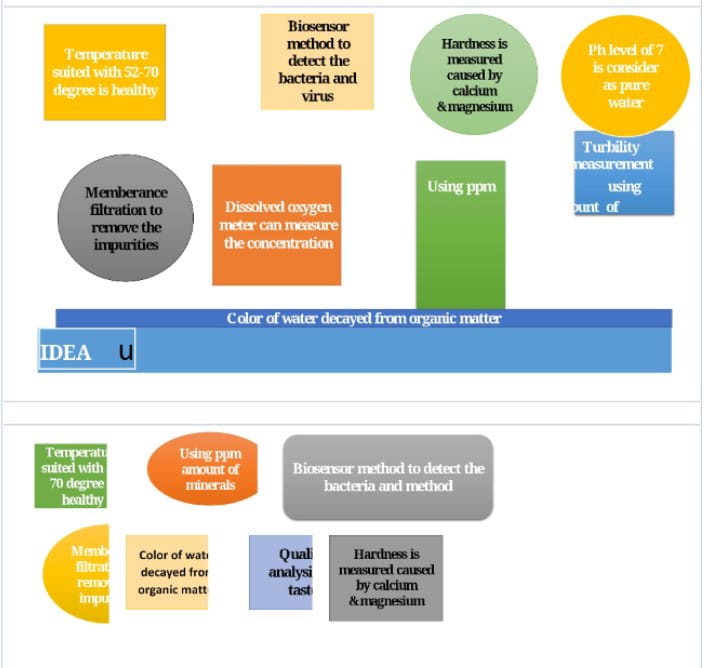
**3.1 EMPATHY MAP CANVAS**

An empathy map is a simple, easy-to-digest visuals that captures knowledge about an user’s behaviour and attitude. It is an useful tool to help team build a better understanding for their users. Creating an effective solution requires understanding the true problem he person who is experiencing it.

****

**3.2 IDEATION AND BRAINSTORMING**

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

****

**3.3 PROPOSED SOLUTION**

Project Design Phase-I Proposed Solution

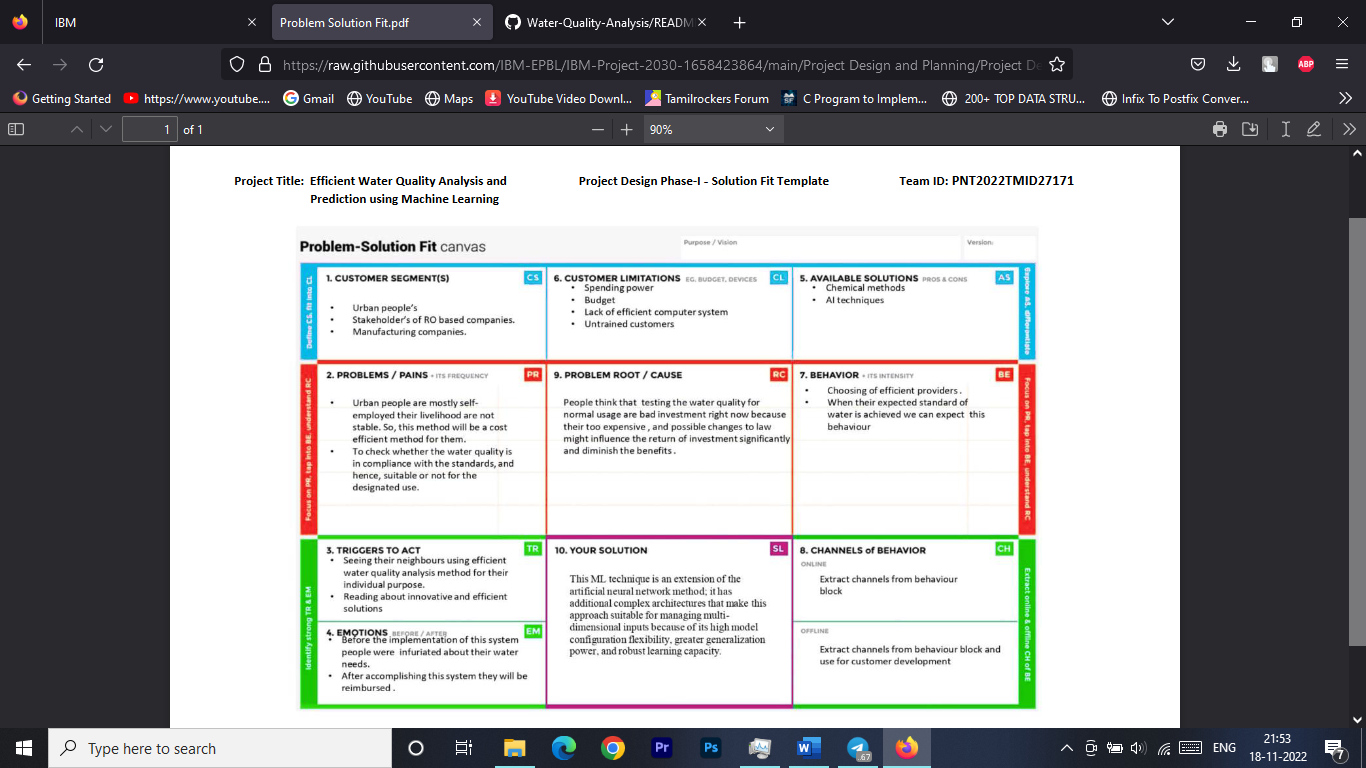
|  |  |
| --- | --- |
| Date | 17 November 2022 |
| Team ID | PNT2022TMID45992 |
| Project Name | Efficient Water Quality Analysis and Prediction using Machine Learning |
| Team Leader | SANTHOSH A |
| Team Members | SRIVIKRAM S, JONATHAN PAUL E, MOHAMED ARSHAD M M |
| Maximum Marks | 2 Marks |

Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

|  |  |  |
| --- | --- | --- |
| S.No. | Parameter | Description |
| 1. | Problem Statement (Problem to be solved) | Efficient Water Quality Analysis and Prediction using Machine Learning. |
| 2. | Idea / Solution description | For the WQI prediction, artificial neural network models, namely nonlinear autoregressive neural network (NARNET) and long short-term memory (LSTM) deep learning algorithm, have been developed. In addition, three machine learning algorithms, namely, support vector machine (SVM), K- nearest neighbour (K-NN), and Naive Bayes, have been used for the WQC forecasting. The used dataset has 7 significant parameters, and the developed models were evaluated based on some statistical parameters |
| 3. | Novelty / Uniqueness | In previous they find water quality with help of WQI and WQC. Now the solution is find with help of advanced artificial intelligence and it include seven  parameters |
| 4. | Social Impact / Customer Satisfaction | During the last years, water quality has been threatened by various pollutants. Therefore, modelling and predicting water quality have become very important in controlling water pollution. In this work, advanced artificial intelligence (AI) algorithms are developed to predict water quality index (WQI) and water quality classification (WQC).This is the impact of this statement. |

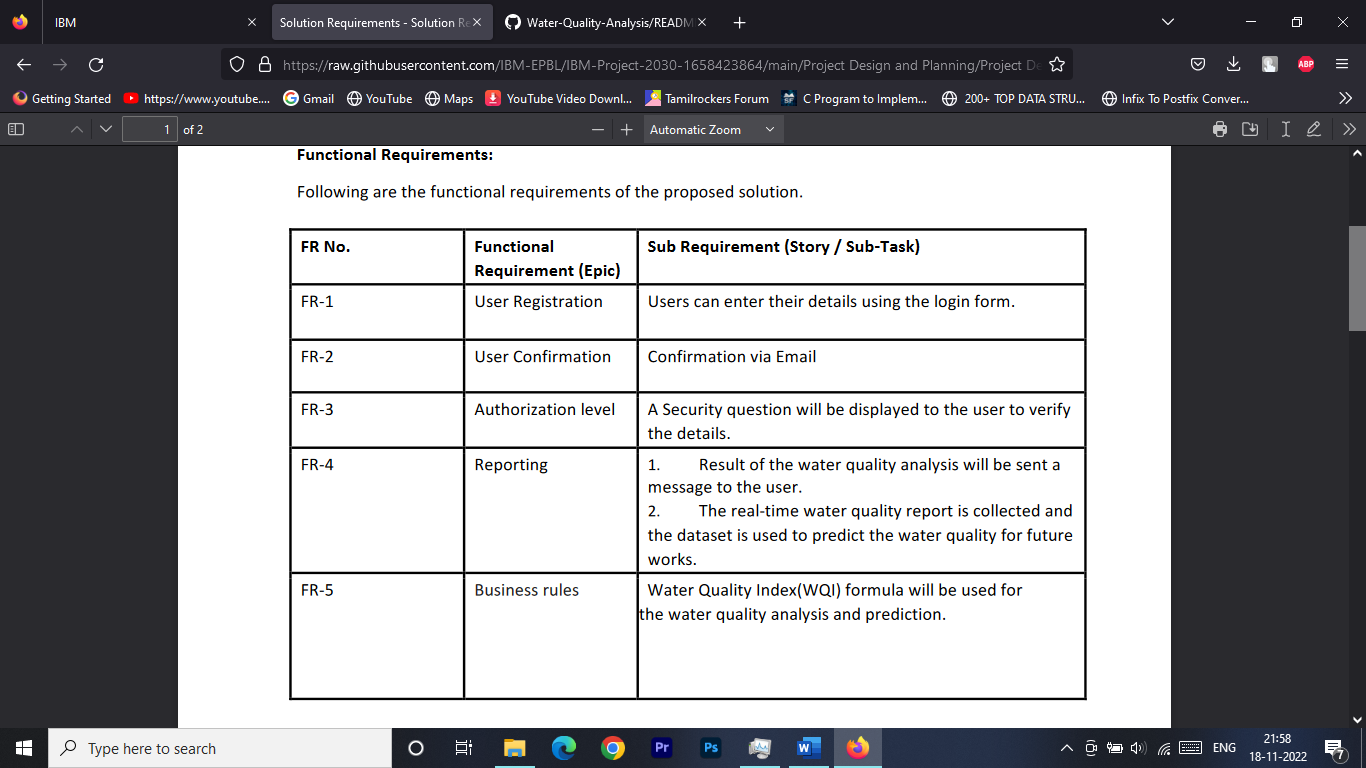
|  |  |  |
| --- | --- | --- |
|  |  |  |
| 5. | Business Model (Revenue Model) | The revenue stream include the Promoted trends and method. Technology and production is improved in business side. It increased the profit and also the logistic  way. |
| 6. | Scalability of the Solution | Scalability of this solution can handle any amount of data and perform many computations in a cost effective and time saving to instantly serve millions of users residing at global location. |

**3.4 PROPOSED SOLUTION FIT**

**4. REQUIREMENT ANALYSIS**

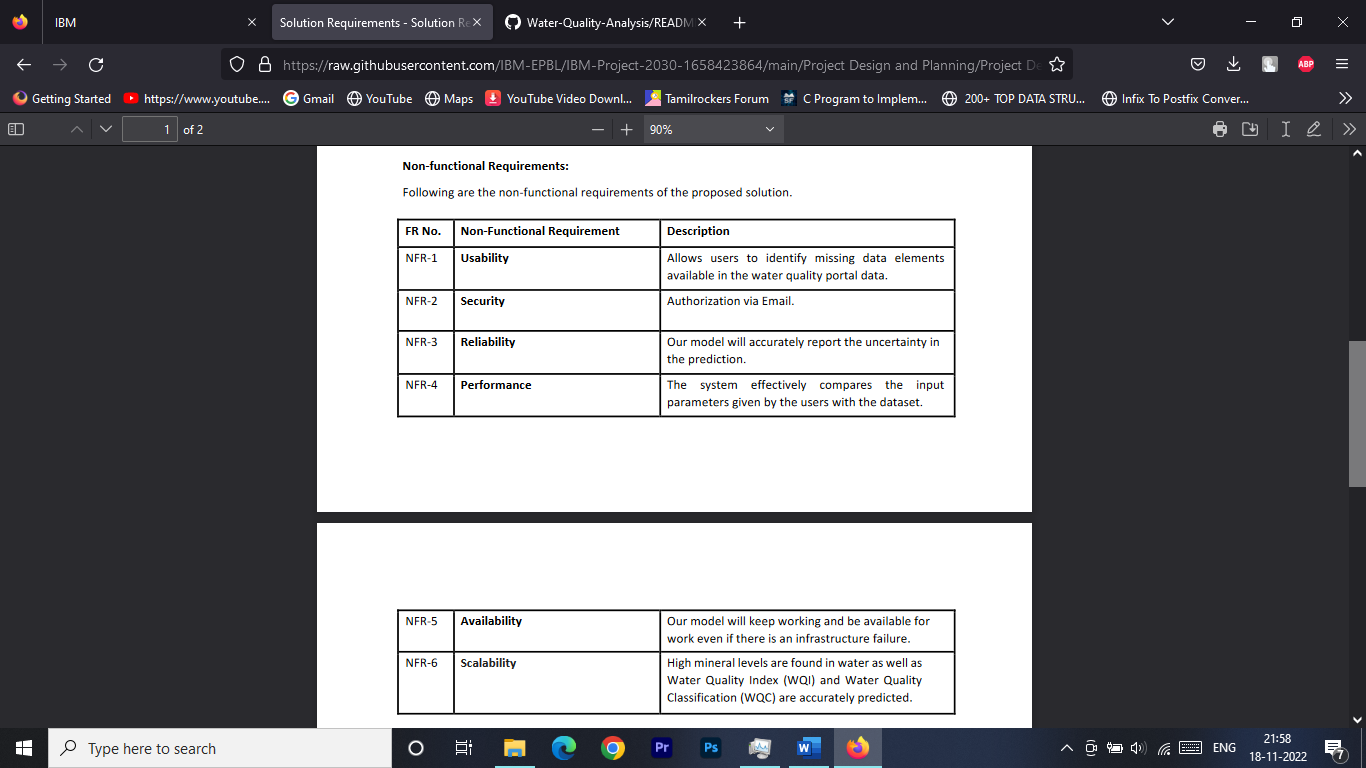
**4.1 FUNCTIONAL REQUIREMENTS**

Following are the functional requirements of the proposed solution.



**4.2 NON-FUNCTIONAL REQUIREMENTS**

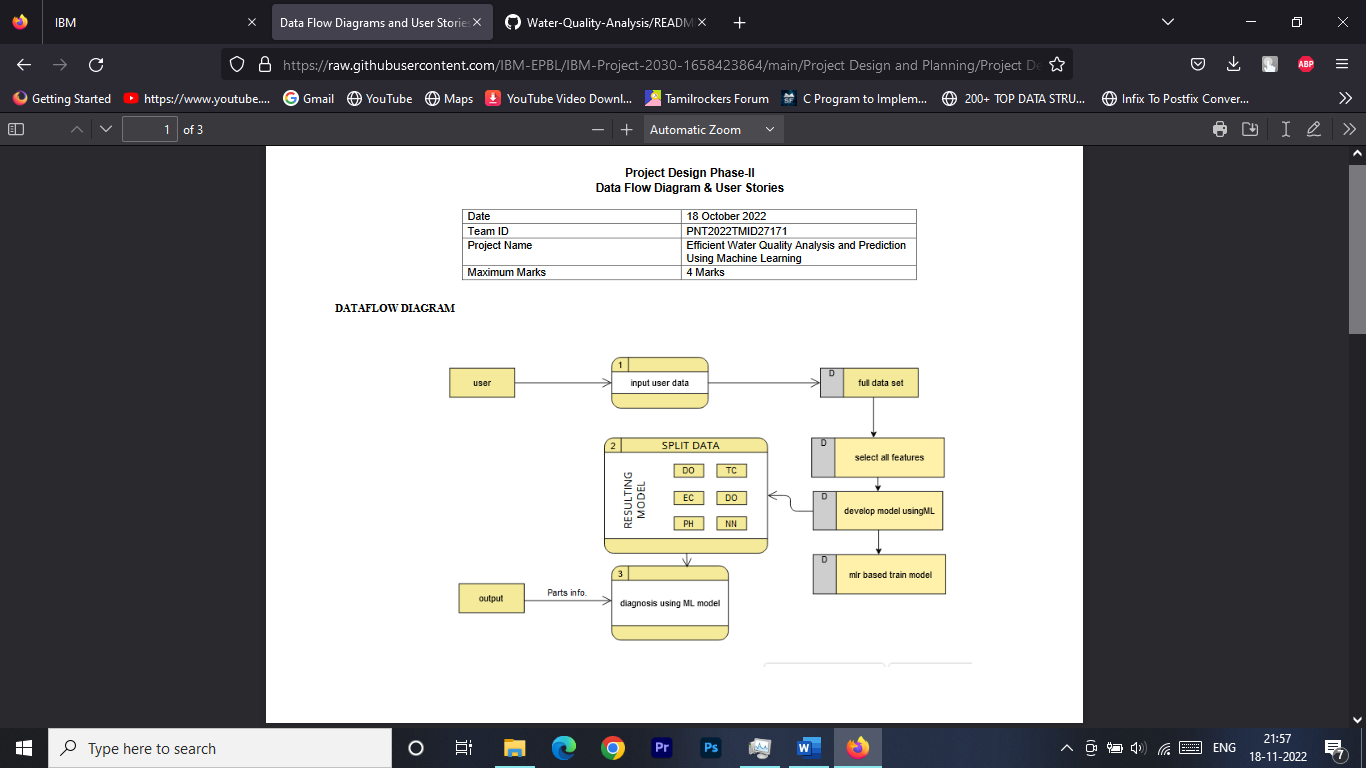
Following are the non-functional requirements of the proposed solution.



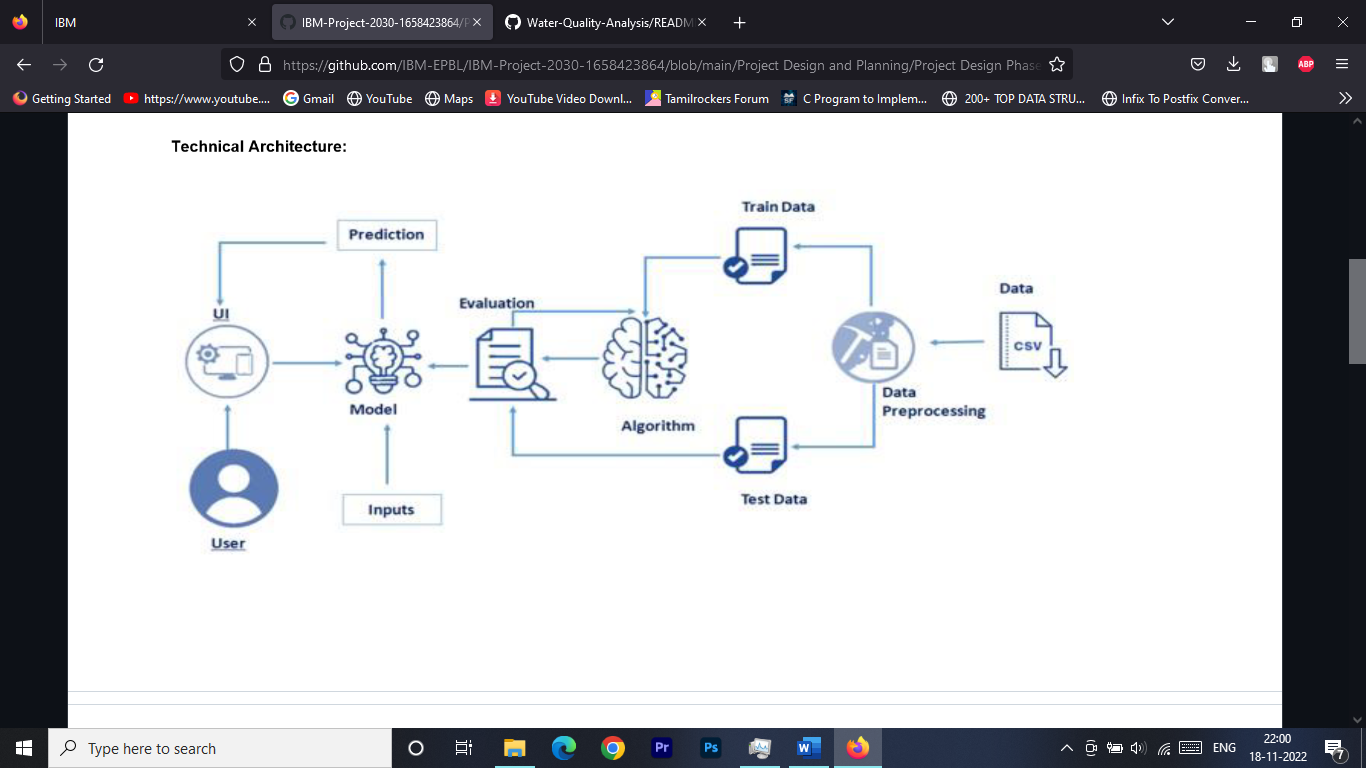
**5. PROJECT DESIGN**

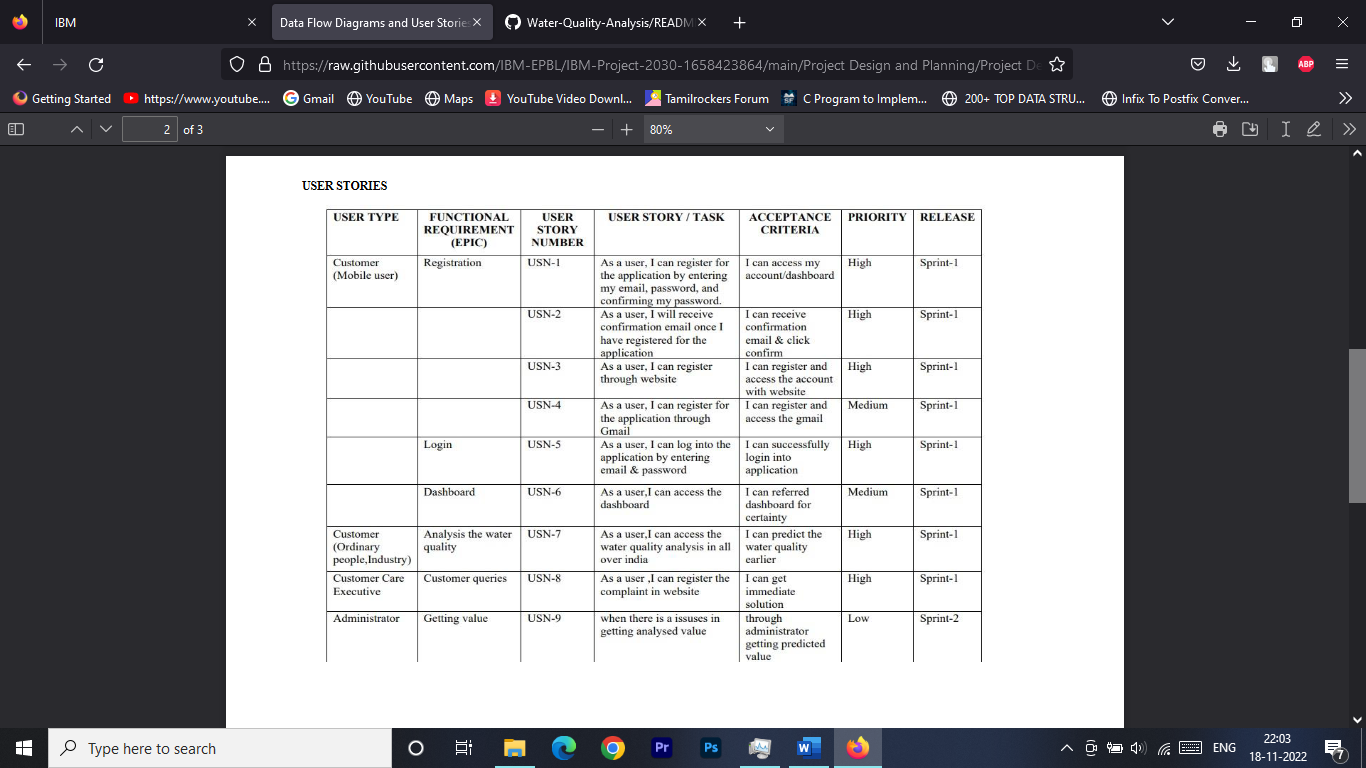
**5.1 DATA FLOW DIAGRAM**

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



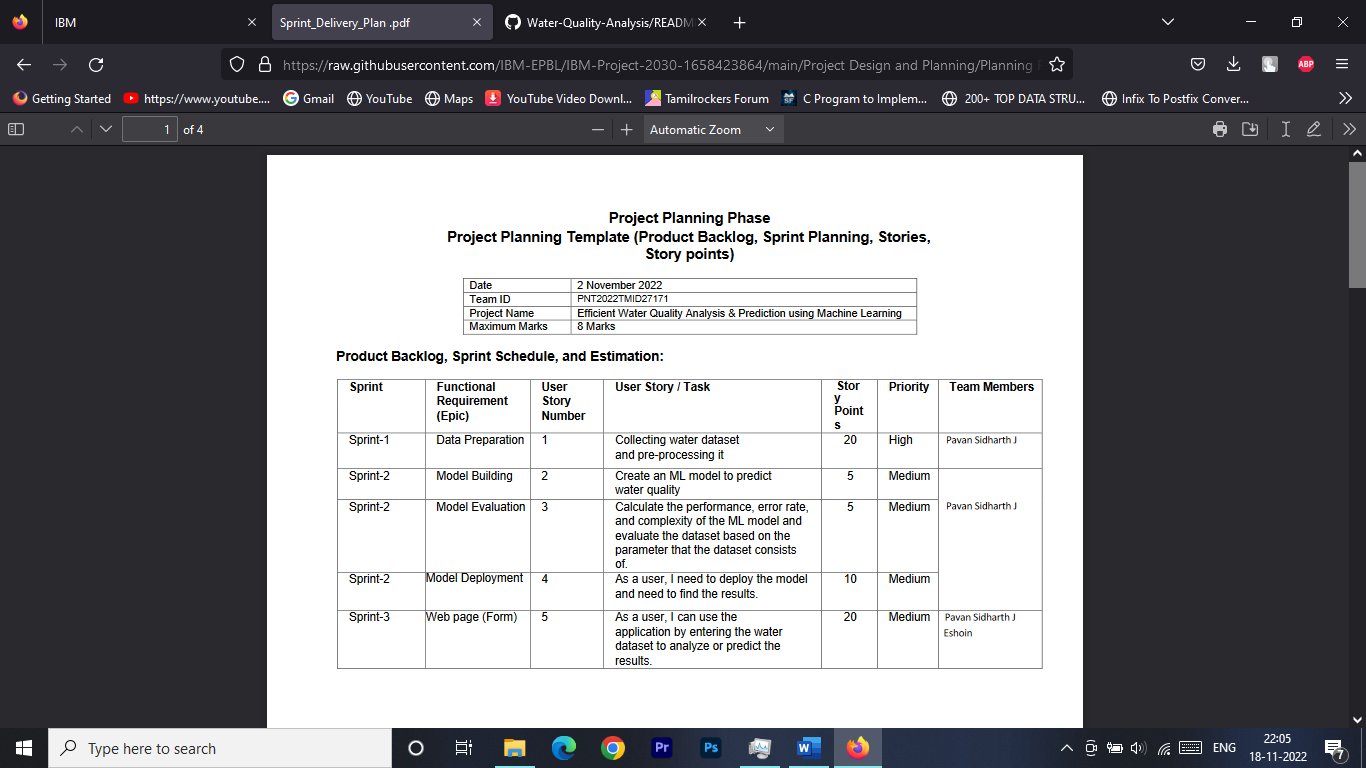
**5.2 SOLUTION AND TECHNICAL ARCHITECTURE**

****

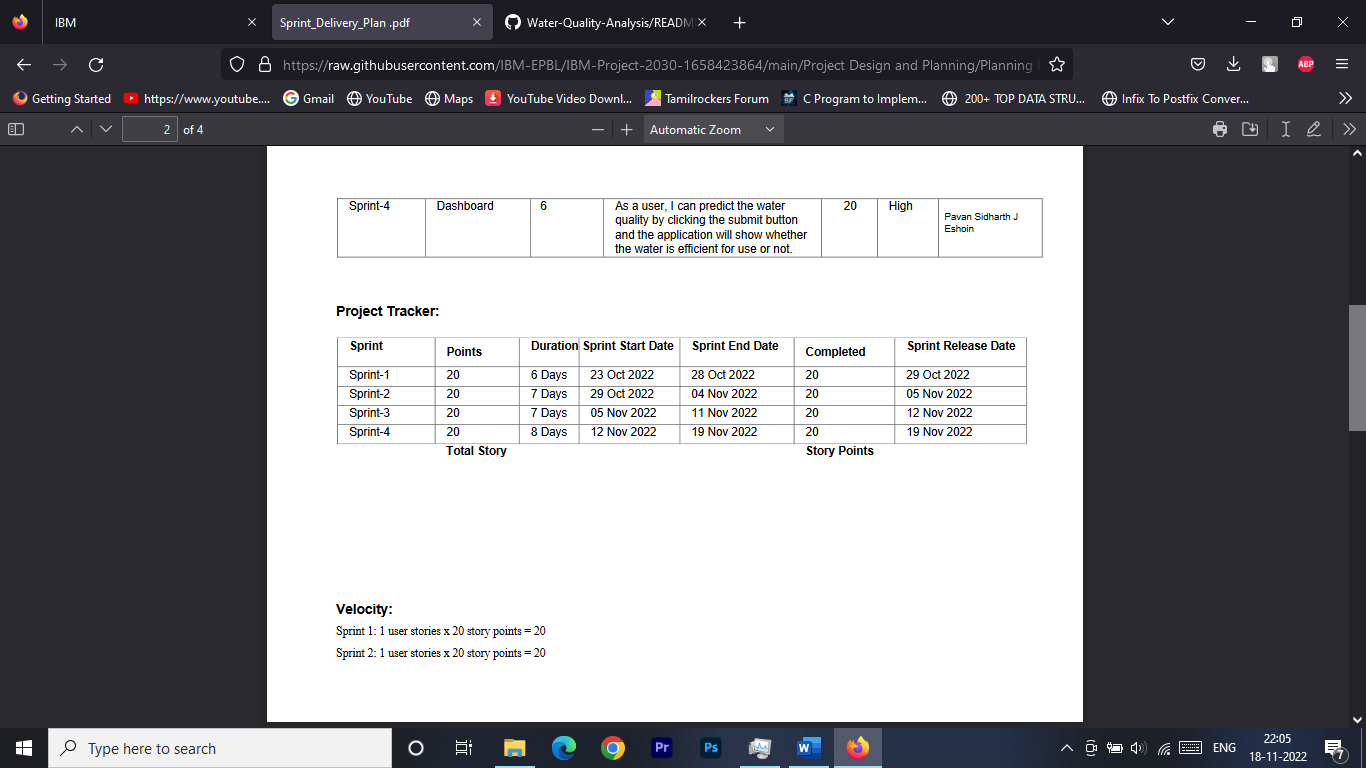
**5.3 USER STORIES**

**6. PROJECT PLANNING AND SCHEDULING**

**6.1 SPRINT PLANNING AND ESTIMATION**



**6.2 SPRINT DELIVERY SCHEDULE**



**7 .CODING AND SOLUTIONING**

**7.1 FEATURE 1**

HTML CODE

<!DOCTYPEhtml>

<htmllang="en">

<head>

<metacharset="UTF-8">

<metahttp-equiv="X-UA-Compatible"content="IE=edge">

<metaname="viewport"content="width=device-width, initial-scale=1.0">

<title>Water Quality Prediction</title>

<style>

body

{

background-image:url({{url\_for('static', filename='image/background.png')}});

background-repeat: no-repeat;

background-size: cover;

background-attachment: fixed;

}

#ip3 {

border-radius: 15px50px30px5px;

background: rgb(244, 248, 249);

padding: 20px;

width: 200px;

height: 15px;

}

</style>

</head>

<body><center><br><br><br>

    <h1>Efficient of water quality and predication</h1>

    <h2>Using Random Forest</h2><br><h3>- By Pavan Sidharth</h3>

    <center><divclass="header1"><fontcolor="#FF0000"font-family="Fascinate Inline"size

    <br><br>

     <formclass="main"action="/login"method="post">

     <br><br>

     <inputtype="text"name="year"id="ip3"placeholder="Enter Year"/>

    <br><br>

    <inputtype="text"name="do"id="ip3"placeholder="Enter D.0"/>

    <br><br>

    <inputtype="text"name="ph"id="ip3"placeholder="Enter PH"/>

    <br><br>

    <inputtype="text"name="co"id="ip3"placeholder="Enter Conductivity"/>

    <br><br>

    <inputtype="text"name="bod"id="ip3"placeholder="Enter B.O.D"/>

    <br><br>

    <inputtype="text"name="na"id="ip3"placeholder="Enter Nitratenen"/>

    <br><br>

    <inputtype="text"name="tc"id="ip3"placeholder="Enter Total Coliform"/>

    <br><br>

    <inputtype="submit"class="logbtn"value="Predict"style="width: 7%;height: 25px;></center>

    <br><br><br>

    <div class="bor"><center><b><fontcolor="red"size=5>{{showcase}}</font></b></center>

     </form>

</body>

</html>

**7.2 FEATURE 2**

**JUPYTER NOTEBOOK – SOURCE CODE**

**Importing the Libraries**

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

import os

from matplotlib import rcParams

import warnings

warnings.filterwarnings(action='ignore')

warnings.warn('this is a warning!')

# Reading the Dataset

data=pd.read\_csv('water\_dataX.csv')

data

# Data Preprocessing

data.head()

# Analyzing the data

data.describe()

data.info()

data.shape

data.dtypes

data.isnull().any()

data.dtypes

data['Temp']=pd.to\_numeric(data['Temp'],errors='coerce')

data['D.O. (mg/l)']=pd.to\_numeric(data['D.O. (mg/l)'],errors='coerce')

data['PH']=pd.to\_numeric(data['PH'],errors='coerce')

data['B.O.D. (mg/l)']=pd.to\_numeric(data['B.O.D. (mg/l)'],errors='coerce')

data['CONDUCTIVITY (µmhos/cm)']=pd.to\_numeric(data['CONDUCTIVITY (µmhos/cm)'],errors='coerce')

data['NITRATENAN N+ NITRITENANN (mg/l)']=pd.to\_numeric(data['NITRATENAN N+ NITRITENANN (mg/l)'],errors='coerce')

data['TOTAL COLIFORM (MPN/100ml)Mean']=pd.to\_numeric(data['TOTAL COLIFORM (MPN/100ml)Mean'],errors='coerce')

data.dtypes

data.describe()

# Removing the Null values

data.isnull().sum()

data['Temp'].fillna(data['Temp'].mean(),inplace=True)

data['D.O. (mg/l)'].fillna(data['D.O. (mg/l)'].mean(),inplace=True)

data['PH'].fillna(data['PH'].mean(),inplace=True)

data['CONDUCTIVITY (µmhos/cm)'].fillna(data['CONDUCTIVITY (µmhos/cm)'].mean(),inplace=True)

data['B.O.D. (mg/l)'].fillna(data['B.O.D. (mg/l)'].mean(),inplace=True)

data['NITRATENAN N+ NITRITENANN (mg/l)'].fillna(data['NITRATENAN N+ NITRITENANN (mg/l)'].mean(),inplace=True)

data['TOTAL COLIFORM (MPN/100ml)Mean'].fillna(data['TOTAL COLIFORM (MPN/100ml)Mean'].mean(),inplace=True)

data.isnull().sum()

data.drop(['FECAL COLIFORM (MPN/100ml)'],axis=1,inplace=True)

data

# Calculating Water quality Index

data['nph']=data.ph.apply(lambda x:(100 if (8.5>=x>=7)

else(80 if (8.6>=x>=8.5) or (6.9>=x>=6.8)

else(60 if (8.8>=x>=8.6) or (6.8>=x>=6.7)

else(40 if (9>=x>=8.8) or (6.7>=x>=6.5)

else 0)))))

data['ndo']=data.co.apply(lambda x:(100 if (x>=6)

else(80 if (6>=x>=5.1)

else(60 if (5>=x>=4.1)

else(40 if (4>=x>=3)

else 0)))))

data['nco']=data.tc.apply(lambda x:(100 if (5>=x>=0)

else(80 if (50>=x>=5)

else(60 if (500>=x>=50)

else(40 if (10000>=x>=500)

else 0)))))

data['nbdo']=data.bod.apply(lambda x:(100 if (3>=x>=0)

else(80 if (6>=x>=3)

else(60 if (80>=x>=6)

else(40 if (125>=x>=80)

else 0)))))

data['nec']=data.co.apply(lambda x:(100 if (75>=x>=0)

else(80 if (150>=x>=75)

else(60 if (225>=x>=150)

else(40 if (300>=x>=225)

else 0)))))

data['nna']=data.na.apply(lambda x:(100 if (20>=x>=0)

else(80 if (50>=x>=20)

else(60 if (100>=x>=50)

else(40 if (200>=x>=100)

else 0)))))

data['wph']=data.nph\*0.165

data['wdo']=data.ndo\*0.281

data['wbdo']=data.nbdo\*0.234

data['wec']=data.nec\*0.009

data['wna']=data.nna\*0.028

data['wco']=data.nco\*0.281

data['wqi']=data.wph+data.wdo+data.wbdo+data.wec+data.wna+data.wco

data

pd.set\_option('display.max\_columns', None)

data

average=data.groupby('year')['wqi'].mean()

average.head()

# Data visualization

# univariate Analysis

# displot

sns.displot(data.Temp)

plt.show()

sns.displot(data.do)

plt.show()

sns.displot(data.bod)

plt.show()

sns.displot(data.na)

plt.show()

sns.displot(data.year)

plt.show()

# Countplot[¶](http://localhost:8888/notebooks/deployment/Efficient%20Water%20Quality%20Analysis%20and%20Prediction.ipynb" \l "Countplot)

sns.countplot(data.ph)

plt.show()

sns.countplot(data.co)

plt.show()

sns.countplot(data.bod)

plt.show()

sns.countplot(data.ndo)

plt.show()

sns.countplot(data.nna)

plt.show()

# Distplot

sns.distplot(data.tc)

plt.show()

sns.distplot(data.nph)

plt.show()

sns.distplot(data.nco)

plt.show()

sns.distplot(data.nec)

plt.show()

# Pie chart

plt.pie(data.year.value\_counts(),[0.1,0,0,0,0,0,0,0,0,0,0,0],labels=[2012,2013,2014,2011,2010,2009,2008,2007,2005,2006,2003,2004 ],autopct='%1.1f%%')

plt.title('YEAR')

plt.show()

plt.pie(data.wph.value\_counts(),[0,0.2,0,0,0],labels=[16.5,0.0,13.2,6.6,9.9],autopct='%1.1f%%')

plt.title('wph')

plt.show()

plt.pie(data.wec.value\_counts(),labels=[0,0.90,0.72,0.54,0.36],autopct='%1.1f%%')

plt.title('wec')

plt.show()

plt.pie(data.nbdo.value\_counts(),labels=[100,60,80,0,40],autopct='%1.1f%%')

plt.title('nbdo')

plt.show()

plt.pie(data.wco.value\_counts(),labels=[11.24,16.86,0,22.48,28.10],autopct='%1.1f%%')

plt.title('wco')

plt.show()

# Bivariate Analysis

# Line plot

sns.lineplot(data.ph,data.do)

plt.show()

sns.lineplot(data.co,data.bod)

plt.show()

sns.lineplot(data.na,data.tc)

plt.show()

sns.lineplot(data.nph,data.ndo)

plt.show()

sns.lineplot(data.nco,data.nbdo)

plt.show()

sns.lineplot(data.nec,data.nna)

plt.show()

sns.lineplot(data.wph,data.wdo)

plt.show()

sns.lineplot(data.wbdo,data.wec)

plt.show()

sns.lineplot(data.wna,data.wco)

plt.show()

# Scatter plot

sns.scatterplot(data.ph,data.bod)

plt.show()

sns.scatterplot(data.co,data.do)

plt.show()

sns.scatterplot(data.bod,data.na)

plt.show()

sns.scatterplot(data.co,data.tc)

plt.show()

sns.scatterplot(data.nph,data.nbdo)

plt.show()

sns.scatterplot(data.nco,data.ndo)

plt.show()

sns.scatterplot(data.nco,data.nna)

plt.show()

sns.scatterplot(data.nbdo,data.nec)

plt.show()

sns.scatterplot(data.wph,data.wec)

plt.show()

sns.scatterplot(data.wdo,data.wbdo)

plt.show()

sns.scatterplot(data.wbdo,data.wco)

plt.show()

sns.scatterplot(data.wec,data.wna)

plt.show()

# Multivariate analysis

data.hist(figsize=(17,17))

plt.show()

# Label Encoding

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

data.location=le.fit\_transform(data.location)

data.state=le.fit\_transform(data.state)

data.head()

# Finding correlation Matrix Using Heatmap

plt.figure(figsize=(20,20))

sns.heatmap(data.corr(),annot=True)

plt.show()

df=data.drop(['nco','nph','ndo','nbdo','nec','nna','location','state','station','wph','wdo','wbdo','wec','wna','wco','Temp'],axis=1)

df

df.corr().wqi.sort\_values(ascending=False)

# Splitting Dependent and Independent Columns

x=df.iloc[:,0:7]

y=df.wqi

x

y

x.shape

y.shape

# Splitting the Data into Train and Test

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.3,random\_state=0)

x\_train.shape

y\_train.shape

x\_test.shape

y\_test.shape

# Model Building Using Random Forest Regressor

from sklearn.ensemble import RandomForestRegressor

lr3=RandomForestRegressor(max\_depth=7)

lr3.fit(x\_train,y\_train)

rfr\_pred\_test=lr3.predict(x\_test)

rfr\_pred\_test

rfr\_pred\_train=lr3.predict(x\_train)

rfr\_pred\_train

rfr\_test=pd.DataFrame({'Actual\_wqi\_value':y\_test,'Predicted\_wqi\_value':rfr\_pred\_test})

rfr\_test

rfr\_train=pd.DataFrame({'Actual\_wqi\_value':y\_train,'Predicted\_wqi\_value':rfr\_pred\_train})

rfr\_train

# Model Evaluation

from sklearn.metrics import r2\_score

from sklearn.metrics import mean\_squared\_error

#mse(mean squared error)-Random Forest Regression

print(mean\_squared\_error(y\_test,rfr\_pred\_test)) #Random Forest regression test data

print(mean\_squared\_error(y\_train,rfr\_pred\_train)) #Random forest regression train data

#r2\_score(accuracy)-Random Forest Regression

print(r2\_score(y\_test,rfr\_pred\_test)) #Random forest regression test data

print(r2\_score(y\_train,rfr\_pred\_train)) #Random forest regression train data

import pickle

pickle.dump(lr3,open('wqi.pkl','wb'))

model=pickle.load(open('wqi.pkl','rb'))

**FLASK APPLICATION CODE :**

app.py CODE

importnumpyas np

from flask import Flask, render\_template, request

import pickle

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

model = pickle.load(open('wqi.pkl','rb'))

@app.route('/')

def home():

    returnrender\_template("index.html")

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

def login():

    year = request.form["year"]

    do = request.form["do"]

    ph = request.form["ph"]

    co = request.form["co"]

    bod = request.form["bod"]

    na = request.form["na"]

    tc = request.form["tc"]

    total = [[float (do), float (ph), float (co), float (bod), float (na), float(tc), int(year)]]

    y\_pred = model.predict(total)

    y\_pred = y\_pred[[0]]

    if(y\_pred>= 95andy\_pred<= 100) :

        returnrender\_template("index.html", showcase = 'Excellent, The predicted value is '+str(y\_pred))

    elif(y\_pred>= 89and y-pred <= 94) :

        returnrender\_template("index.html", showcase = 'Very good, The predicted value is '+str(y\_pred))

    elif(y\_pred>= 88andy\_pred<= 88) :

        returnrender\_template("index.html", showcase = 'Good, The predicted value is '+str(y\_pred))

    elif(y\_pred>= 65andy\_pred<= 79) :

        returnrender\_template("index.html", showcase = 'Fair, The predicted value is '+str(y\_pred))

    elif(y\_pred>= 45andy\_pred<= 64) :

        returnrender\_template("index.html", showcase = 'Marginal, The predicted value is '+str(y\_pred))

    else :

        returnrender\_template("index.html", showcase = 'Poor, The predicted value is '+str(y\_pred))

if \_\_name\_\_=='\_\_main\_\_' :

    app.run(debug = True,port=8000)

ibm\_app.py CODE

importnumpyas np

from flask import Flask, render\_template, request

import pickle

import requests

# NOTE: you must manually set API\_KEY below using information retrieved from your IBM Cloud account.

API\_KEY = "P-v0uUtXoamzjb6MZFyGXQnh9ql2xObgQaTMWSjkbXJg"

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}

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

model = pickle.load(open('wqi.pkl','rb'))

@app.route('/')

def home():

    returnrender\_template("index.html",)

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

def login():

    year = request.form["year"]

    do = request.form["do"]

    ph = request.form["ph"]

    co = request.form["co"]

    bod = request.form["bod"]

    na = request.form["na"]

    tc = request.form["tc"]

    total = [[float (do), float (ph), float (co), float (bod), float (na), float(tc), int(year)]]

    # NOTE: manually define and pass the array(s) of values to be scored in the next line

    payload\_scoring = {"input\_data": [{"fields": [float (do), float (ph), float (co), float (bod), float (na), float(tc), int(year)], "values": total}]}

    response\_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/823bcd15-d246-4027-ae6d-a984d3e1b053/predictions?version=2022-11-03', json=payload\_scoring,

    headers={'Authorization': 'Bearer ' + mltoken})

    print("Scoring response")

    print(response\_scoring.json())

    predictions=response\_scoring.json()

    predict = int(predictions['predictions'][0]['values'][0][0])

    #print("Final prediction :",predict)

    if(predict >= 95and predict<= 100) :

        returnrender\_template("index.html", showcase = 'Excellent, The predicted value is '+str(predict))

    elif(predict >= 89and predict <= 94) :

        returnrender\_template("index.html", showcase = 'Very good, The predicted value is '+str(predict))

    elif(predict >= 88and predict <= 88) :

        returnrender\_template("index.html", showcase = 'Good, The predicted value is '+str(predict))

    elif(predict >= 65and predict <= 79) :

        returnrender\_template("index.html", showcase = 'Fair, The predicted value is '+str(predict))

    elif(predict >= 45and predict <= 64) :

        returnrender\_template("index.html", showcase = 'Marginal, The predicted value is '+str(predict))

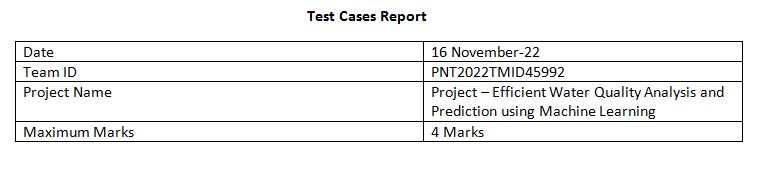
    else :

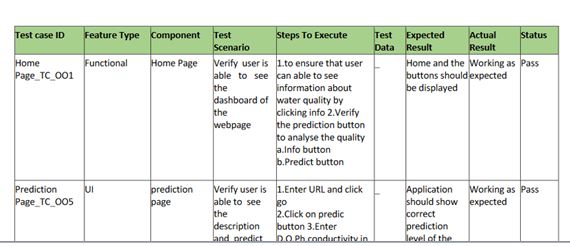
        returnrender\_template("index.html", showcase = 'Poor, The predicted value is '+str(predict))

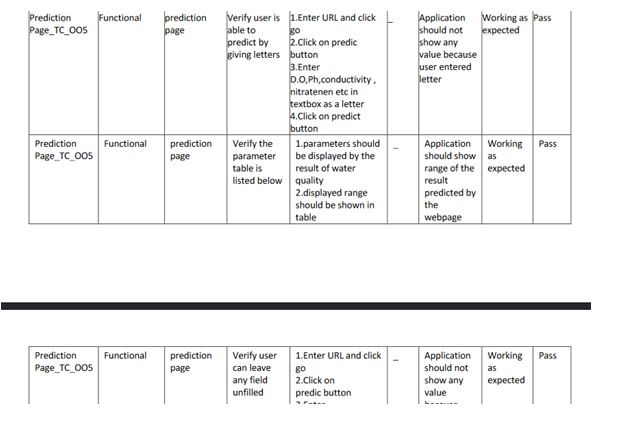
if \_\_name\_\_=='\_\_main\_\_' :

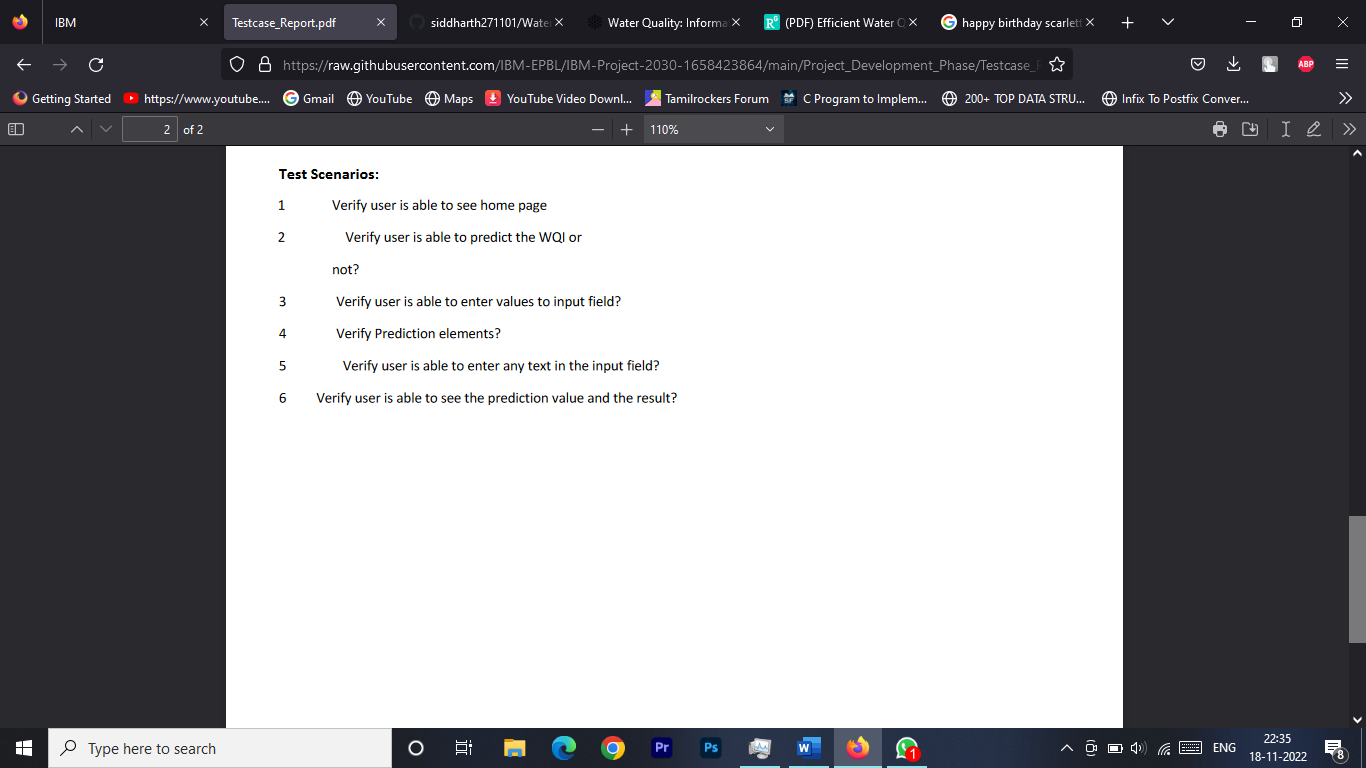
    app.run(debug = True,port=500)

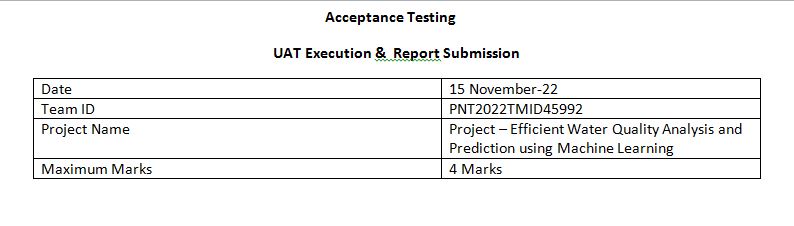
**8. TESTING**

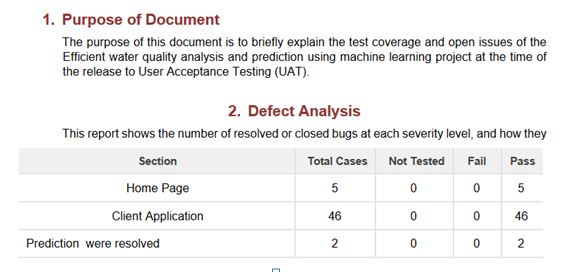


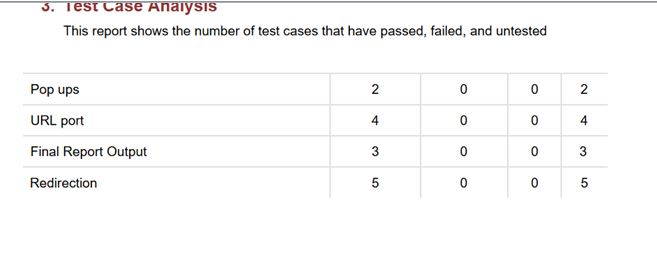
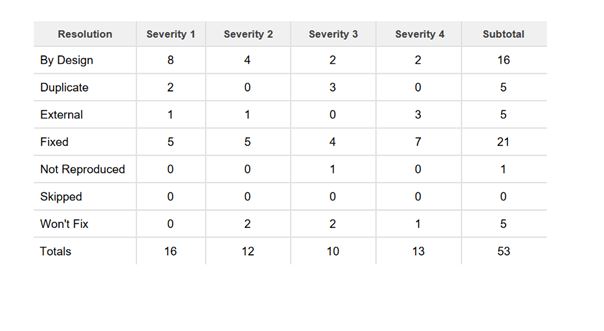
****

****

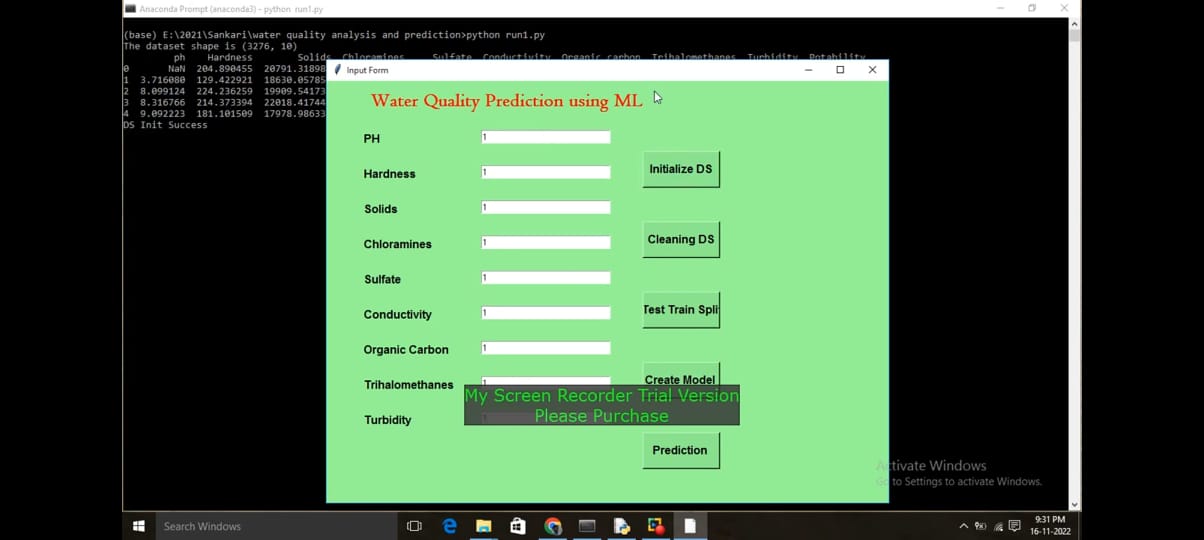
****

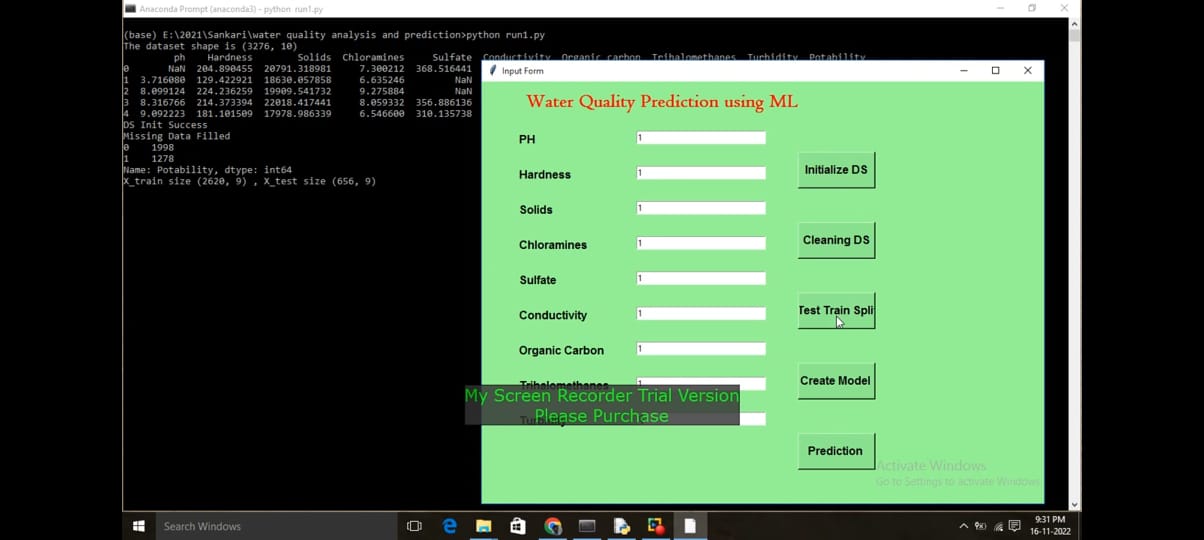
****

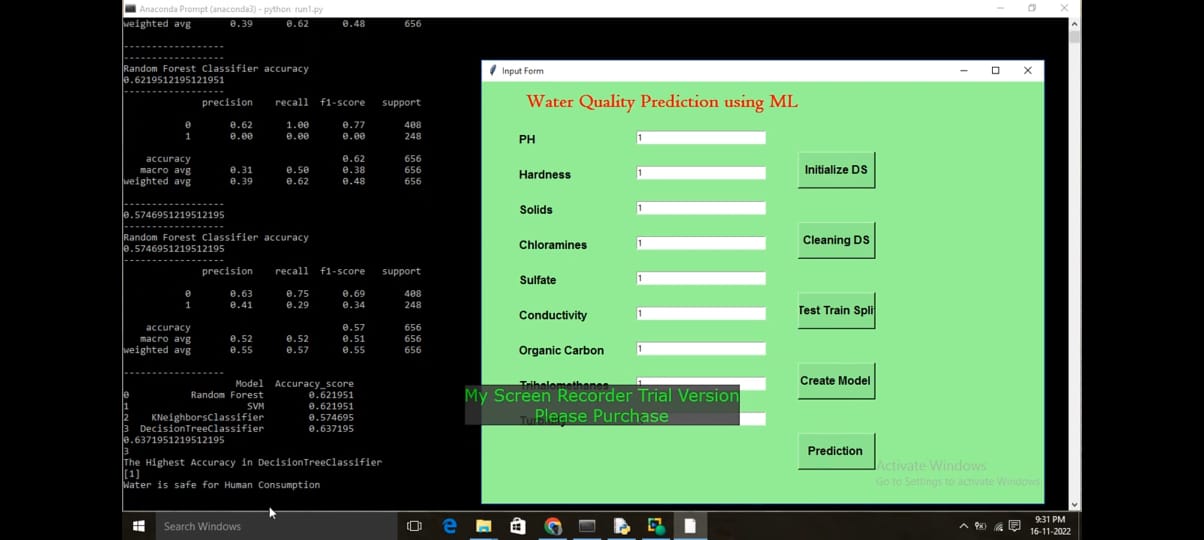
****

****

**9 RESULTS**

****

****

****

**10. ADVANTAGES AND DISADVANTAGES**

**Advantages:**

1. Labor-saving
2. Reduce public health risk.
3. Environment friendly.
4. Friendly UI

**Disadvantages:**

1. Very sensitive to matrix interferences.
2. Limited sensitivity with lighter elements
3. Considerable time/effort requires to run complete sample analysis.
4. Use of x-rays brings up safety concerns

**11. CONCLUSION**

Water is one of the most essential resources for survival and its quality is determined through WQI. Conventionally, to test water quality, one has to go through expensive and cumbersome lab analysis. This research explored an alternative method of machine learning to predict water quality using minimal and easily available water quality parameters. The data used to conduct the study were acquired from PCRWR and contained 663 samples from 12 diﬀerent sources of Rawal Lake, Pakistan. A set of representative supervised machine learning algorithms were employed to estimate WQI. This showed that polynomial regression with a degree of 2, and gradient boosting, with a learning rate of 0.1, outperformed other regression algorithms by predicting WQI most eﬃciently, while MLP with a conﬁguration of (3, 7) outperformed other classiﬁcation algorithms by classifying WQC most eﬃciently

**12. FUTURE SCOPE**

The following are the features that can be added in our application:

In future works, we propose integrating the ﬁndings of this research in a large-scale IoT-based

online monitoring system using only the sensors of the required parameters. The tested algorithms would predict the water quality immediately based on the real-time data fed from the IoT system. The proposed IoT system would employ the parameter sensors of pH, turbidity, temperature anDS for parameter readings and communicate those readings using an Arduino microcontroller and ZigBee transceiver. It would identify poor quality water before it is released for consumption and alert concerned authorities. It will hopefully result in curtailment of people consuming poor quality water and consequently de-escalate harrowing diseases like typhoid and diarrhea. In this regard, the application of a prescriptive analysis from the expected values would lead to future facilities to support decision and policy maker.

**GitHub:**

[**https://github.com/IBM-EPBL/IBM-Project-47065-1660796337.git**](https://github.com/IBM-EPBL/IBM-Project-47065-1660796337.git)

**Project Demo Link:**

[**https://youtu.be/QkFzaRJXxls**](https://youtu.be/QkFzaRJXxls)