PROJECT REPORT

Efficient Water Quality Analysis and Prediction using Machine Learning

Submitted by

PNT2022TMID23264

Vidhya Lakshmi K	**913119205052**
Priyadharshini J	**913119205030**
Priyanga P	**913119205031**
Jeevitha Sai G	**913119205014**

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 PROJECT OVERVIEW

Water is the most important source for sustaining all kinds of life. Natural water resources and aquifers are being polluted due to indiscriminate urbanization and industrialization; as a result, it may be contaminated with physical, chemical, and biological impurities. As reported, 80% of the diseases are water borne diseases. Several criteria are used to measure the quality of water, including the quantity of salt (or salinity), bacteria levels, the percentage of dissolved oxygen or the amount of particles suspended in the water (turbidity). Good water quality implies that harmful substances (pollutants) are absent from the water, and needed substances (oxygen, nutrients) are present. The traditional and common estimation of water quality has been Laboratory analysis which is time consuming and not very practical. This method can be processed efficiently by applying machine learning algorithms and big data tools. Machine learning (ML) is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks.

Machine learning (ML) is a topic of study focused on analyzing and developing "learning" methods, or methods that use data to enhance performance on a certain set of tasks. With the use of machine learning (ML), which is a form of artificial intelligence (AI), software programs can predict outcomes more accurately without having to be explicitly instructed to do so. In order to forecast new output values, machine learning algorithms use historical data as input. A data analysis technique called machine learning automates the creation of analytical models.

1.2 PURPOSE

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. The purpose of this project is to Predict Water Quality by considering all water quality standard indicators.

CHAPTER 2
LITERATURE SURVEY

2.1 EXISTING PROBLEM

For testing the water quality we have to conduct lab tests on the water which is costly and time-consuming as well. So, in this paper, we propose an alternative approach using artificial intelligence to predict water quality. This method uses a significant and easily available water quality index which is set by the WHO(World Health Organisation). The data taken in this paper is taken from the PCPB India which includes 3277 examples of the distinct wellspring. In this paper, WQI(Water Quality Index) is calculated using AI techniques. So in future work, we can integrate this with an IoT based framework to study large datasets and to expand our study to a larger scale. By using that it can predict the water quality fast and more accurately than any other IoT framework. That IoT framework system uses some limits for the sensor to check the parameters like ph, Temperature, Turbidity, and so on. And further after reading this parameter pass these readings to the Arduino microcontroller and ZigBee handset for further prediction.

2.2 REFERENCES

Water quality prediction using machine learning methods

Amir Hamzeh Haghiabi,Ali Heidar Nasrolahi,Abbas Parsaie

The study of water quality of rivers is a common theme in earth sciences. To evaluate the quality of rivers two approaches are considered, including measuring the water quality components and defining the mechanism of pollution transmission. Among water quality components, measuring the dissolved oxygen (DO), chemical oxygen demand (COD), biochemical oxygen demand (BOD), electrical conductivity (EC), pH,temperature, K, Na, Mg, etc. have been proposed.They stated that for developing the ANN, some steps should be considered to reduce the trial and error process. They stated that for the initial design of ANN model, after dataset division, in the first step one hidden layer consisting of numbers of neurons equal to input features is considered. At this stage, the performance of different transfer functions is evaluated and the best ones are chosen. In the next step, the size of the network is modified to improve the precision of the developed model. The last two stages of this approach are also applicable to the design of SVM.

Machine learning algorithms for efficient water quality Prediction

Mourade Azrour,Jamal Mabrouki,Ghizlane Fattah,Azedine Guezzaz, Faissal Aziz

In this study, we take the advantages of machine learning algorithms to develop a model that is capable of predicting the water quality index and then the water quality class. The method They propose is based on four water parameters: temperature, pH, turbidity and coliforms. The use of the multiple regression algorithms has proven to be important and effective in predicting the water quality index.The method they propose is based on four water parameters: temperature, pH, turbidity and coliforms. The use of the multiple regression algorithms has proven to be important and effective in predicting the water quality index. In addition, the adoption of the artificial neural network provides the most highly efficient way to classify the water quality.

Predicting and analyzing water quality using Machine Learning: A comprehensive model

Yafra Khan, Chai Soo See

The goal of this study is to develop a water quality prediction model with the help of water quality factors using Artificial Neural Network (ANN) and time-series analysis. For this paper, the data includes the measurements of 4 parameters which affect and influence water quality. For the purpose of evaluating the performance of model, the performance evaluation measures used are Mean-Squared Error (MSE), Root MeanSquared Error (RMSE) and Regression Analysis.

Efficient Water Quality Prediction Using Supervised Machine Learning (2019)

Umair Ahmed, Rafia Mumtaz,Hirra Anwar,Asad A. Shah,Rabia Irfan,Jose García-Nieto

This research explores a series of supervised machine learning algorithms to estimate the water quality index (WQI), which is a singular index to describe the general quality of water, and the water quality class (WQC), which is a distinctive class defined on the basis of the WQI. The proposed methodology employs four input parameters, namely, temperature, turbidity, pH and total dissolved solids. The proposed methodology achieves reasonable accuracy using a minimal number of parameters to validate the possibility of its use in real time water quality detection systems

Water quality analysis using ML (2021)

Manya Kakkar , Vansh Gupta , Jai Garg , Dr. Surender Dhiman

This research explores the Machine Learning (ML) algorithms for comparing AutoML and an expert architecture built by the authors for Water Quality Assessment to evaluate the Water

Quality Index, which gives the general water quality, and the Water Quality Class, a term classified on the basis of the Water Quality Index.

Detection of Water Quality using Machine Learning (2021)

Manya Kakkar , Vansh Gupta , Jai Garg , Dr. Surender Dhiman

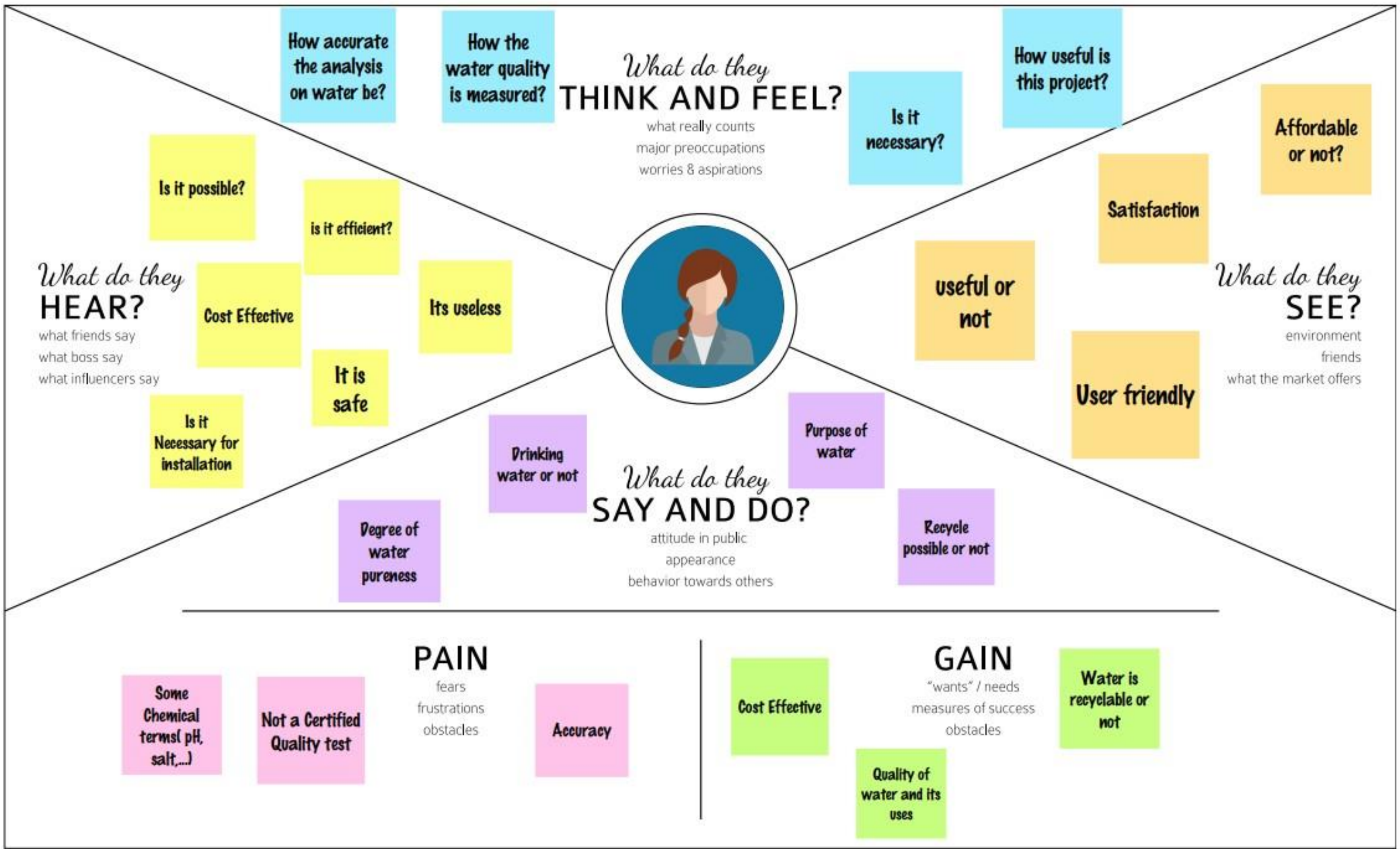
The system makes use of IoT and Machine Learning technology. It consists of physical and chemical sensors that detect pH, Turbidity, Color, Dissolved Oxygen, Conductivity to check influencing factors. The data collected by the sensors is saved in a database and then submitted for analysis. The neural network method is used to forecast the outcome. It is employed in order to generate a non-linear connection for projected output. When any of the parameters falls below the standard values, the system sends an alarm notification to the user. This enables the user to be aware of water pollution in their home tanks ahead of time.

2.3 PROBLEM STATEMENT DEFINITION

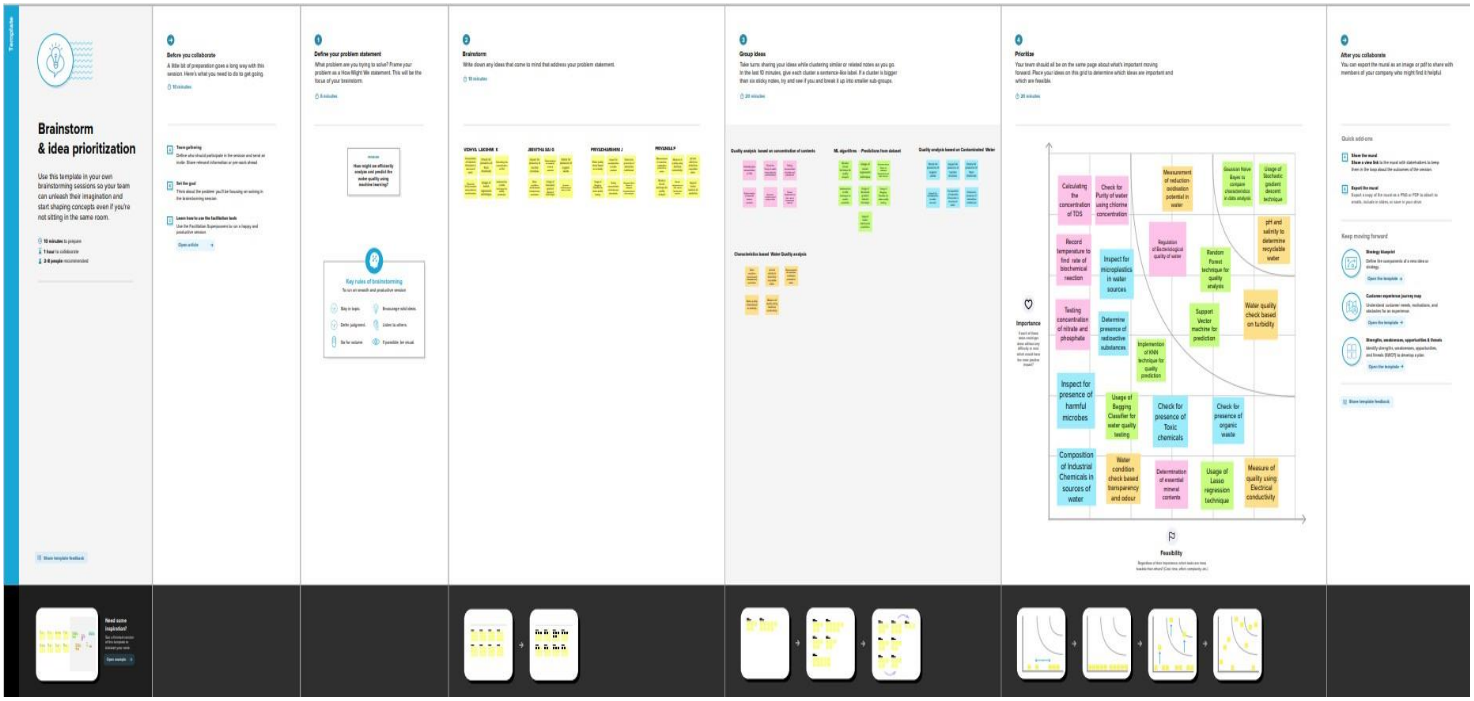
Water is considered as a vital resource that affects various aspects of human health and lives. The quality of water is a major concern for people living in urban areas. The quality of water serves as a powerful environmental determinant and a foundation for the prevention and control of waterborne diseases. 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.

CHAPTER 3
IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION & BRAINSTORMING



Template

Brainstorm
& idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.

Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

Open article

1 Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

How might we efficiently analyze and predict the water quality using machine learning?

Key rules of brainstorming

To run an smooth and productive session

Stay in topic.
Encourage wild ideas.
Defer judgment.
Listen to others.
Go for volume.
If possible, be visual.

2 Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

VIDHYA LAKSHMI E
JEEVITHA SAI G
PRIYADHARSHINI J
PRIYANGA P

3 Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

Quality analysis based on concentration of contents

ML algorithms - Predictions from dataset

Quality analysis based on Contaminated Water

Characteristics based Water Quality analysis

4 Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

Calculating the concentration of TDS

Check for Purity of water using chlorine concentration

Measurement of reduction-oxidisation potential in water

Gaussian Naive Bayes to compare characteristics in data analysis

Usage of Stochastic gradient descent technique

Record temperature to find rate of biochemical reaction

Inspect for microplastics in water sources

Regulation of Bacteriological quality of water

Random Forest technique for quality analysis

pH and salinity to determine recyclable water

Testing concentration of nitrate and phosphate

Determine presence of radioactive substances

Implemention of KNN technique for quality prediction

Support Vector machine for prediction

Water quality check based on turbidity

Inspect for presence of harmful microbes

Usage of Bagging Classifier for water quality testing

Check for presence of Toxic chemicals

Check for presence of organic waste

Composition of Industrial Chemicals in sources of water

Water condition check based transparency and odour

Determination of essential mineral contents

Usage of Lasso regression technique

Measure of quality using Electrical conductivity

Importance

Feasibility

After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

Share the mural

Export the mural

Keep moving forward

Strategy blueprint

Customer experience journey map

Strengths, weaknesses, opportunities & threats

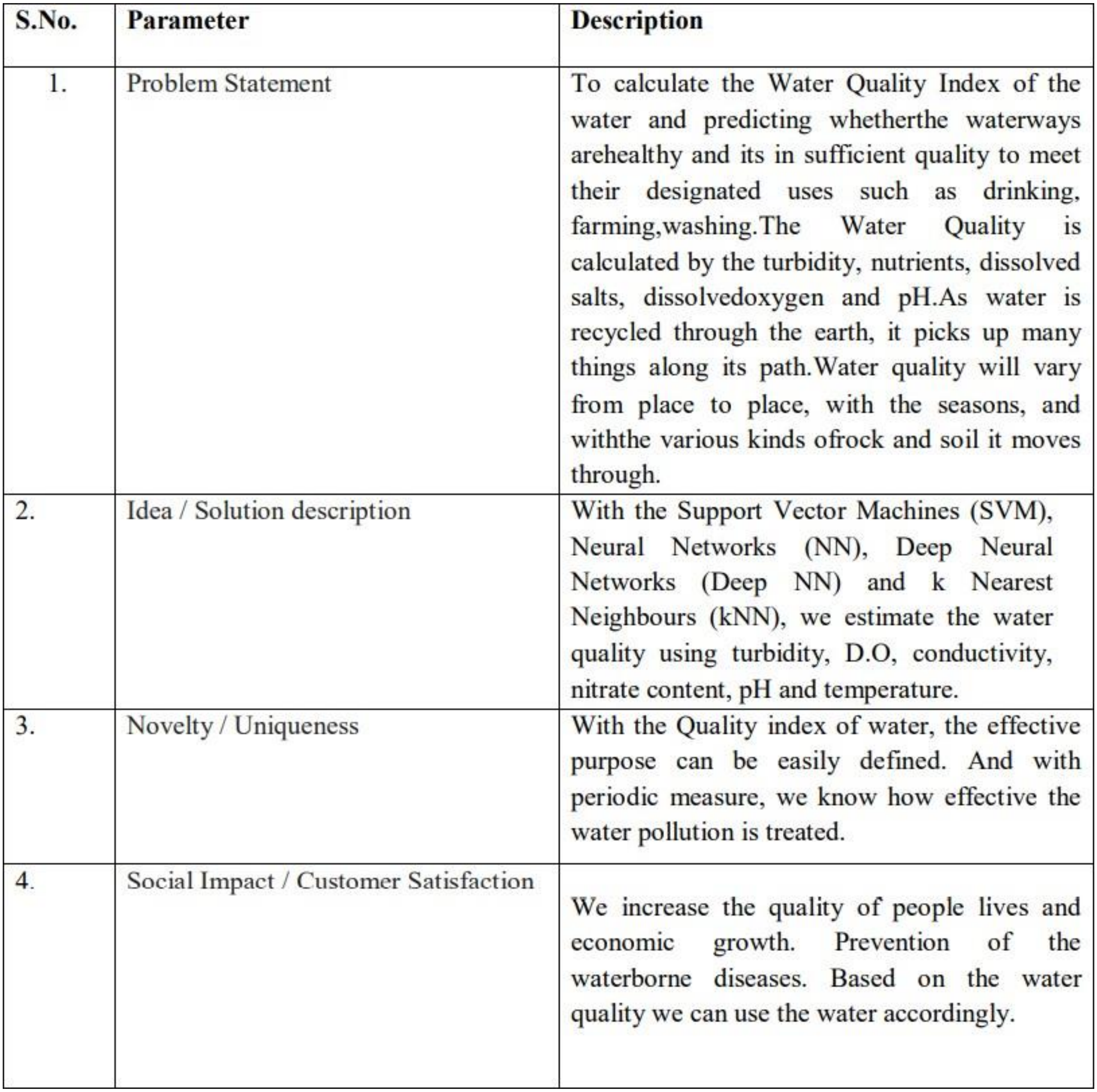
Open the template

Share template feedback

Need some inspiration?

Open example

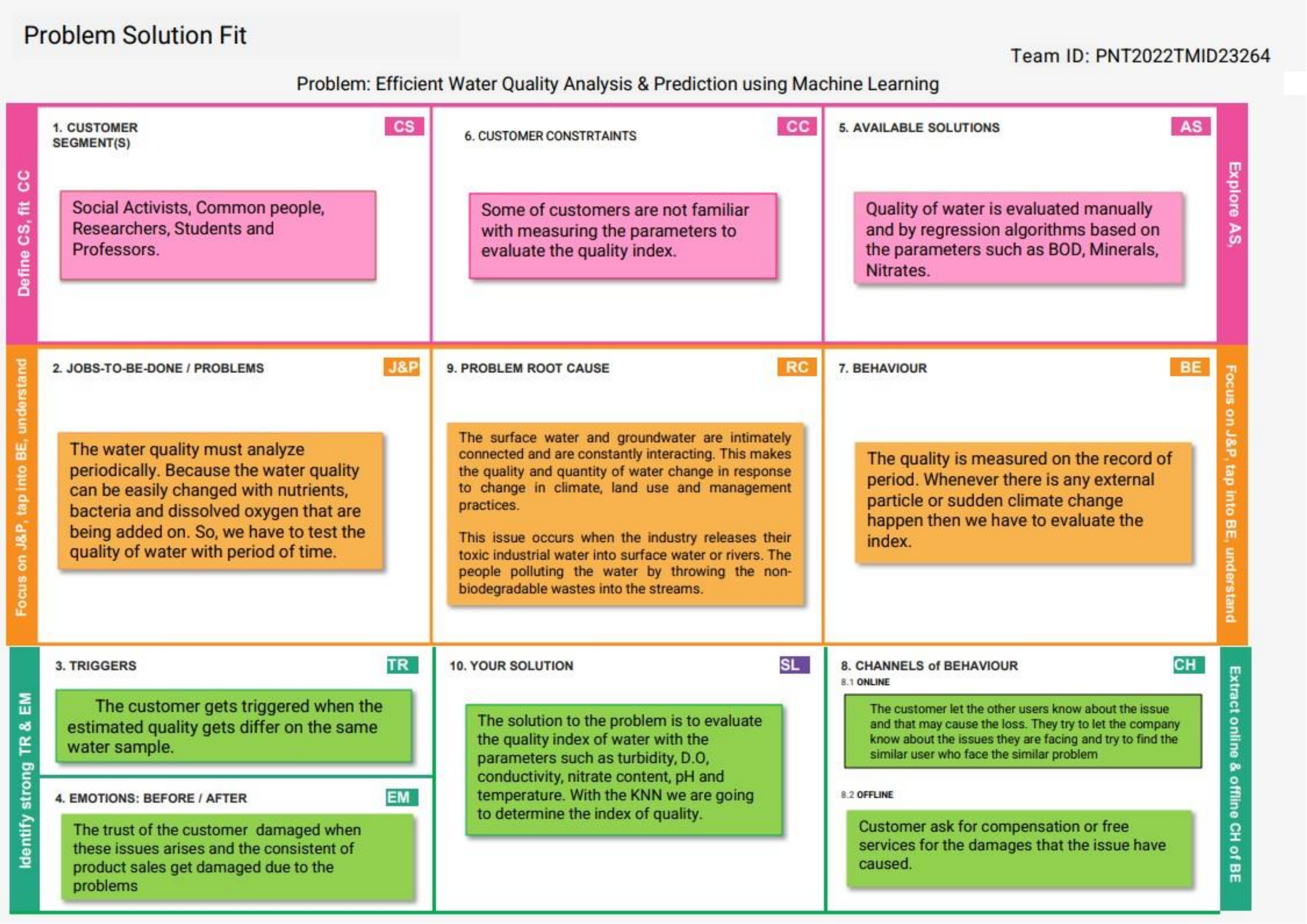
3.3 PROPOSED SOLUTION



S.No.	Parameter	Description
1.	Problem Statement	To calculate the Water Quality Index of the water and predicting whetherthe waterways arehealthy and its in sufficient quality to meet their designated uses such as drinking, farming,washing.The Water Quality is calculated by the turbidity, nutrients, dissolved salts, dissolvedoxygen and pH.As water is recycled through the earth, it picks up many things along its path.Water quality will vary from place to place, with the seasons, and withthe various kinds ofrock and soil it moves through.
2.	Idea / Solution description	With the Support Vector Machines (SVM), Neural Networks (NN), Deep Neural Networks (Deep NN) and k Nearest Neighbours (kNN), we estimate the water quality using turbidity, D.O, conductivity, nitrate content, pH and temperature.
3.	Novelty / Uniqueness	With the Quality index of water, the effective purpose can be easily defined. And with periodic measure, we know how effective the water pollution is treated.
4.	Social Impact / Customer Satisfaction	We increase the quality of people lives and economic growth. Prevention of the waterborne diseases. Based on the water quality we can use the water accordingly.

5.	Business Model (Revenue Model)	With this model, we can evaluate the quality and major purpose of water based on the quality of water.
6.	Scalability of the Solution	The water quality index is changed in response to change in climate, land use and management practices. By setting the criteria for water quality index calculation, we can overcome this issue.

3.4 PROBLEM SOLUTION FIT



Problem Solution Fit

Team ID: PNT2022TMID23264

Problem: Efficient Water Quality Analysis & Prediction using Machine Learning

Define CS, fit CC

1. CUSTOMER SEGMENT(S) CS

Social Activists, Common people, Researchers, Students and Professors.

6. CUSTOMER CONSTRTAINTS CC

Some of customers are not familiar with measuring the parameters to evaluate the quality index.

5. AVAILABLE SOLUTIONS AS

Quality of water is evaluated manually and by regression algorithms based on the parameters such as BOD, Minerals, Nitrates.

Explore AS,

Focus on J&P, tap into BE, understand

2. JOBS-TO-BE-DONE / PROBLEMS J&P

The water quality must analyze periodically. Because the water quality can be easily changed with nutrients, bacteria and dissolved oxygen that are being added on. So, we have to test the quality of water with period of time.

9. PROBLEM ROOT CAUSE RC

The surface water and groundwater are intimately connected and are constantly interacting. This makes the quality and quantity of water change in response to change in climate, land use and management practices.

This issue occurs when the industry releases their toxic industrial water into surface water or rivers. The people polluting the water by throwing the non-biodegradable wastes into the streams.

7. BEHAVIOUR BE

The quality is measured on the record of period. Whenever there is any external particle or sudden climate change happen then we have to evaluate the index.

Focus on J&P, tap into BE, understand

Identify strong TR & EM

3. TRIGGERS TR

The customer gets triggered when the estimated quality gets differ on the same water sample.

4. EMOTIONS: BEFORE / AFTER EM

The trust of the customer damaged when these issues arises and the consistent of product sales get damaged due to the problems

10. YOUR SOLUTION SL

The solution to the problem is to evaluate the quality index of water with the parameters such as turbidity, D.O, conductivity, nitrate content, pH and temperature. With the KNN we are going to determine the index of quality.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

The customer let the other users know about the issue and that may cause the loss. They try to let the company know about the issues they are facing and try to find the similar user who face the similar problem

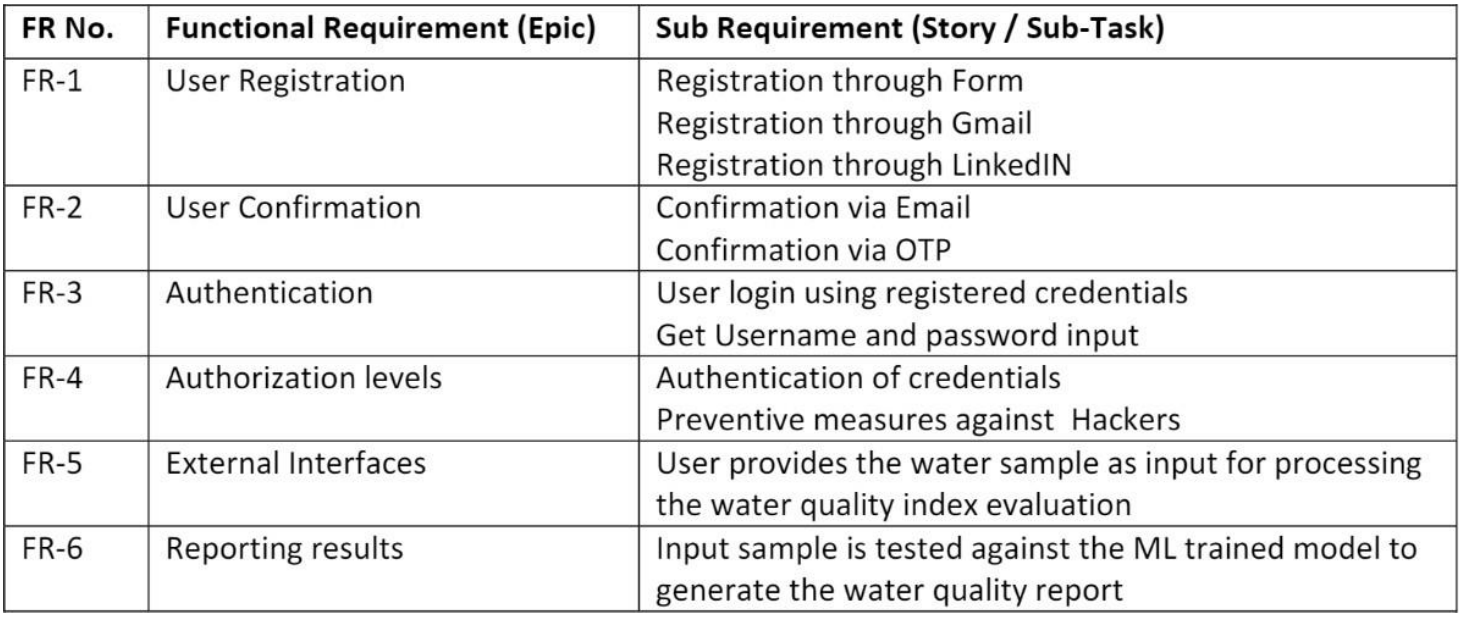
8.2 OFFLINE

Customer ask for compensation or free services for the damages that the issue have caused.

Extract online & offline CH of BE

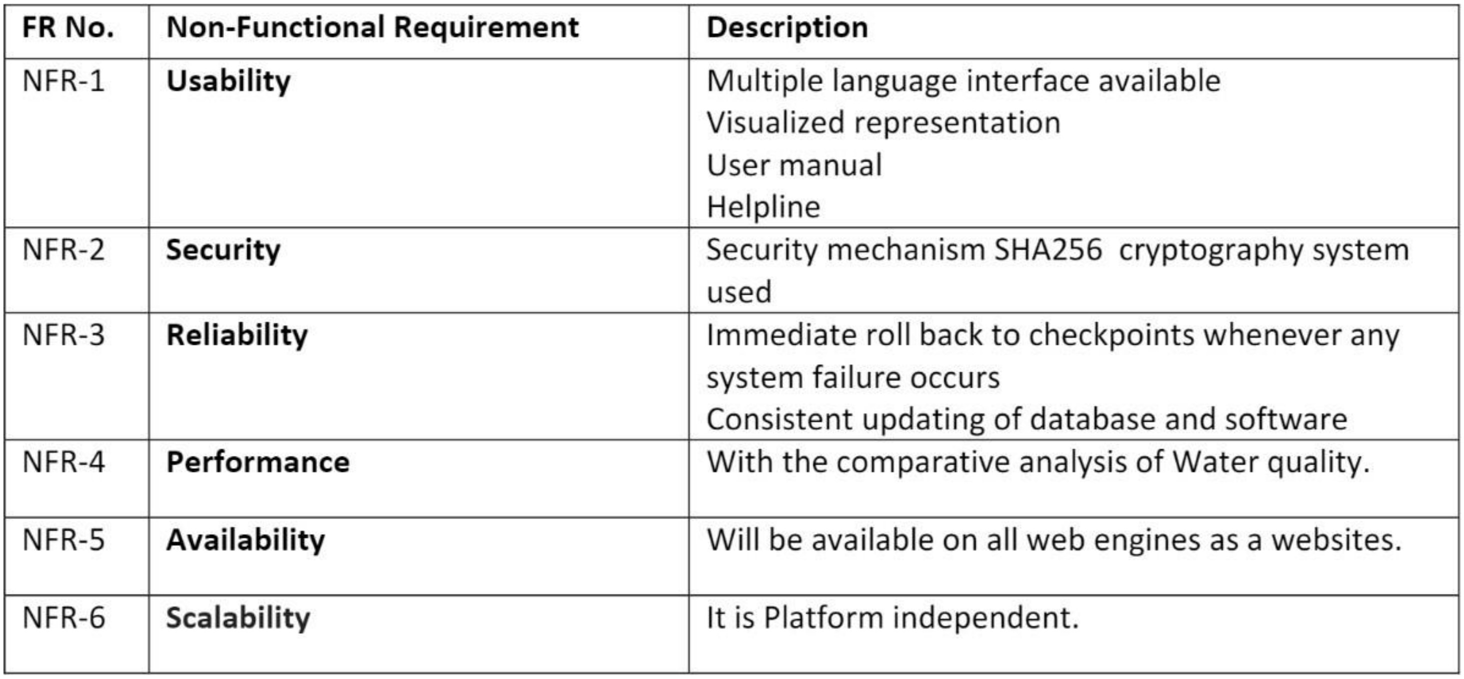
CHAPTER 4
REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS



FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Authentication	User login using registered credentials Get Username and password input
FR-4	Authorization levels	Authentication of credentials Preventive measures against Hackers
FR-5	External Interfaces	User provides the water sample as input for processing the water quality index evaluation
FR-6	Reporting results	Input sample is tested against the ML trained model to generate the water quality report

4.2 NON FUNCTIONAL REQUIREMENTS

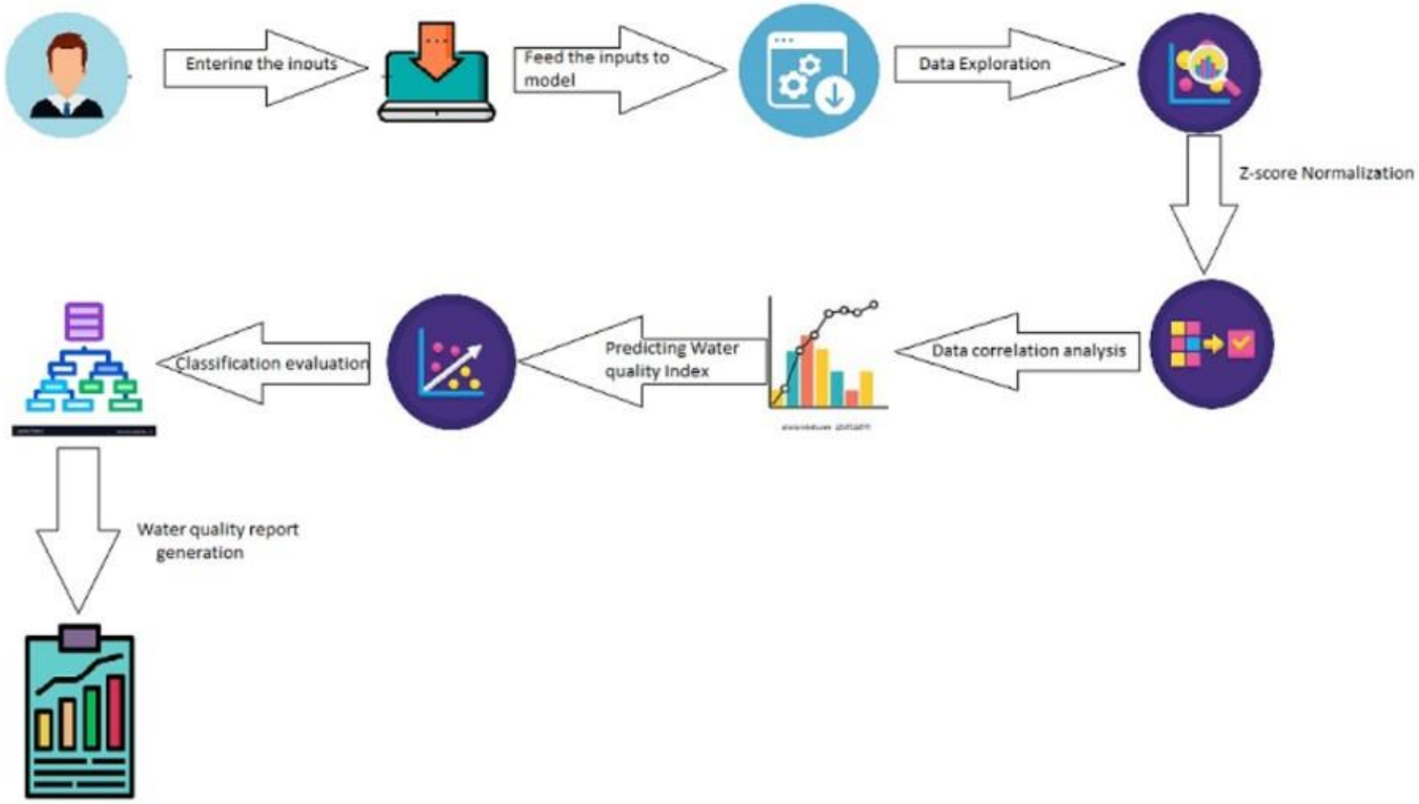


FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Multiple language interface available Visualized representation User manual Helpline
NFR-2	**Security**	Security mechanism SHA256 cryptography system used
NFR-3	**Reliability**	Immediate roll back to checkpoints whenever any system failure occurs Consistent updating of database and software
NFR-4	**Performance**	With the comparative analysis of Water quality.
NFR-5	**Availability**	Will be available on all web engines as a websites.
NFR-6	**Scalability**	It is Platform independent.

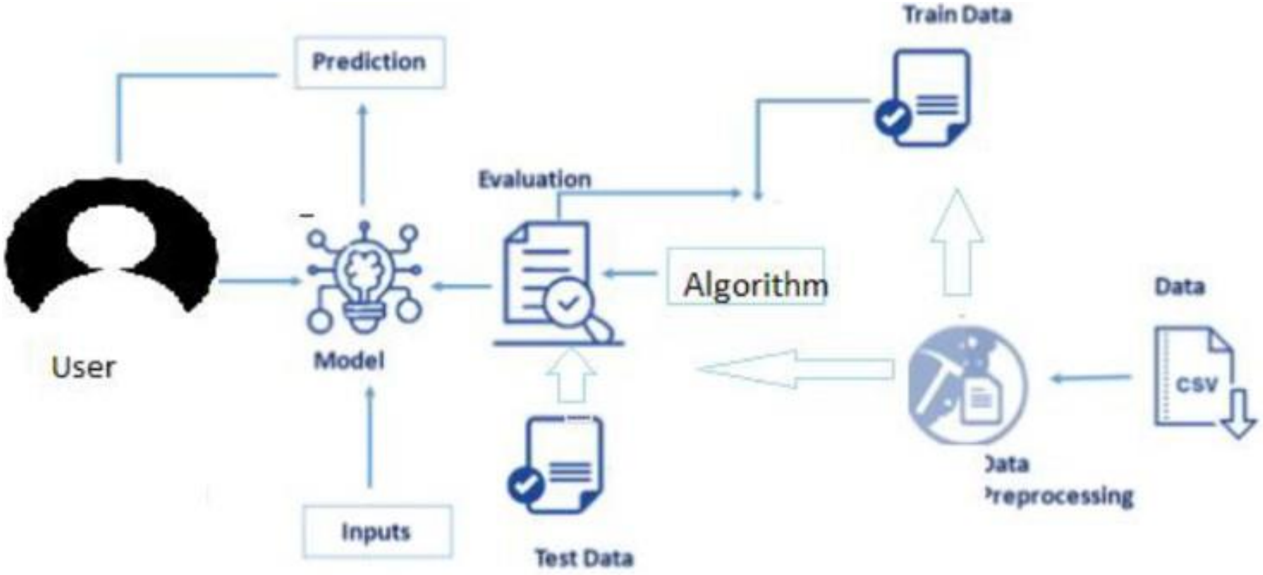
CHAPTER 5

PROJECT DESIGN

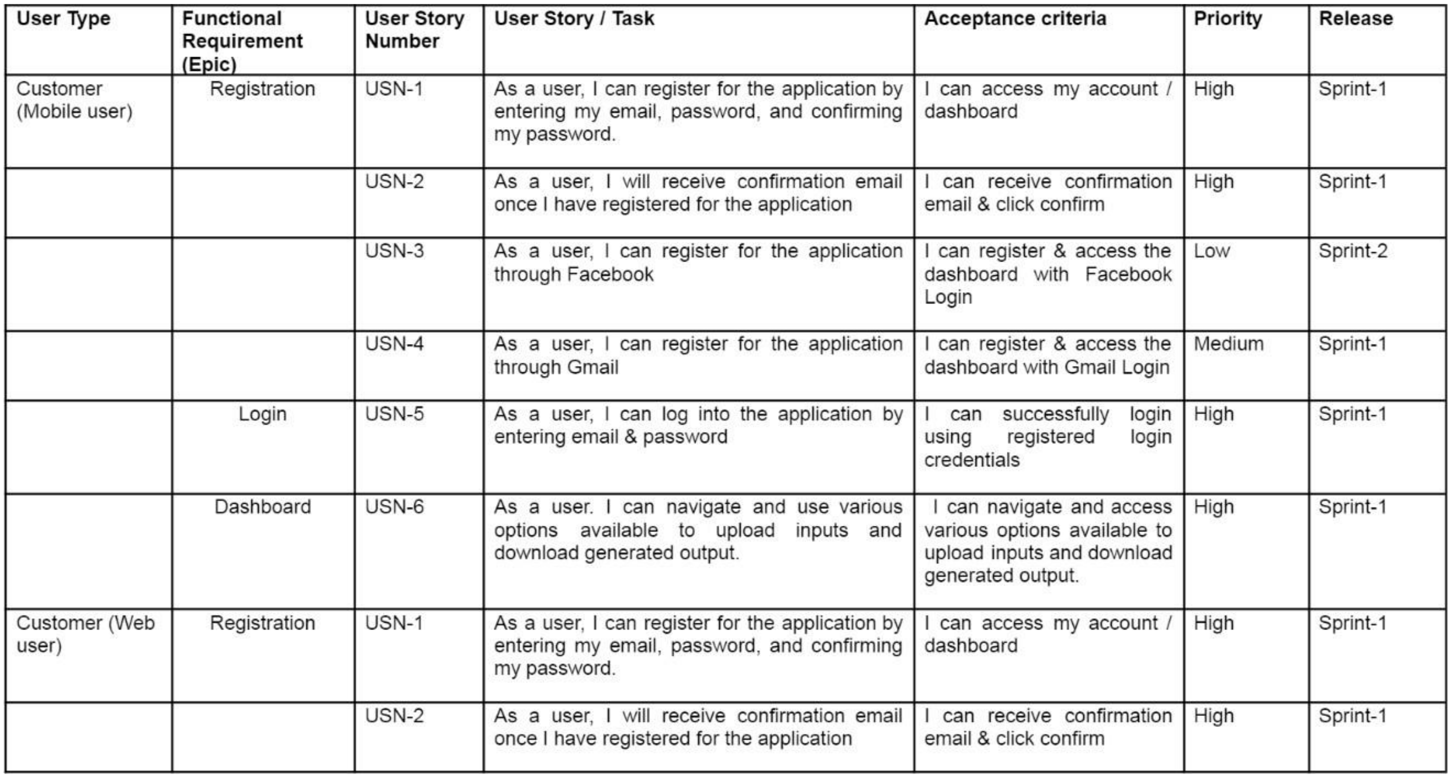
5.1 DATA FLOW DIAGRAM


Entering the inputs
Feed the inputs to model
Data Exploration
Z-score Normalization
Data correlation analysis
Predicting Water quality Index
Classification evaluation
Water quality report generation

5.2 SOLUTION & TECHNICAL ARCHITECTURE



5.3 USER STORIES

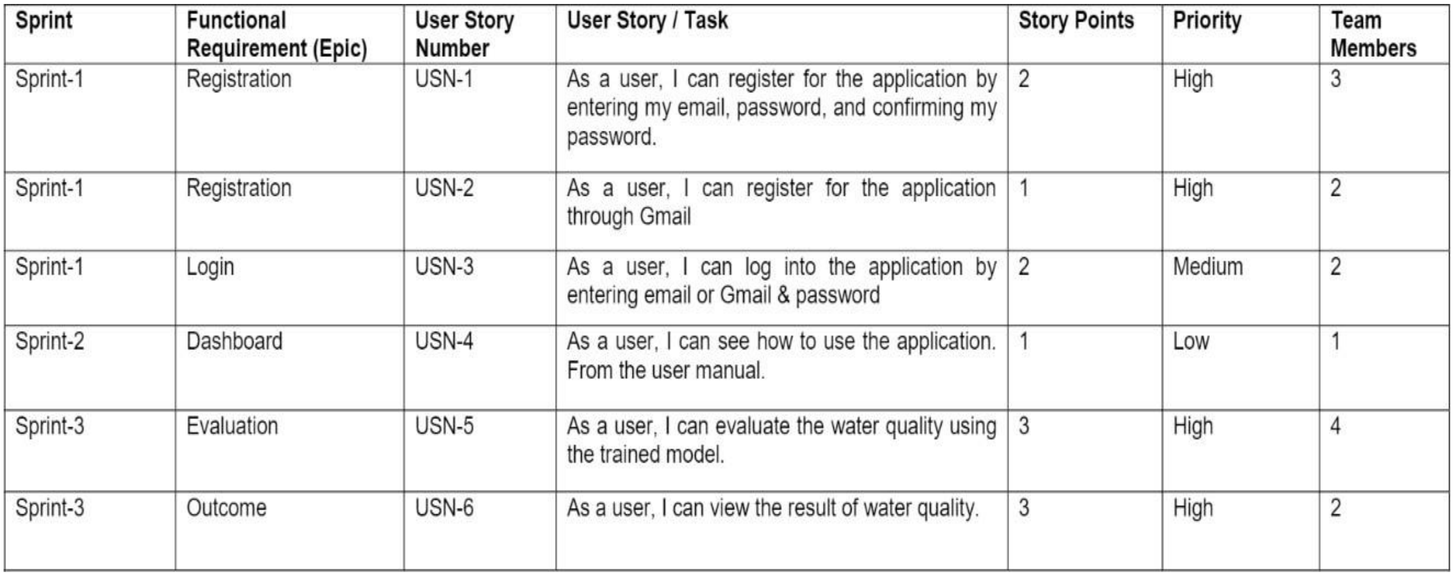


User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	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 Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register & access the dashboard with Gmail Login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can successfully login using registered login credentials	High	Sprint-1
	Dashboard	USN-6	As a user. I can navigate and use various options available to upload inputs and download generated output.	I can navigate and access various options available to upload inputs and download generated output.	High	Sprint-1
Customer (Web user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my 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

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register & access the dashboard with Gmail Login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can successfully login using registered login credentials	High	Sprint-1
	Dashboard	USN-6	As a user. I can navigate and use various options available to upload inputs and download generated output.	I can navigate and access various options available to upload inputs and download generated output.	High	Sprint-1
Customer Care Executive	Support Helpline	USN-1	Customer support is provided to resolve issues within 48 hrs	As a customer care executive, I can provide efficient support to user complaints raised as soon as possible.	High	Sprint-1
Administrator	Management	USN-1	Management of web UI for smooth working on the user side.	As an administrator, I have control to organize and manage the working of Web UI.	High	Sprint-1
	Security	USN-2	As an administrator, security mechanisms such as SHA256 are implemented to secure the website.	I have enough control over the security system of Web UI	High	Sprint-1

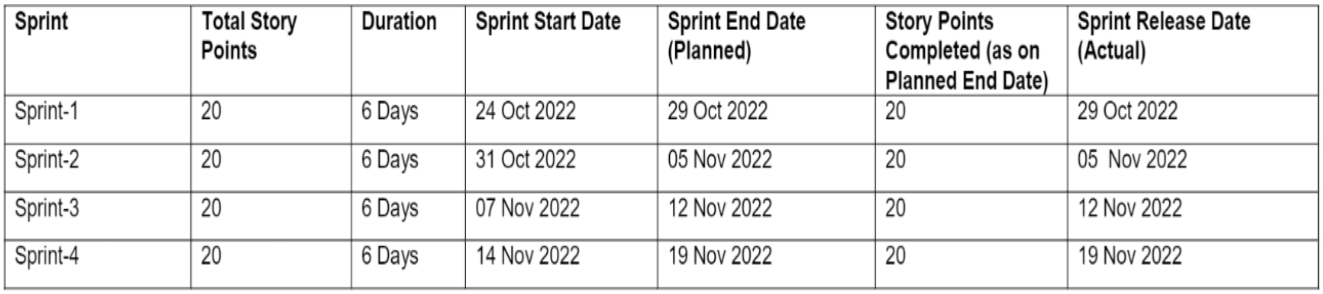
CHAPTER 6
PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

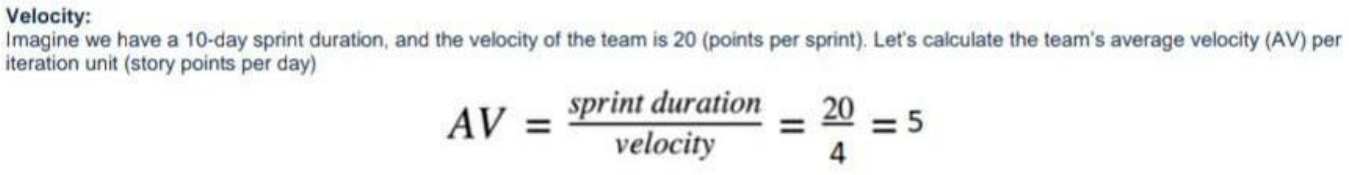


Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	3
Sprint-1	Registration	USN-2	As a user, I can register for the application through Gmail	1	High	2
Sprint-1	Login	USN-3	As a user, I can log into the application by entering email or Gmail & password	2	Medium	2
Sprint-2	Dashboard	USN-4	As a user, I can see how to use the application. From the user manual.	1	Low	1
Sprint-3	Evaluation	USN-5	As a user, I can evaluate the water quality using the trained model.	3	High	4
Sprint-3	Outcome	USN-6	As a user, I can view the result of water quality.	3	High	2

6.2 SPRINT DELIVERY SCHEDULE



Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	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

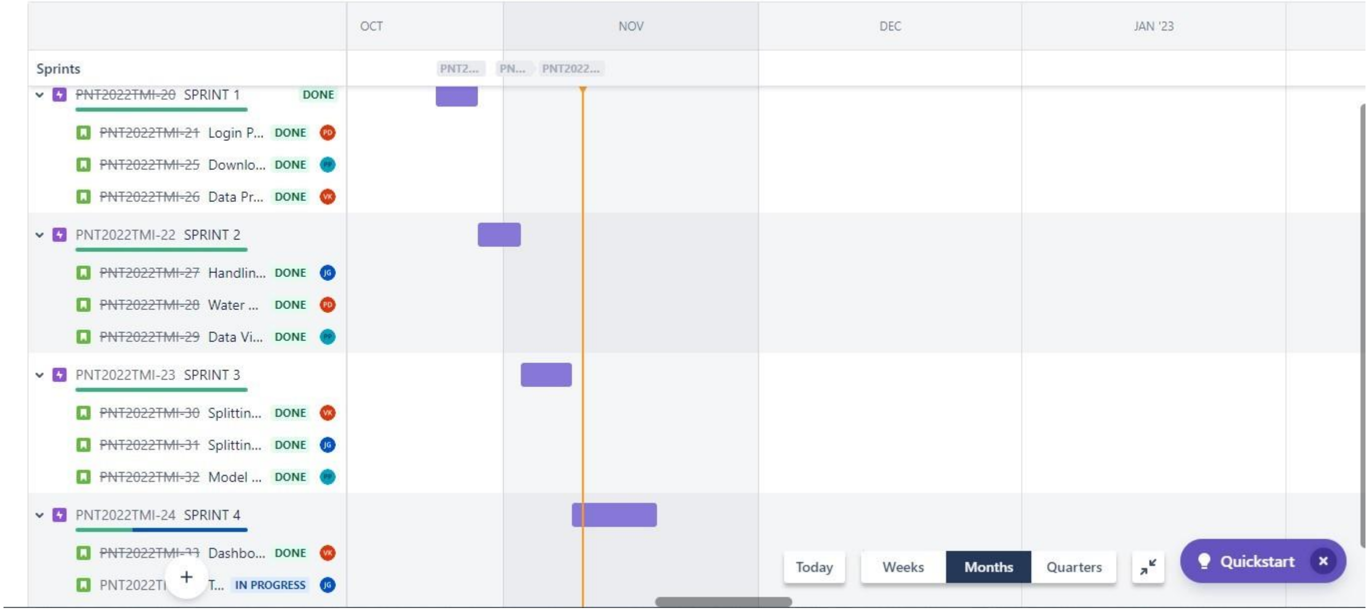


Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{4} = 5$$

6.3 REPORTS FROM JIRA



Sprints

~~PNT2022TMI-20~~ SPRINT 1 DONE

~~PNT2022TMI-21~~ Login P... DONE PD

~~PNT2022TMI-25~~ Downlo... DONE PP

~~PNT2022TMI-26~~ Data Pr... DONE VK

PNT2022TMI-22 SPRINT 2

~~PNT2022TMI-27~~ Handlin... DONE JG

~~PNT2022TMI-28~~ Water ... DONE PD

~~PNT2022TMI-29~~ Data Vi... DONE PP

PNT2022TMI-23 SPRINT 3

~~PNT2022TMI-30~~ Splittin... DONE VK

~~PNT2022TMI-31~~ Splittin... DONE JG

~~PNT2022TMI-32~~ Model ... DONE PP

PNT2022TMI-24 SPRINT 4

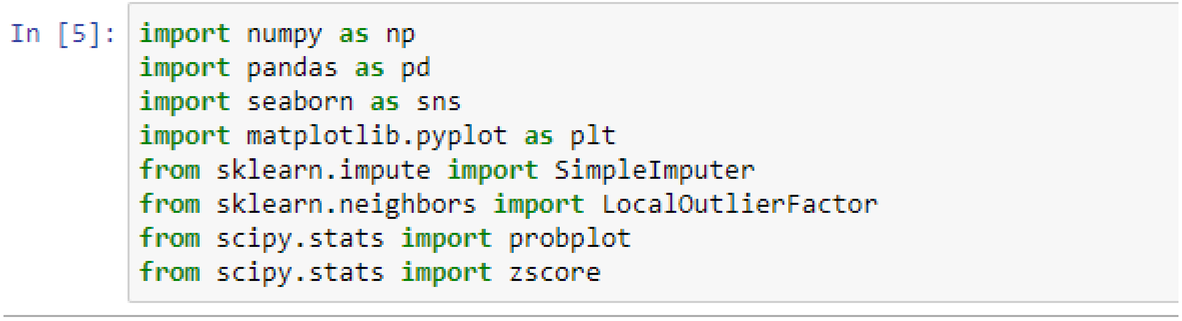
~~PNT2022TMI-33~~ Dashbo... DONE VK

PNT2022TI T... IN PROGRESS JG

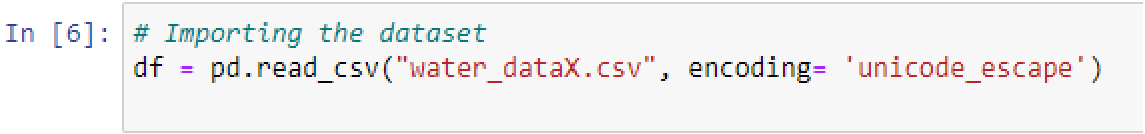
CHAPTER 7

CODING & SOLUTIONING

Importing Libraries and Dataset

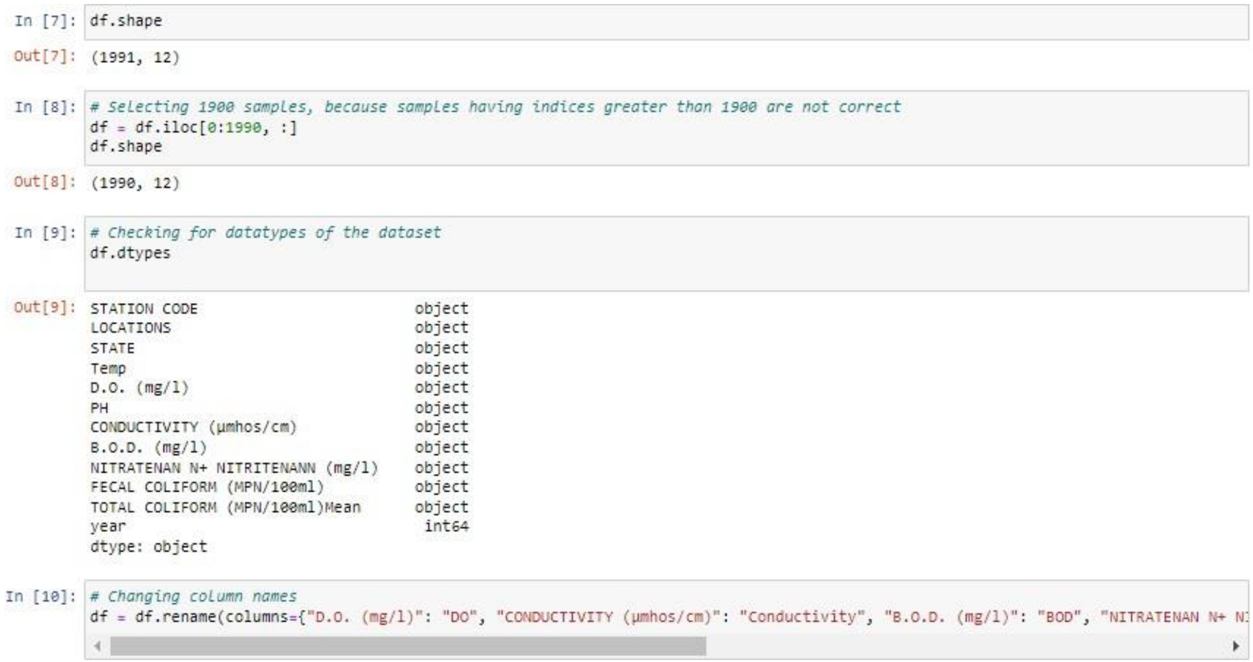


```
In [5]: import numpy as np
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        from sklearn.impute import SimpleImputer
        from sklearn.neighbors import LocalOutlierFactor
        from scipy.stats import probplot
        from scipy.stats import zscore
```

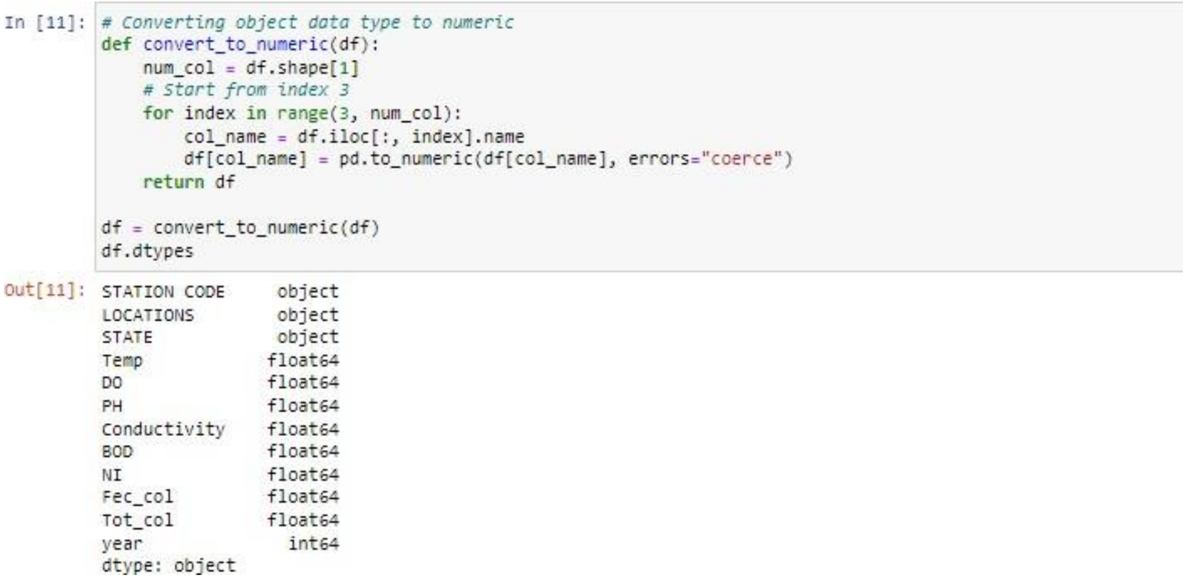


```
In [6]: # Importing the dataset
        df = pd.read_csv("water_dataX.csv", encoding= 'unicode_escape')
```

Data Preprocessing

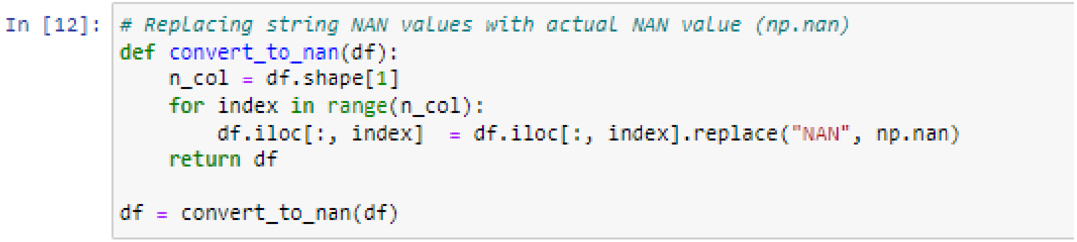


```
In [7]: df.shape

Out[7]: (1991, 12)

In [8]: # Selecting 1900 samples, because samples having indices greater than 1900 are not correct
        df = df.iloc[0:1990, :]
        df.shape

Out[8]: (1990, 12)

In [9]: # Checking for datatypes of the dataset
        df.dtypes

Out[9]: STATION CODE                             object
        LOCATIONS                                object
        STATE                                    object
        Temp                                     object
        D.O. (mg/l)                              object
        PH                                       object
        CONDUCTIVITY (µmhos/cm)                  object
        B.O.D. (mg/l)                            object
        NITRATENAN N+ NITRITENANN (mg/l)         object
        FECAL COLIFORM (MPN/100ml)               object
        TOTAL COLIFORM (MPN/100ml)Mean           object
        year                                      int64
        dtype: object

In [10]: # Changing column names
         df = df.rename(columns={"D.O. (mg/l)": "DO", "CONDUCTIVITY (µmhos/cm)": "Conductivity", "B.O.D. (mg/l)": "BOD", "NITRATENAN N+ NI
```

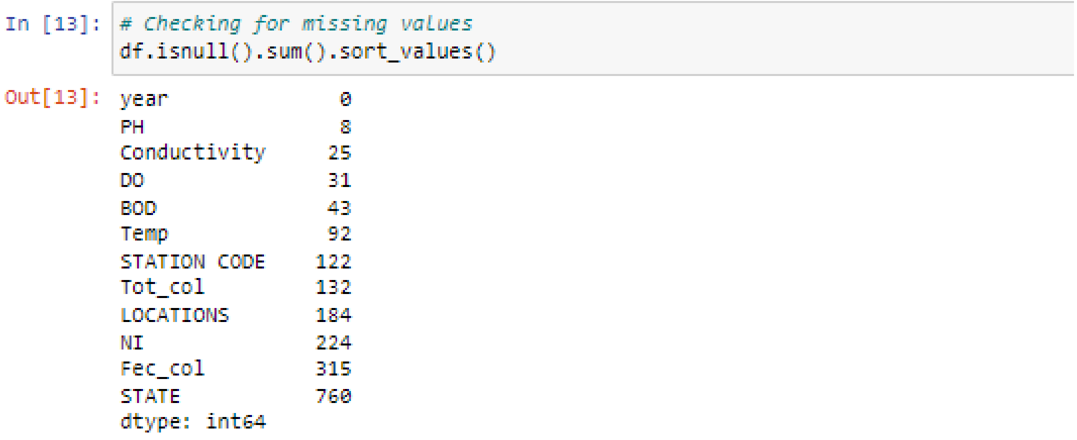


```
In [11]: # Converting object data type to numeric
         def convert_to_numeric(df):
             num_col = df.shape[1]
             # Start from index 3
             for index in range(3, num_col):
                 col_name = df.iloc[:, index].name
                 df[col_name] = pd.to_numeric(df[col_name], errors="coerce")
             return df

         df = convert_to_numeric(df)
         df.dtypes

Out[11]: STATION CODE     object
         LOCATIONS        object
         STATE            object
         Temp            float64
         DO              float64
         PH              float64
         Conductivity    float64
         BOD             float64
         NI              float64
         Fec_col         float64
         Tot_col         float64
         year              int64
         dtype: object
```

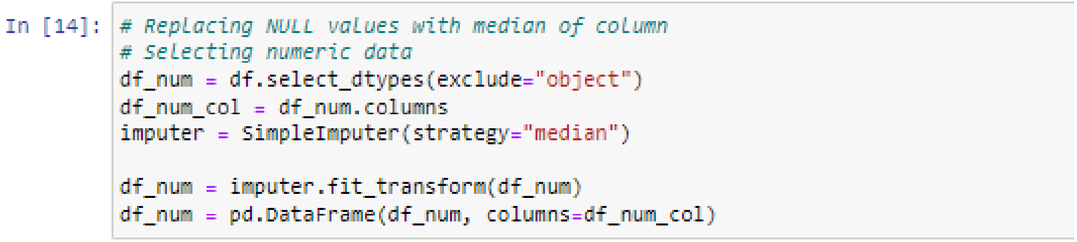
Handling missing values



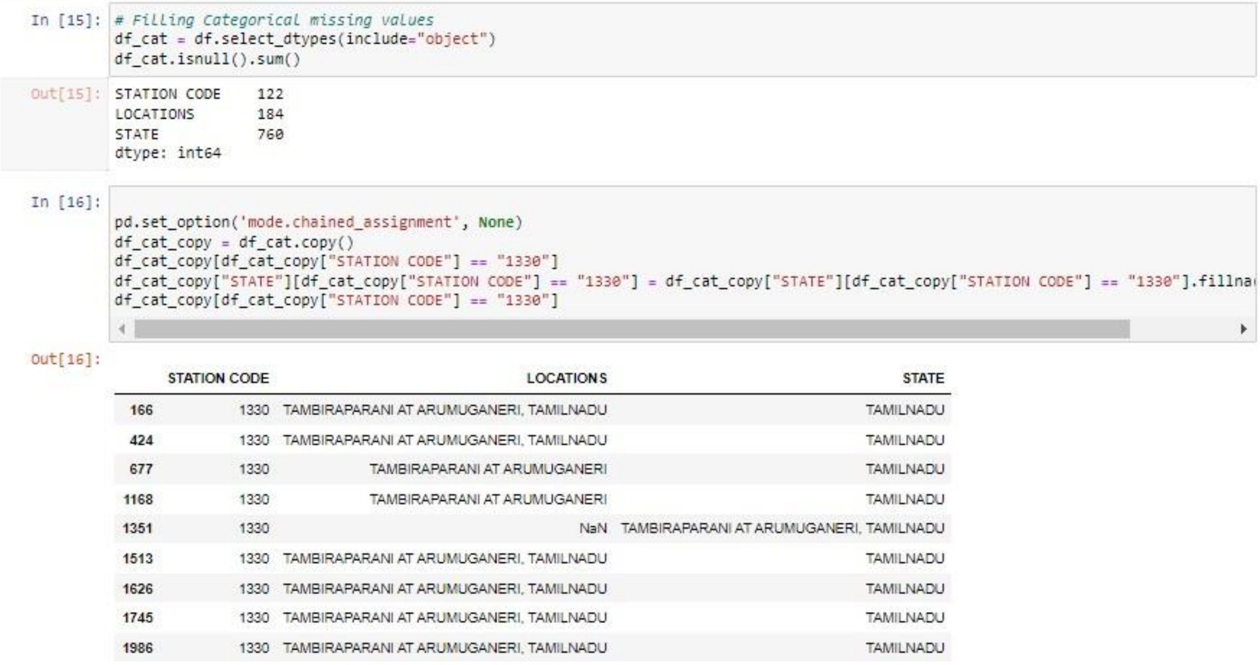
```
In [12]: # Replacing string NAN values with actual NAN value (np.nan)
         def convert_to_nan(df):
             n_col = df.shape[1]
             for index in range(n_col):
                 df.iloc[:, index]  = df.iloc[:, index].replace("NAN", np.nan)
             return df

         df = convert_to_nan(df)
```



```
In [13]: # Checking for missing values
         df.isnull().sum().sort_values()

Out[13]: year              0
         PH                8
         Conductivity     25
         DO               31
         BOD              43
         Temp             92
         STATION CODE    122
         Tot_col         132
         LOCATIONS       184
         NI              224
         Fec_col         315
         STATE           760
         dtype: int64
```



```
In [14]: # Replacing NULL values with median of column
         # Selecting numeric data
         df_num = df.select_dtypes(exclude="object")
         df_num_col = df_num.columns
         imputer = SimpleImputer(strategy="median")

         df_num = imputer.fit_transform(df_num)
         df_num = pd.DataFrame(df_num, columns=df_num_col)
```



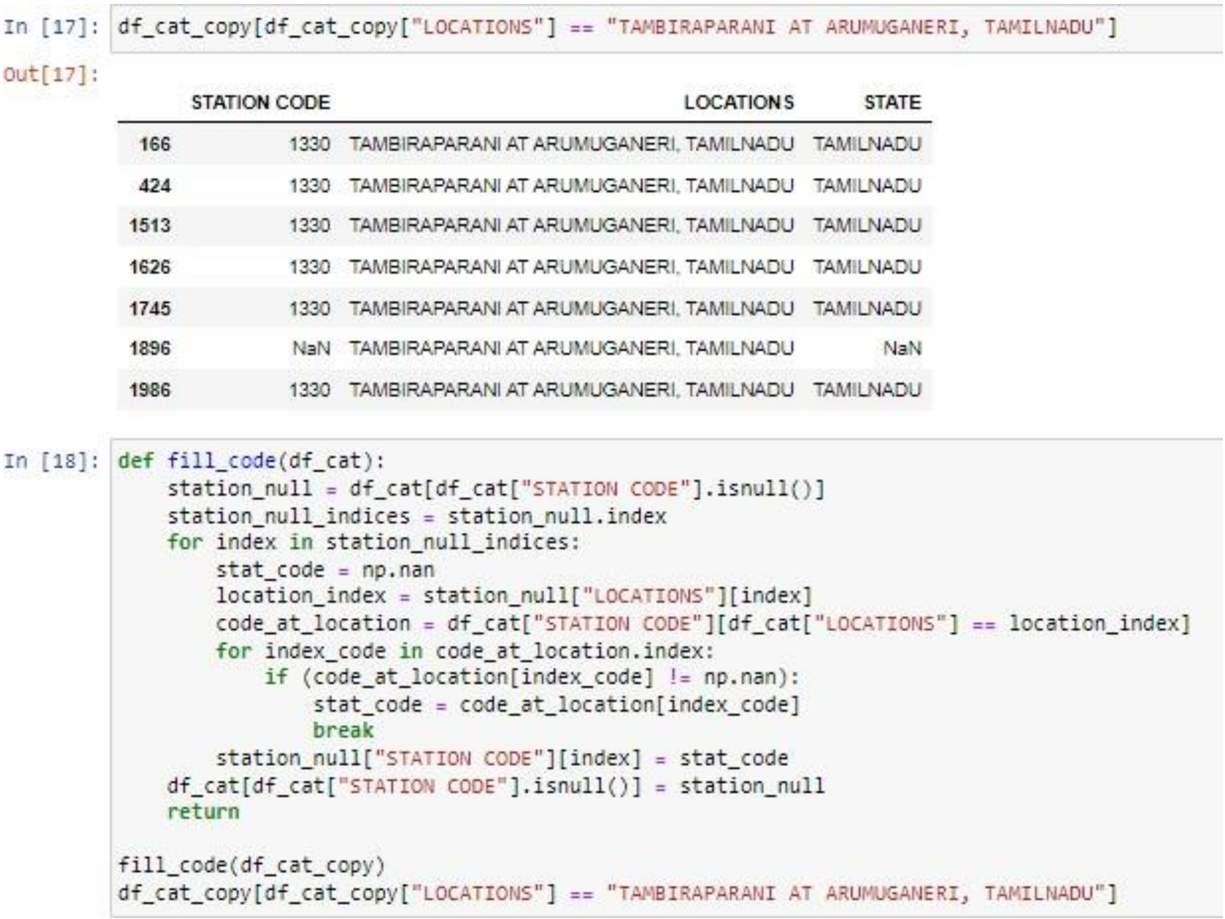
In [15]:
```
# Filling Categorical missing values
df_cat = df.select_dtypes(include="object")
df_cat.isnull().sum()
```

Out[15]:
```
STATION CODE    122
LOCATIONS       184
STATE           760
dtype: int64
```

In [16]:
```

pd.set_option('mode.chained_assignment', None)
df_cat_copy = df_cat.copy()
df_cat_copy[df_cat_copy["STATION CODE"] == "1330"]
df_cat_copy["STATE"][df_cat_copy["STATION CODE"] == "1330"] = df_cat_copy["STATE"][df_cat_copy["STATION CODE"] == "1330"].fillna
df_cat_copy[df_cat_copy["STATION CODE"] == "1330"]
```

Out[16]:

	STATION CODE	LOCATIONS	STATE
166	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
424	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
677	1330	TAMBIRAPARANI AT ARUMUGANERI	TAMILNADU
1168	1330	TAMBIRAPARANI AT ARUMUGANERI	TAMILNADU
1351	1330	NaN	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU
1513	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1626	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1745	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1986	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU



In [17]:
```
df_cat_copy[df_cat_copy["LOCATIONS"] == "TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU"]
```

Out[17]:

	STATION CODE	LOCATIONS	STATE
166	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
424	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1513	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1626	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1745	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1896	NaN	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	NaN
1986	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU

In [18]:
```
def fill_code(df_cat):
    station_null = df_cat[df_cat["STATION CODE"].isnull()]
    station_null_indices = station_null.index
    for index in station_null_indices:
        stat_code = np.nan
        location_index = station_null["LOCATIONS"][index]
        code_at_location = df_cat["STATION CODE"][df_cat["LOCATIONS"] == location_index]
        for index_code in code_at_location.index:
            if (code_at_location[index_code] != np.nan):
                stat_code = code_at_location[index_code]
                break
        station_null["STATION CODE"][index] = stat_code
    df_cat[df_cat["STATION CODE"].isnull()] = station_null
    return

fill_code(df_cat_copy)
df_cat_copy[df_cat_copy["LOCATIONS"] == "TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU"]
```

Out[18]:

	STATION CODE	LOCATIONS	STATE
166	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
424	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1513	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1626	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1745	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1896	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	NaN
1986	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU

In [19]:

```
# Filling all state NAN values which have corresponding station code value
def fill_state(df_cat):
    station_code = df_cat["STATION CODE"].unique()
    for index in range(station_code.shape[0]):
        if (station_code[index] != np.nan):
            df_state = df_cat["STATE"][df_cat["STATION CODE"] == station_code[index]]
            state_values = df_cat["STATE"][df_cat["STATION CODE"] == station_code[index]]
            state = np.nan
            for index_state in range(state_values.shape[0]):
                if (state_values.iloc[index_state] != np.nan):
                    state = state_values.iloc[index_state]
                    break
            df_state_fill = df_state.fillna(state)
            df_cat["STATE"][df_cat["STATION CODE"] == station_code[index]] = df_state_fill
    return
fill_state(df_cat_copy)
df_cat_copy[df_cat_copy["STATION CODE"] == "1330"]
```

Out[19]:

	STATION CODE	LOCATIONS	STATE
166	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
424	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
677	1330	TAMBIRAPARANI AT ARUMUGANERI	TAMILNADU
1168	1330	TAMBIRAPARANI AT ARUMUGANERI	TAMILNADU
1351	1330	NaN	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU
1513	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1626	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1745	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1896	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU
1986	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU

In [20]:

```
df_cat_copy.isnull().sum()
```

Out[20]:

```
STATION CODE      5
LOCATIONS       184
STATE            13
dtype: int64
```

In [21]:
```
df_cat_copy[df_cat_copy["STATE"].isnull()]
```

Out[21]:

	STATION CODE	LOCATIONS	STATE
260	NaN	NaN	NaN
431	NaN	NaN	NaN
1106	1207	KABBANI AT MUTHANKARA	NaN
1107	1208	BHAVANI AT ELACHIVAZHY	NaN
1650	2047	NNANCHOE (ATTAWA CHOE), CHANDIGARH	NaN
1651	2048	PATIALA KI RAO, CHANDIGARH	NaN
1652	2049	SUKHNA CHOE, CHANDIGARH	NaN
1770	2047	NNANCHOE (ATTAWA CHOE)	NaN
1771	2048	PATIALA KI RAO	NaN
1772	2049	SUKHNA CHOE	NaN
1784	NaN	DAMANGANGA AFTER CONFL. OF PIPARIA DRAIN, DAMAN	NaN
1785	NaN	DAMANGANGA AT CIRCUIT HOUSE, SILVASA, DADRA AN...	NaN
1912	NaN	NaN	NaN

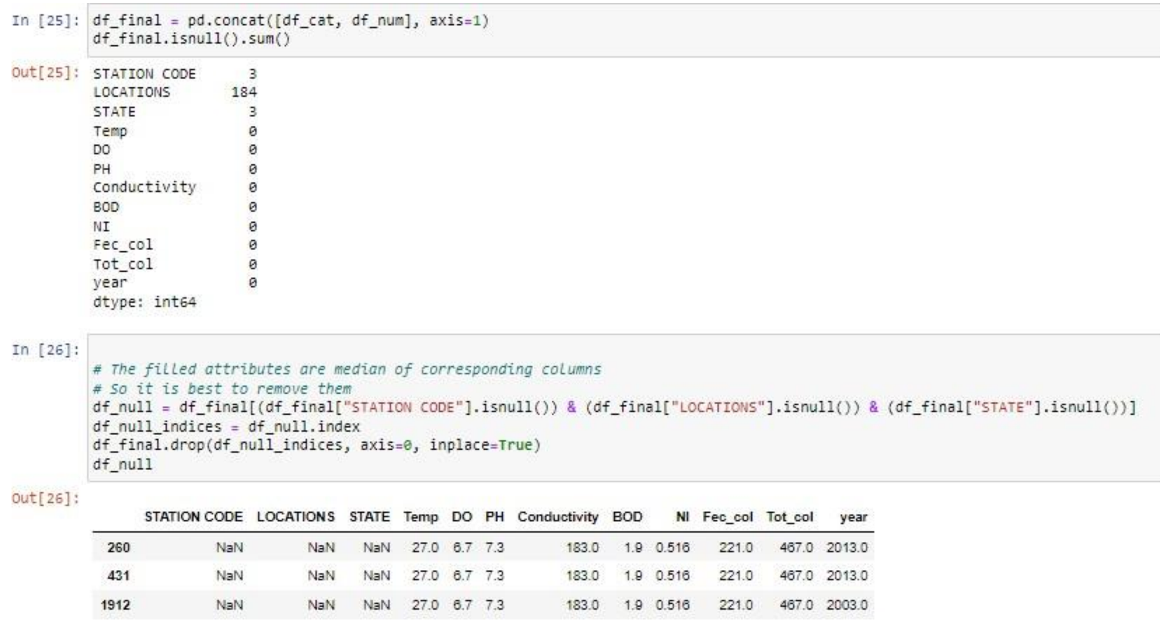
In [22]:
```
# The first location KABBANI AT MUTHANKARA is in STATE Kerela
df_cat_copy["STATE"][1106] = "KERALA"
df_cat_copy["STATE"][1107] = "KERALA"
df_cat_copy["STATE"][1650] = "CHANDIGARH"
df_cat_copy["STATE"][1651] = "CHANDIGARH"
df_cat_copy["STATE"][1652] = "CHANDIGARH"
df_cat_copy["STATE"][1770] = "CHANDIGARH"
df_cat_copy["STATE"][1771] = "CHANDIGARH"
df_cat_copy["STATE"][1772] = "CHANDIGARH"
df_cat_copy["STATE"][1784] = "DAMAN & DIU"
df_cat_copy["STATE"][1785] = "DAMAN & DIU"
df_cat_copy["STATION CODE"][1784] = "0000" # I am setting this according to myself
df_cat_copy["STATION CODE"][1785] = "0000"
```

In [23]:
```
df_cat = df_cat_copy
df_cat.isnull().sum()
```

Out[23]:
```
STATION CODE      3
LOCATIONS       184
STATE             3
dtype: int64
```

In [24]:
```
df_num.isnull().sum()
```

Out[24]:
```
Temp            0
DO              0
PH              0
Conductivity    0
BOD             0
NI              0
Fec_col         0
Tot_col         0
year            0
dtype: int64
```



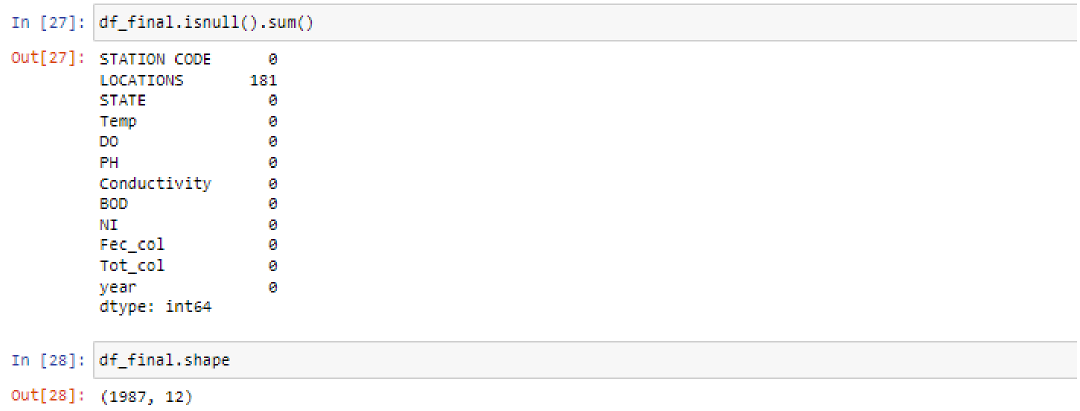
```
In [25]: df_final = pd.concat([df_cat, df_num], axis=1)
         df_final.isnull().sum()
```

```
Out[25]: STATION CODE      3
         LOCATIONS       184
         STATE             3
         Temp              0
         DO                0
         PH                0
         Conductivity      0
         BOD               0
         NI                0
         Fec_col           0
         Tot_col           0
         year              0
         dtype: int64
```

```
In [26]: 
         # The filled attributes are median of corresponding columns
         # So it is best to remove them
         df_null = df_final[(df_final["STATION CODE"].isnull()) & (df_final["LOCATIONS"].isnull()) & (df_final["STATE"].isnull())]
         df_null_indices = df_null.index
         df_final.drop(df_null_indices, axis=0, inplace=True)
         df_null
```

Out[26]:

	STATION CODE	LOCATIONS	STATE	Temp	DO	PH	Conductivity	BOD	NI	Fec_col	Tot_col	year
260	NaN	NaN	NaN	27.0	6.7	7.3	183.0	1.9	0.516	221.0	467.0	2013.0
431	NaN	NaN	NaN	27.0	6.7	7.3	183.0	1.9	0.516	221.0	467.0	2013.0
1912	NaN	NaN	NaN	27.0	6.7	7.3	183.0	1.9	0.516	221.0	467.0	2003.0



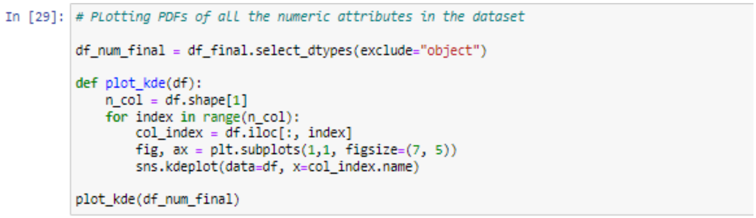
```
In [27]: df_final.isnull().sum()
```

```
Out[27]: STATION CODE      0
         LOCATIONS       181
         STATE             0
         Temp              0
         DO                0
         PH                0
         Conductivity      0
         BOD               0
         NI                0
         Fec_col           0
         Tot_col           0
         year              0
         dtype: int64
```

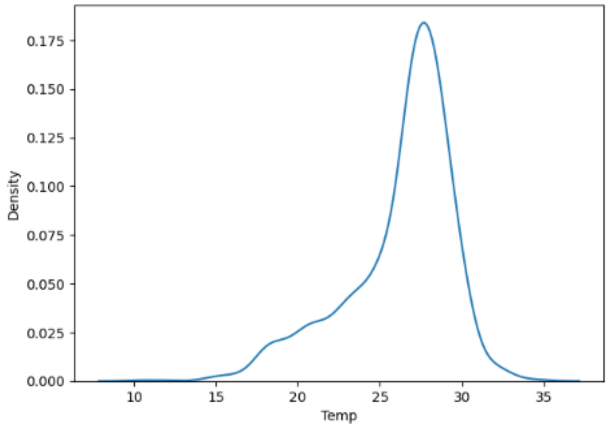
```
In [28]: df_final.shape
```

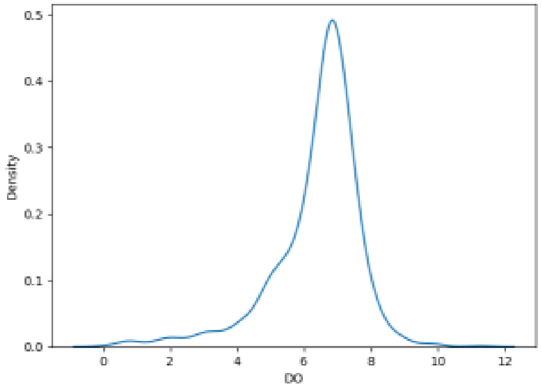
```
Out[28]: (1987, 12)
```

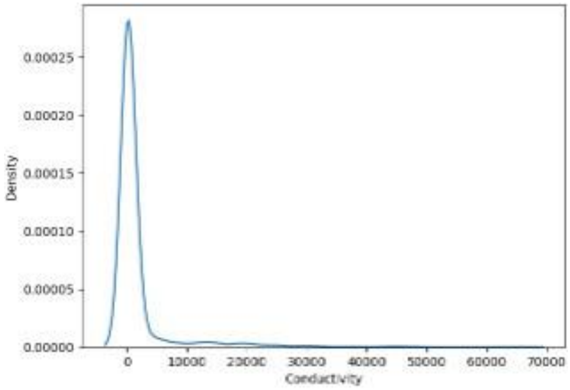
Data Visualization

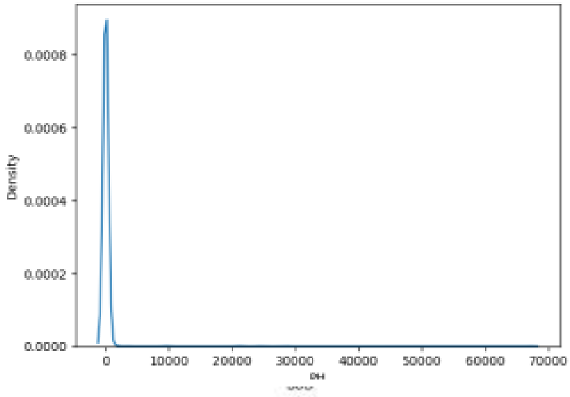


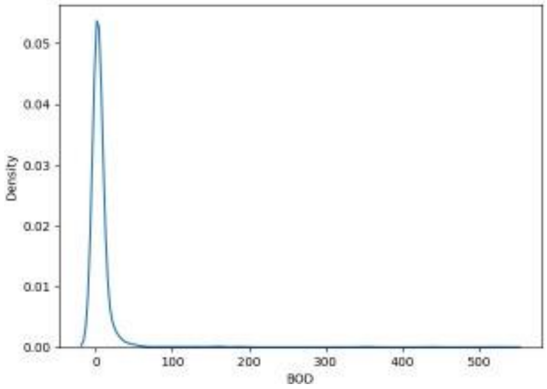
```
In [29]: # Plotting PDFs of all the numeric attributes in the dataset

         df_num_final = df_final.select_dtypes(exclude="object")

         def plot_kde(df):
             n_col = df.shape[1]
             for index in range(n_col):
                 col_index = df.iloc[:, index]
                 fig, ax = plt.subplots(1,1, figsize=(7, 5))
                 sns.kdeplot(data=df, x=col_index.name)

         plot_kde(df_num_final)
```

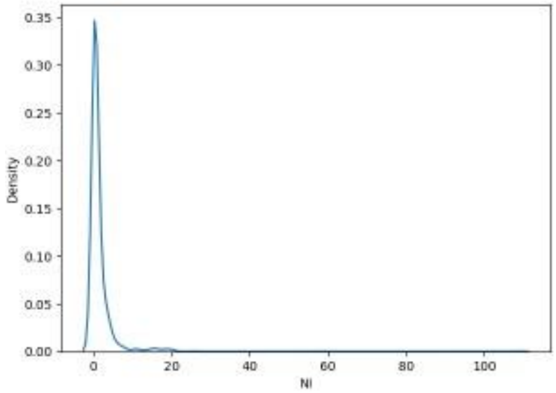


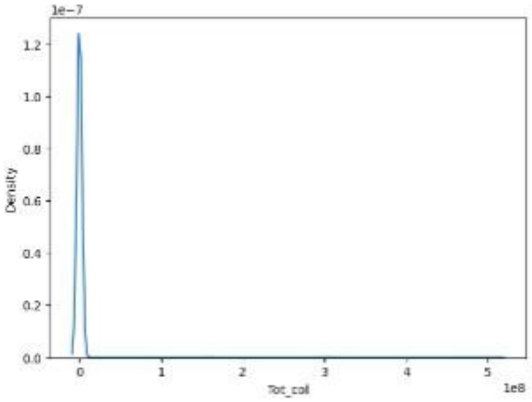

Density
0.5
0.4
0.3
0.2
0.1
0.0
0
2
4
6
8
10
12
DO

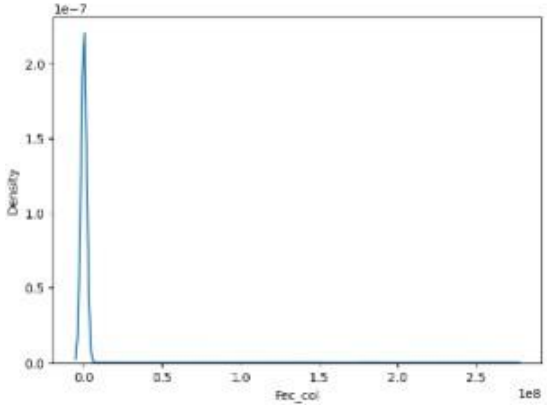

Density
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
0
10000
20000
30000
40000
50000
60000
70000
Conductivity

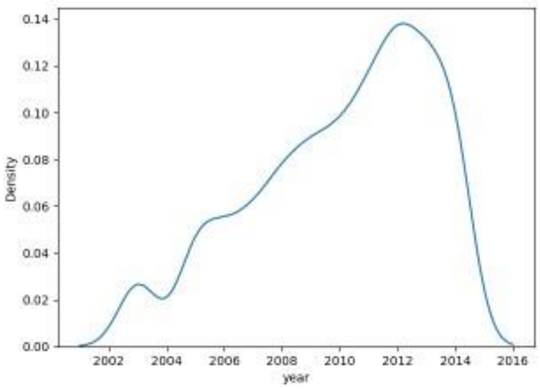

Density
0.0008
0.0006
0.0004
0.0002
0.0000
0
10000
20000
30000
40000
50000
60000
70000
PH


Density
0.05
0.04
0.03
0.02
0.01
0.00
0
100
200
300
400
500
BOD

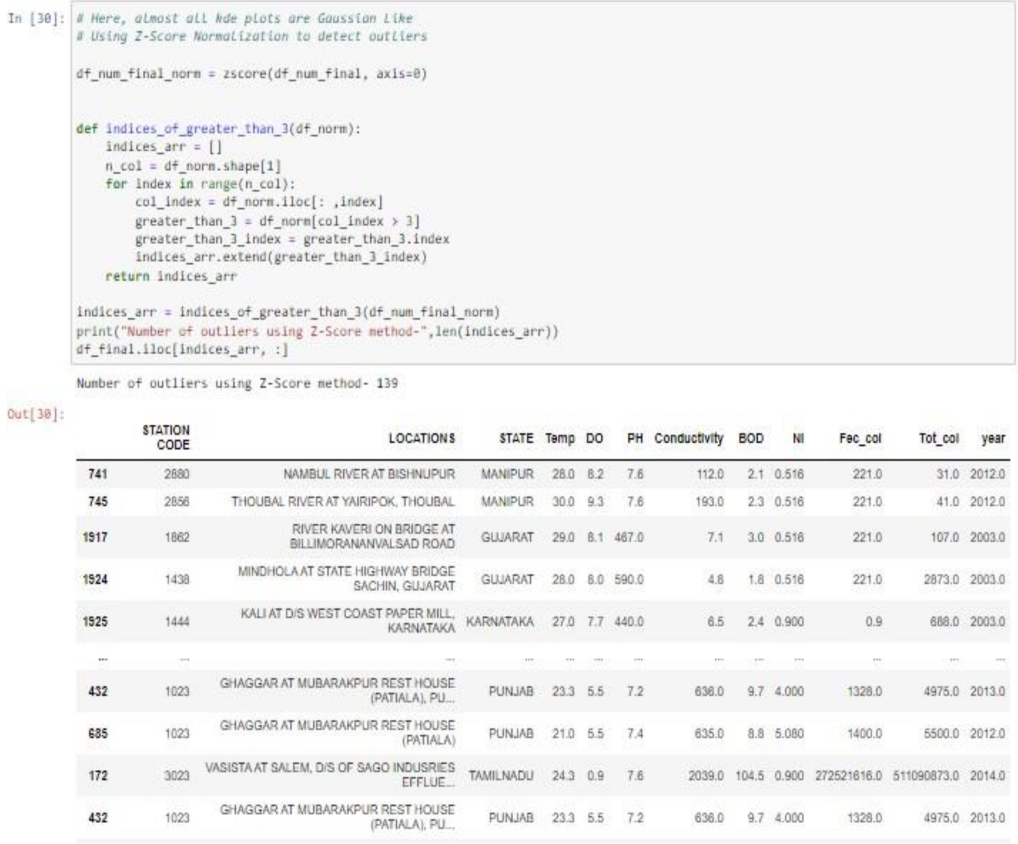

Density
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
20
40
60
80
100
NI


1e−7
Density
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
1e8
Tot_coli


1e−7
Density
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
1e8
Fec_coli


Density
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
2002
2004
2006
2008
2010
2012
2014
2016
year

Handling Outliers and replacing the outliers



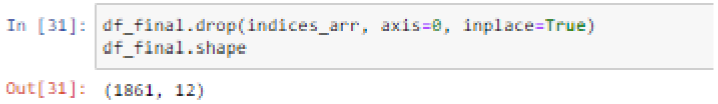
In [30]:

```
# Here, almost all kde plots are Gaussian Like
# Using Z-Score Normalization to detect outliers

df_num_final_norm = zscore(df_num_final, axis=0)


def indices_of_greater_than_3(df_norm):
    indices_arr = []
    n_col = df_norm.shape[1]
    for index in range(n_col):
        col_index = df_norm.iloc[: ,index]
        greater_than_3 = df_norm[col_index > 3]
        greater_than_3_index = greater_than_3.index
        indices_arr.extend(greater_than_3_index)
    return indices_arr

indices_arr = indices_of_greater_than_3(df_num_final_norm)
print("Number of outliers using Z-Score method-",len(indices_arr))
df_final.iloc[indices_arr, :]
```

```
Number of outliers using Z-Score method- 139
```

Out[30]:

	STATION CODE	LOCATIONS	STATE	Temp	DO	PH	Conductivity	BOD	NI	Fec_col	Tot_col	year
741	2880	NAMBUL RIVER AT BISHNUPUR	MANIPUR	28.0	8.2	7.6	112.0	2.1	0.516	221.0	31.0	2012.0
745	2856	THOUBAL RIVER AT YAIRIPOK, THOUBAL	MANIPUR	30.0	9.3	7.6	193.0	2.3	0.516	221.0	41.0	2012.0
1917	1862	RIVER KAVERI ON BRIDGE AT BILLIMORANANVALSAD ROAD	GUJARAT	29.0	8.1	467.0	7.1	3.0	0.516	221.0	107.0	2003.0
1924	1438	MINDHOLA AT STATE HIGHWAY BRIDGE SACHIN, GUJARAT	GUJARAT	28.0	8.0	590.0	4.8	1.8	0.516	221.0	2873.0	2003.0
1925	1444	KALI AT D/S WEST COAST PAPER MILL, KARNATAKA	KARNATAKA	27.0	7.7	440.0	6.5	2.4	0.900	0.9	688.0	2003.0
...	...	...	...	...	...	...	...	...	...	...	...	...
432	1023	GHAGGAR AT MUBARAKPUR REST HOUSE (PATIALA), PU...	PUNJAB	23.3	5.5	7.2	636.0	9.7	4.000	1328.0	4975.0	2013.0
685	1023	GHAGGAR AT MUBARAKPUR REST HOUSE (PATIALA)	PUNJAB	21.0	5.5	7.4	635.0	8.8	5.080	1400.0	5500.0	2012.0
172	3023	VASISTA AT SALEM, D/S OF SAGO INDUSRIES EFFLUE...	TAMILNADU	24.3	0.9	7.6	2039.0	104.5	0.900	272521616.0	511090873.0	2014.0
432	1023	GHAGGAR AT MUBARAKPUR REST HOUSE (PATIALA), PU...	PUNJAB	23.3	5.5	7.2	636.0	9.7	4.000	1328.0	4975.0	2013.0

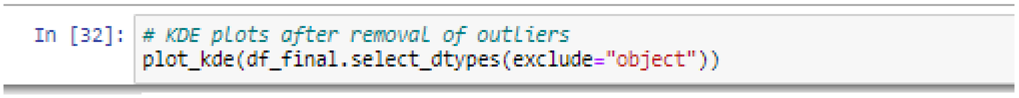


In [31]:

```
df_final.drop(indices_arr, axis=0, inplace=True)
df_final.shape
```

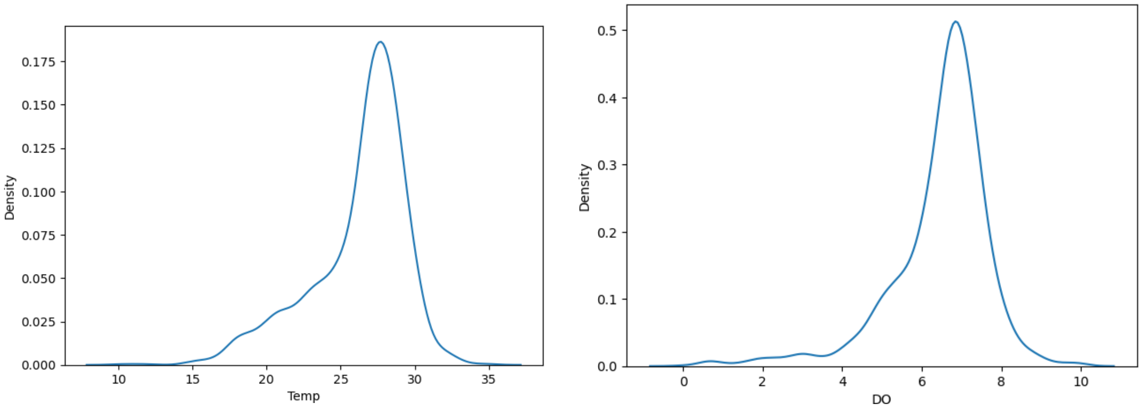
Out[31]:

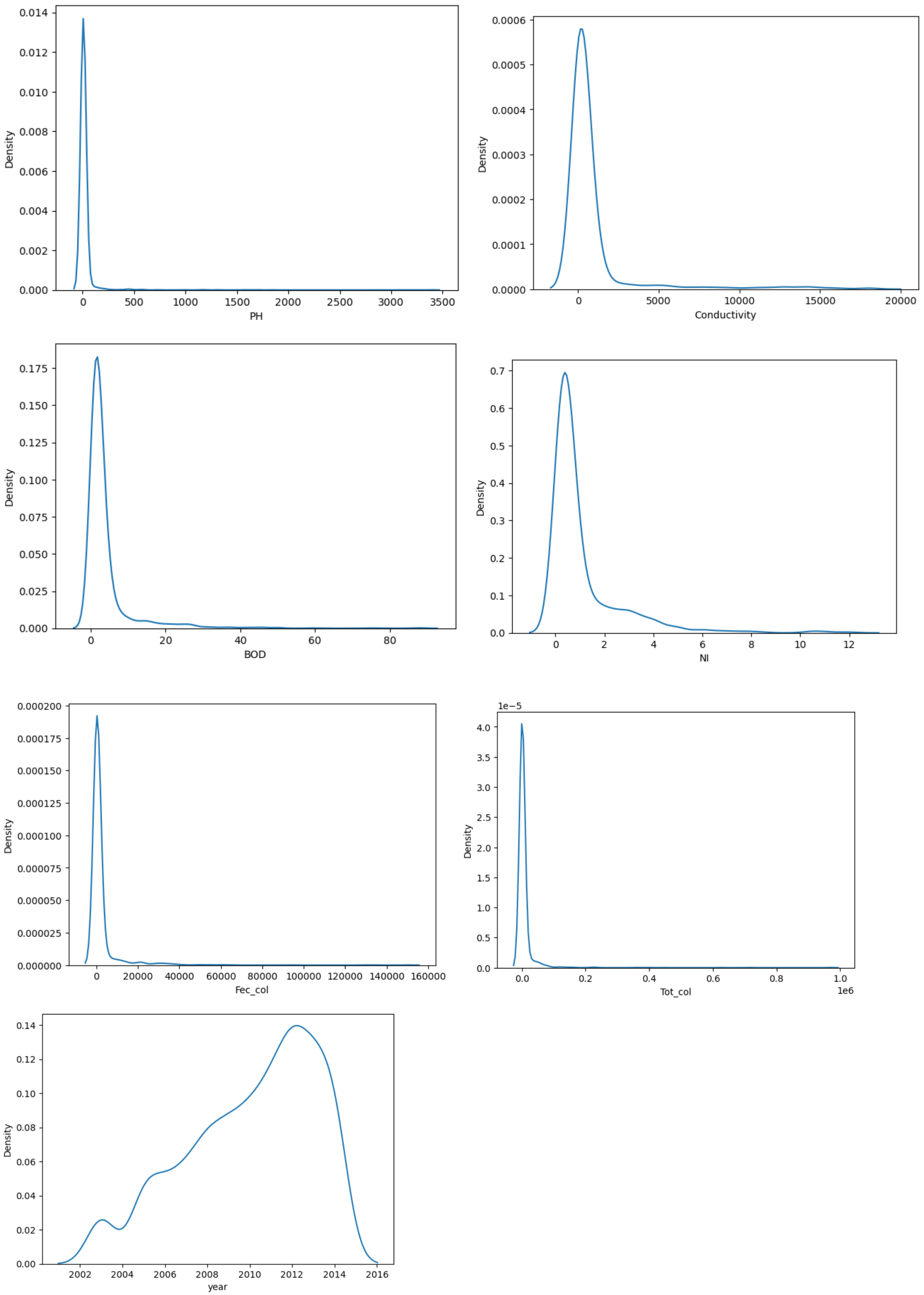
```
(1861, 12)
```



In [32]:

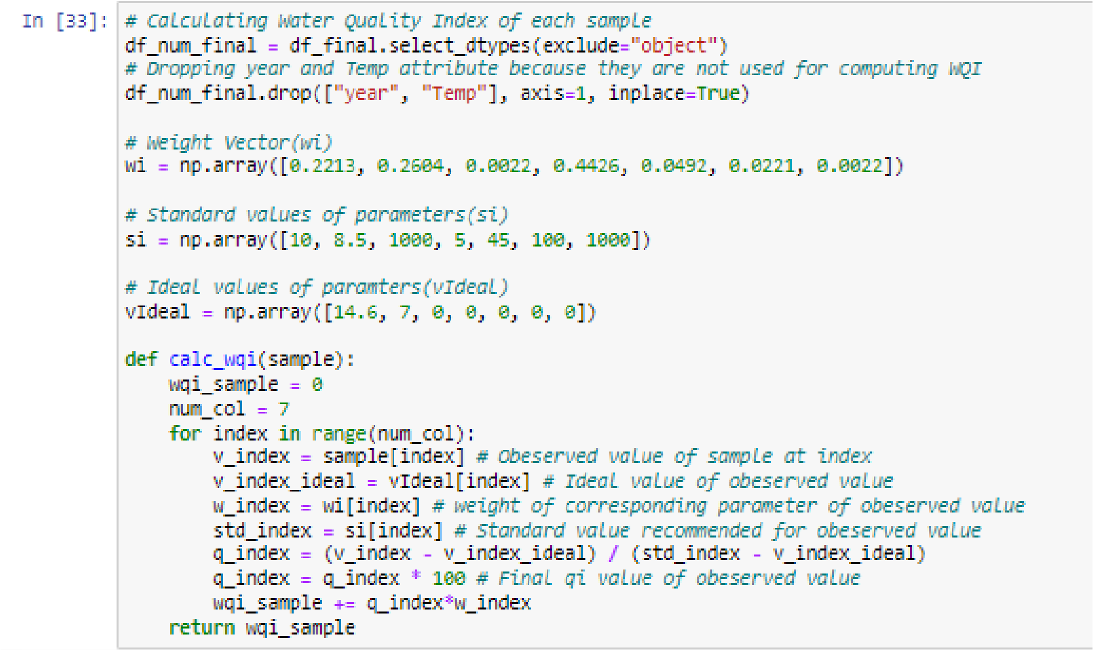
```
# KDE plots after removal of outliers
plot_kde(df_final.select_dtypes(exclude="object"))
```



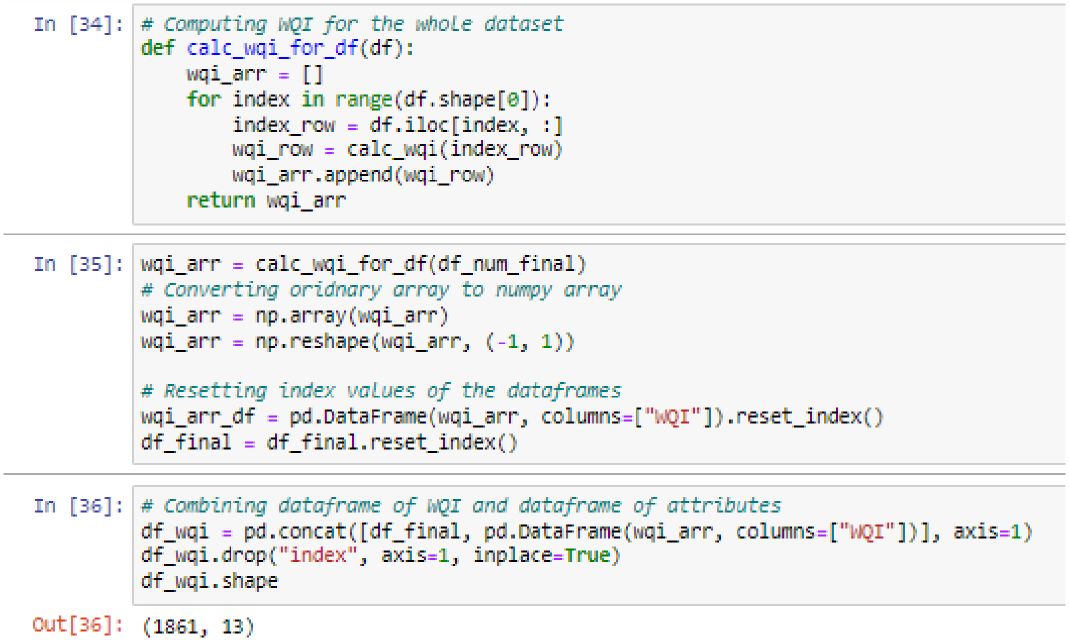


Density
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
500
1000
1500
2000
2500
3000
3500
PH
Density
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
5000
10000
15000
20000
Conductivity
Density
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0
20
40
60
80
BOD
Density
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
2
4
6
8
10
12
NI
Density
0.000200
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000
0
20000
40000
60000
80000
100000
120000
140000
160000
Fec_col
1e−5
Density
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1e6
Tot_col
Density
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
2002
2004
2006
2008
2010
2012
2014
2016
year

Calculating WQI



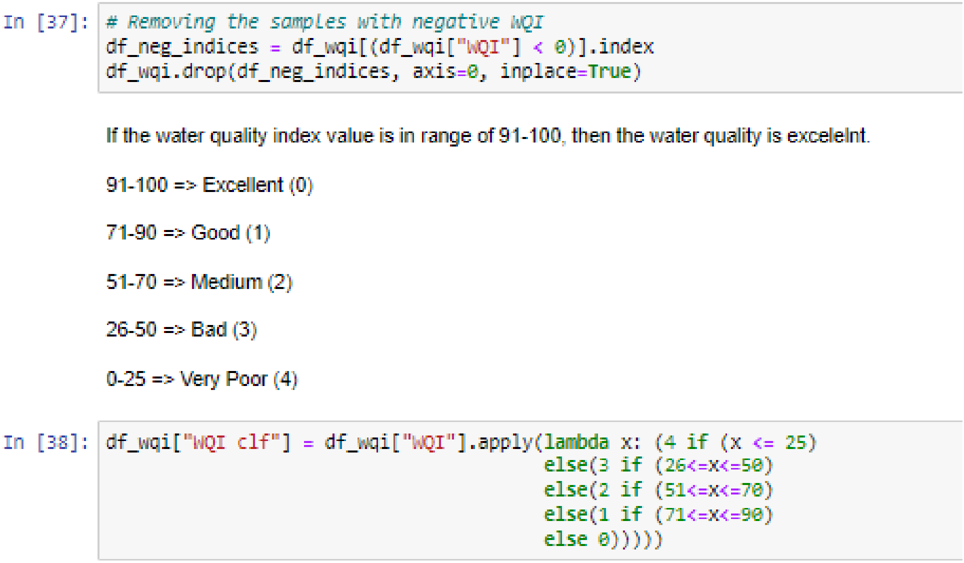
```
In [33]: # Calculating Water Quality Index of each sample
         df_num_final = df_final.select_dtypes(exclude="object")
         # Dropping year and Temp attribute because they are not used for computing WQI
         df_num_final.drop(["year", "Temp"], axis=1, inplace=True)

         # Weight Vector(wi)
         wi = np.array([0.2213, 0.2604, 0.0022, 0.4426, 0.0492, 0.0221, 0.0022])

         # Standard values of parameters(si)
         si = np.array([10, 8.5, 1000, 5, 45, 100, 1000])

         # Ideal values of paramters(vIdeal)
         vIdeal = np.array([14.6, 7, 0, 0, 0, 0, 0])

         def calc_wqi(sample):
             wqi_sample = 0
             num_col = 7
             for index in range(num_col):
                 v_index = sample[index] # Obeserved value of sample at index
                 v_index_ideal = vIdeal[index] # Ideal value of obeserved value
                 w_index = wi[index] # weight of corresponding parameter of obeserved value
                 std_index = si[index] # Standard value recommended for obeserved value
                 q_index = (v_index - v_index_ideal) / (std_index - v_index_ideal)
                 q_index = q_index * 100 # Final qi value of obeserved value
                 wqi_sample += q_index*w_index
             return wqi_sample
```



```
In [34]: # Computing WQI for the whole dataset
         def calc_wqi_for_df(df):
             wqi_arr = []
             for index in range(df.shape[0]):
                 index_row = df.iloc[index, :]
                 wqi_row = calc_wqi(index_row)
                 wqi_arr.append(wqi_row)
             return wqi_arr
```

```
In [35]: wqi_arr = calc_wqi_for_df(df_num_final)
         # Converting oridnary array to numpy array
         wqi_arr = np.array(wqi_arr)
         wqi_arr = np.reshape(wqi_arr, (-1, 1))

         # Resetting index values of the dataframes
         wqi_arr_df = pd.DataFrame(wqi_arr, columns=["WQI"]).reset_index()
         df_final = df_final.reset_index()
```

```
In [36]: # Combining dataframe of WQI and dataframe of attributes
         df_wqi = pd.concat([df_final, pd.DataFrame(wqi_arr, columns=["WQI"])], axis=1)
         df_wqi.drop("index", axis=1, inplace=True)
         df_wqi.shape

Out[36]: (1861, 13)
```



```
In [37]: # Removing the samples with negative WQI
         df_neg_indices = df_wqi[(df_wqi["WQI"] < 0)].index
         df_wqi.drop(df_neg_indices, axis=0, inplace=True)
```

If the water quality index value is in range of 91-100, then the water quality is exceleInt.

91-100 => Excellent (0)

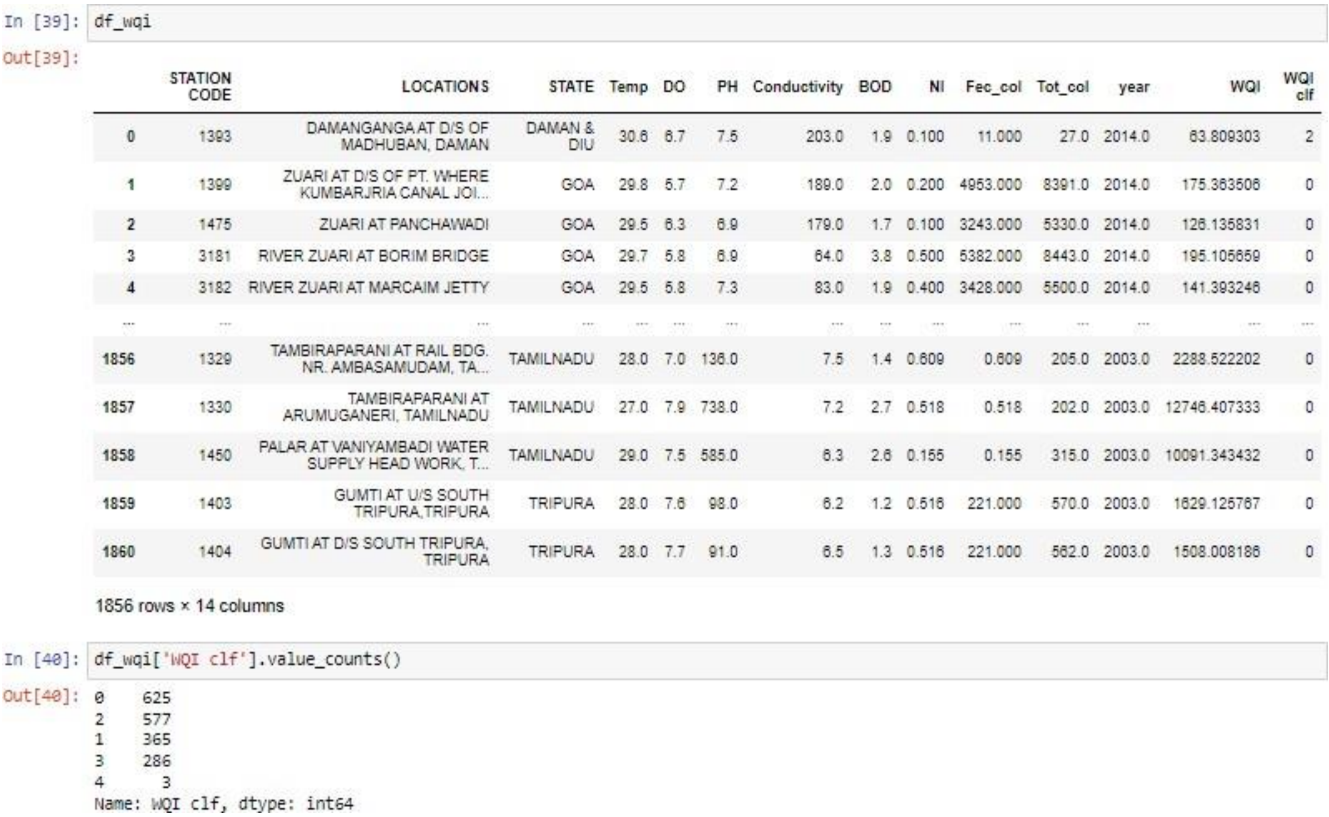
71-90 => Good (1)

51-70 => Medium (2)

26-50 => Bad (3)

0-25 => Very Poor (4)

```
In [38]: df_wqi["WQI clf"] = df_wqi["WQI"].apply(lambda x: (4 if (x <= 25)
                                                 else(3 if (26<=x<=50)
                                                 else(2 if (51<=x<=70)
                                                 else(1 if (71<=x<=90)
                                                 else 0)))))
```



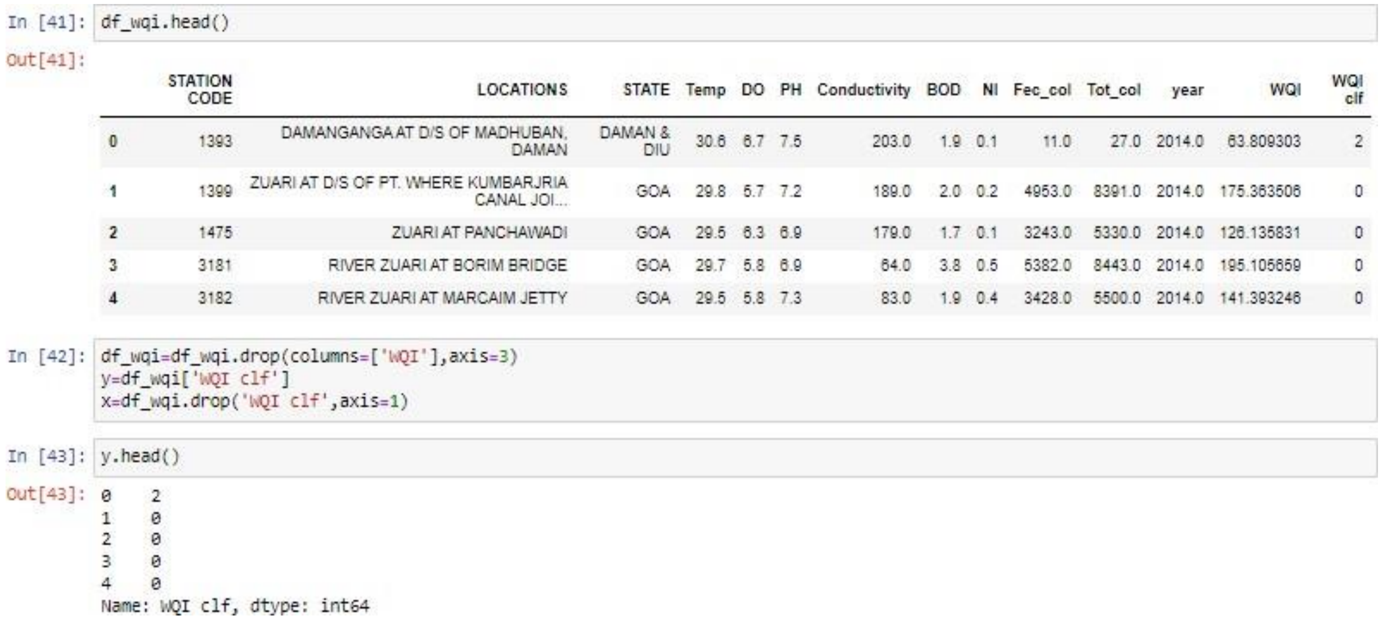
```
In [39]: df_wqi
Out[39]:
```

	STATION CODE	LOCATIONS	STATE	Temp	DO	PH	Conductivity	BOD	NI	Fec_col	Tot_col	year	WQI	WQI clf
0	1393	DAMANGANGA AT D/S OF MADHUBAN, DAMAN	DAMAN & DIU	30.6	6.7	7.5	203.0	1.9	0.100	11.000	27.0	2014.0	63.809303	2
1	1399	ZUARI AT D/S OF PT. WHERE KUMBARJRIA CANAL JOI...	GOA	29.8	5.7	7.2	189.0	2.0	0.200	4953.000	8391.0	2014.0	175.363506	0
2	1475	ZUARI AT PANCHAWADI	GOA	29.5	6.3	6.9	179.0	1.7	0.100	3243.000	5330.0	2014.0	126.135831	0
3	3181	RIVER ZUARI AT BORIM BRIDGE	GOA	29.7	5.8	6.9	64.0	3.8	0.500	5382.000	8443.0	2014.0	195.105659	0
4	3182	RIVER ZUARI AT MARCAIM JETTY	GOA	29.5	5.8	7.3	83.0	1.9	0.400	3428.000	5500.0	2014.0	141.393246	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1856	1329	TAMBIRAPARANI AT RAIL BDG. NR. AMBASAMUDAM, TA...	TAMILNADU	28.0	7.0	136.0	7.5	1.4	0.609	0.609	205.0	2003.0	2288.522202	0
1857	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU	27.0	7.9	738.0	7.2	2.7	0.518	0.518	202.0	2003.0	12746.407333	0
1858	1450	PALAR AT VANIYAMBADI WATER SUPPLY HEAD WORK, T...	TAMILNADU	29.0	7.5	585.0	6.3	2.6	0.155	0.155	315.0	2003.0	10091.343432	0
1859	1403	GUMTI AT U/S SOUTH TRIPURA,TRIPURA	TRIPURA	28.0	7.6	98.0	6.2	1.2	0.516	221.000	570.0	2003.0	1629.125767	0
1860	1404	GUMTI AT D/S SOUTH TRIPURA, TRIPURA	TRIPURA	28.0	7.7	91.0	6.5	1.3	0.516	221.000	562.0	2003.0	1508.008186	0

1856 rows × 14 columns

```
In [40]: df_wqi['WQI clf'].value_counts()
Out[40]: 0    625
         2    577
         1    365
         3    286
         4      3
         Name: WQI clf, dtype: int64
```

Independent variable and Dependent variable



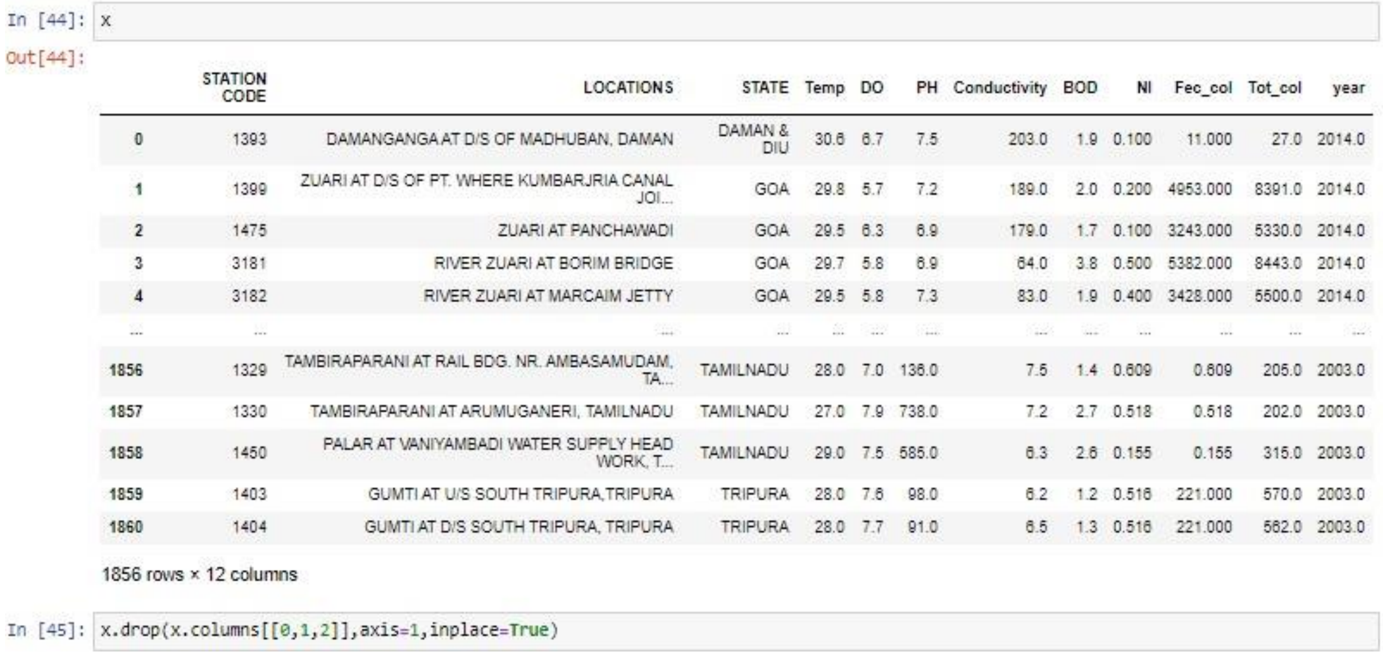
```
In [41]: df_wqi.head()
```

Out[41]:

	STATION CODE	LOCATIONS	STATE	Temp	DO	PH	Conductivity	BOD	NI	Fec_col	Tot_col	year	WQI	WQI clf
0	1393	DAMANGANGA AT D/S OF MADHUBAN, DAMAN	DAMAN & DIU	30.6	6.7	7.5	203.0	1.9	0.1	11.0	27.0	2014.0	63.809303	2
1	1399	ZUARI AT D/S OF PT. WHERE KUMBARJRIA CANAL JOI...	GOA	29.8	5.7	7.2	189.0	2.0	0.2	4953.0	8391.0	2014.0	175.363506	0
2	1475	ZUARI AT PANCHAWADI	GOA	29.5	6.3	6.9	179.0	1.7	0.1	3243.0	5330.0	2014.0	126.135831	0
3	3181	RIVER ZUARI AT BORIM BRIDGE	GOA	29.7	5.8	6.9	64.0	3.8	0.5	5382.0	8443.0	2014.0	195.105659	0
4	3182	RIVER ZUARI AT MARCAIM JETTY	GOA	29.5	5.8	7.3	83.0	1.9	0.4	3428.0	5500.0	2014.0	141.393246	0

```
In [42]: df_wqi=df_wqi.drop(columns=['WQI'],axis=3)
         y=df_wqi['WQI clf']
         x=df_wqi.drop('WQI clf',axis=1)
```

```
In [43]: y.head()

Out[43]: 0    2
         1    0
         2    0
         3    0
         4    0
         Name: WQI clf, dtype: int64
```



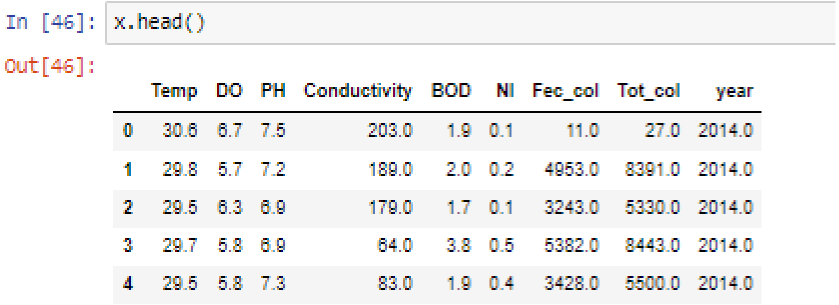
```
In [44]: x
```

Out[44]:

	STATION CODE	LOCATIONS	STATE	Temp	DO	PH	Conductivity	BOD	NI	Fec_col	Tot_col	year
0	1393	DAMANGANGA AT D/S OF MADHUBAN, DAMAN	DAMAN & DIU	30.6	6.7	7.5	203.0	1.9	0.100	11.000	27.0	2014.0
1	1399	ZUARI AT D/S OF PT. WHERE KUMBARJRIA CANAL JOI...	GOA	29.8	5.7	7.2	189.0	2.0	0.200	4953.000	8391.0	2014.0
2	1475	ZUARI AT PANCHAWADI	GOA	29.5	6.3	6.9	179.0	1.7	0.100	3243.000	5330.0	2014.0
3	3181	RIVER ZUARI AT BORIM BRIDGE	GOA	29.7	5.8	6.9	64.0	3.8	0.500	5382.000	8443.0	2014.0
4	3182	RIVER ZUARI AT MARCAIM JETTY	GOA	29.5	5.8	7.3	83.0	1.9	0.400	3428.000	5500.0	2014.0
...	...	...	...	...	...	...	...	...	...	...	...	...
1856	1329	TAMBIRAPARANI AT RAIL BDG. NR. AMBASAMUDAM, TA...	TAMILNADU	28.0	7.0	136.0	7.5	1.4	0.609	0.609	205.0	2003.0
1857	1330	TAMBIRAPARANI AT ARUMUGANERI, TAMILNADU	TAMILNADU	27.0	7.9	738.0	7.2	2.7	0.518	0.518	202.0	2003.0
1858	1450	PALAR AT VANIYAMBADI WATER SUPPLY HEAD WORK, T...	TAMILNADU	29.0	7.5	585.0	6.3	2.6	0.155	0.155	315.0	2003.0
1859	1403	GUMTI AT U/S SOUTH TRIPURA,TRIPURA	TRIPURA	28.0	7.6	98.0	6.2	1.2	0.516	221.000	570.0	2003.0
1860	1404	GUMTI AT D/S SOUTH TRIPURA, TRIPURA	TRIPURA	28.0	7.7	91.0	6.5	1.3	0.516	221.000	562.0	2003.0

1856 rows × 12 columns

```
In [45]: x.drop(x.columns[[0,1,2]],axis=1,inplace=True)
```

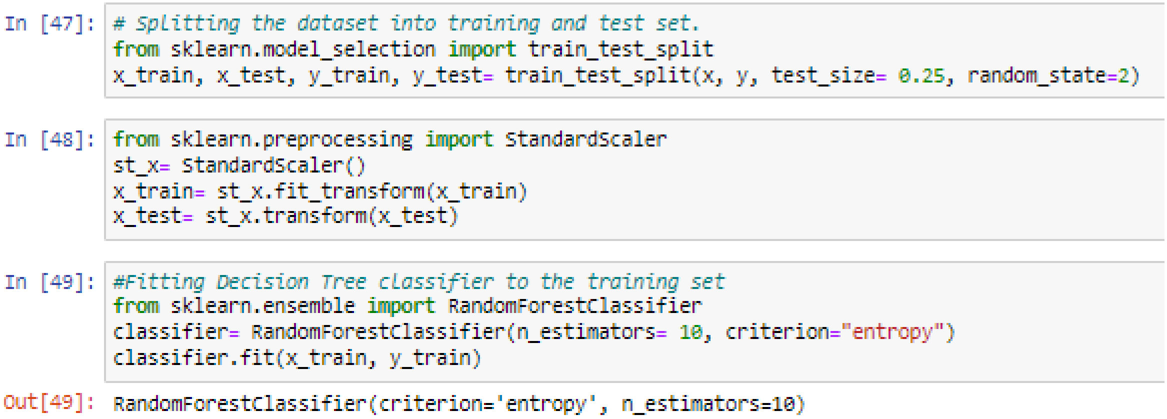


```
In [46]: x.head()
```

Out[46]:

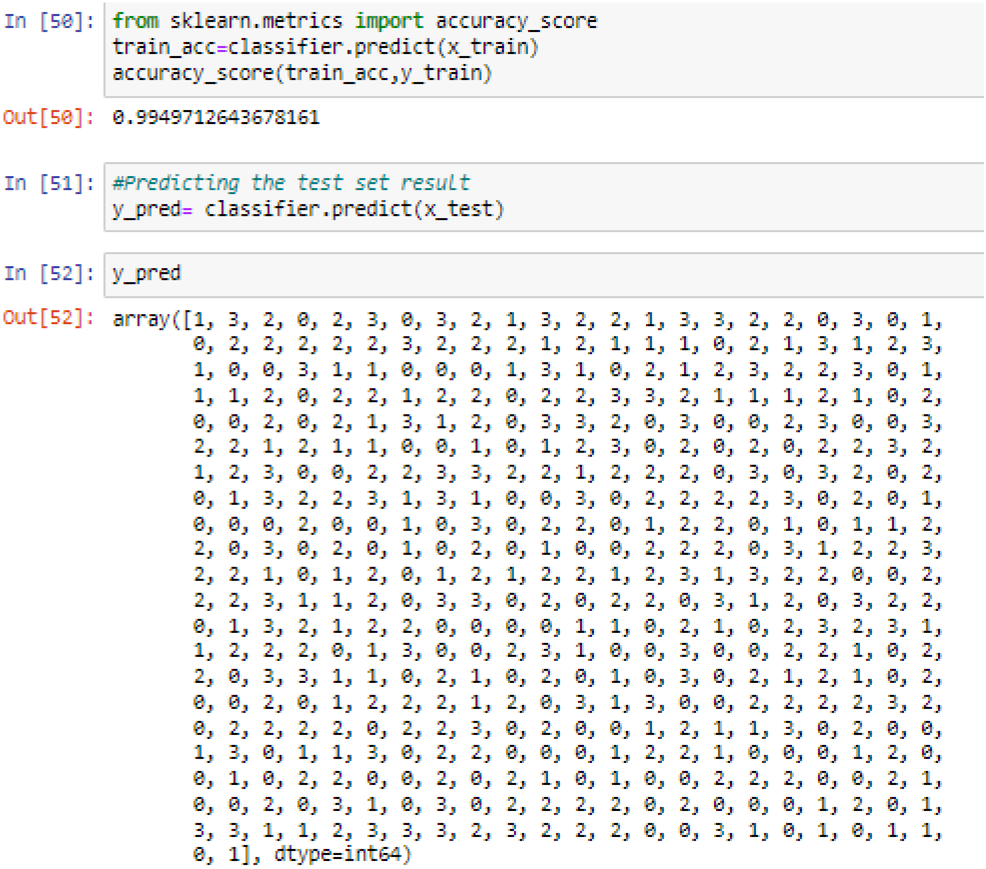
	Temp	DO	PH	Conductivity	BOD	NI	Fec_col	Tot_col	year
0	30.6	6.7	7.5	203.0	1.9	0.1	11.0	27.0	2014.0
1	29.8	5.7	7.2	189.0	2.0	0.2	4953.0	8391.0	2014.0
2	29.5	6.3	6.9	179.0	1.7	0.1	3243.0	5330.0	2014.0
3	29.7	5.8	6.9	64.0	3.8	0.5	5382.0	8443.0	2014.0
4	29.5	5.8	7.3	83.0	1.9	0.4	3428.0	5500.0	2014.0

Splitting of Train and Test data



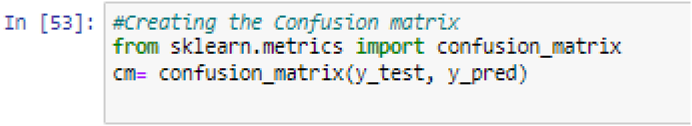
```
In [47]: # Splitting the dataset into training and test set.
         from sklearn.model_selection import train_test_split
         x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.25, random_state=2)

In [48]: from sklearn.preprocessing import StandardScaler
         st_x= StandardScaler()
         x_train= st_x.fit_transform(x_train)
         x_test= st_x.transform(x_test)

In [49]: #Fitting Decision Tree classifier to the training set
         from sklearn.ensemble import RandomForestClassifier
         classifier= RandomForestClassifier(n_estimators= 10, criterion="entropy")
         classifier.fit(x_train, y_train)

Out[49]: RandomForestClassifier(criterion='entropy', n_estimators=10)
```

Accuracy score of training dataset



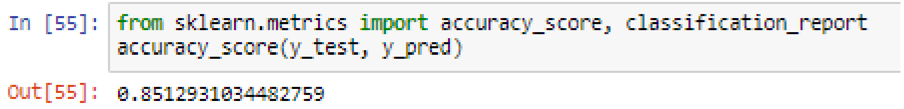
```
In [50]: from sklearn.metrics import accuracy_score
         train_acc=classifier.predict(x_train)
         accuracy_score(train_acc,y_train)

Out[50]: 0.9949712643678161

In [51]: #Predicting the test set result
         y_pred= classifier.predict(x_test)

In [52]: y_pred

Out[52]: array([1, 3, 2, 0, 2, 3, 0, 3, 2, 1, 3, 2, 2, 1, 3, 3, 2, 2, 0, 3, 0, 1,
                0, 2, 2, 2, 2, 2, 3, 2, 2, 2, 1, 2, 1, 1, 1, 0, 2, 1, 3, 1, 2, 3,
                1, 0, 0, 3, 1, 1, 0, 0, 0, 1, 3, 1, 0, 2, 1, 2, 3, 2, 2, 3, 0, 1,
                1, 1, 2, 0, 2, 2, 1, 2, 2, 0, 2, 2, 3, 3, 2, 1, 1, 1, 2, 1, 0, 2,
                0, 0, 2, 0, 2, 1, 3, 1, 2, 0, 3, 3, 2, 0, 3, 0, 0, 2, 3, 0, 0, 3,
                2, 2, 1, 2, 1, 1, 0, 0, 1, 0, 1, 2, 3, 0, 2, 0, 2, 0, 2, 2, 3, 2,
                1, 2, 3, 0, 0, 2, 2, 3, 3, 2, 2, 1, 2, 2, 2, 0, 3, 0, 3, 2, 0, 2,
                0, 1, 3, 2, 2, 3, 1, 3, 1, 0, 0, 3, 0, 2, 2, 2, 2, 3, 0, 2, 0, 1,
                0, 0, 0, 2, 0, 0, 1, 0, 3, 0, 2, 2, 0, 1, 2, 2, 0, 1, 0, 1, 1, 2,
                2, 0, 3, 0, 2, 0, 1, 0, 2, 0, 1, 0, 0, 2, 2, 2, 0, 3, 1, 2, 2, 3,
                2, 2, 1, 0, 1, 2, 0, 1, 2, 1, 2, 2, 1, 2, 3, 1, 3, 2, 2, 0, 0, 2,
                2, 2, 3, 1, 1, 2, 0, 3, 3, 0, 2, 0, 2, 2, 0, 3, 1, 2, 0, 3, 2, 2,
                0, 1, 3, 2, 1, 2, 2, 0, 0, 0, 0, 1, 1, 0, 2, 1, 0, 2, 3, 2, 3, 1,
                1, 2, 2, 2, 0, 1, 3, 0, 0, 2, 3, 1, 0, 0, 3, 0, 0, 2, 2, 1, 0, 2,
                2, 0, 3, 3, 1, 1, 0, 2, 1, 0, 2, 0, 1, 0, 3, 0, 2, 1, 2, 1, 0, 2,
                0, 0, 2, 0, 1, 2, 2, 2, 1, 2, 0, 3, 1, 3, 0, 0, 2, 2, 2, 2, 3, 2,
                0, 2, 2, 2, 2, 0, 2, 2, 3, 0, 2, 0, 0, 1, 2, 1, 1, 3, 0, 2, 0, 0,
                1, 3, 0, 1, 1, 3, 0, 2, 2, 0, 0, 0, 1, 2, 2, 1, 0, 0, 0, 1, 2, 0,
                0, 1, 0, 2, 2, 0, 0, 2, 0, 2, 1, 0, 1, 0, 0, 2, 2, 2, 0, 0, 2, 1,
                0, 0, 2, 0, 3, 1, 0, 3, 0, 2, 2, 2, 2, 0, 2, 0, 0, 0, 1, 2, 0, 1,
                3, 3, 1, 1, 2, 3, 3, 3, 2, 3, 2, 2, 2, 0, 0, 3, 1, 0, 1, 0, 1, 1,
                0, 1], dtype=int64)
```

Confusion matrix



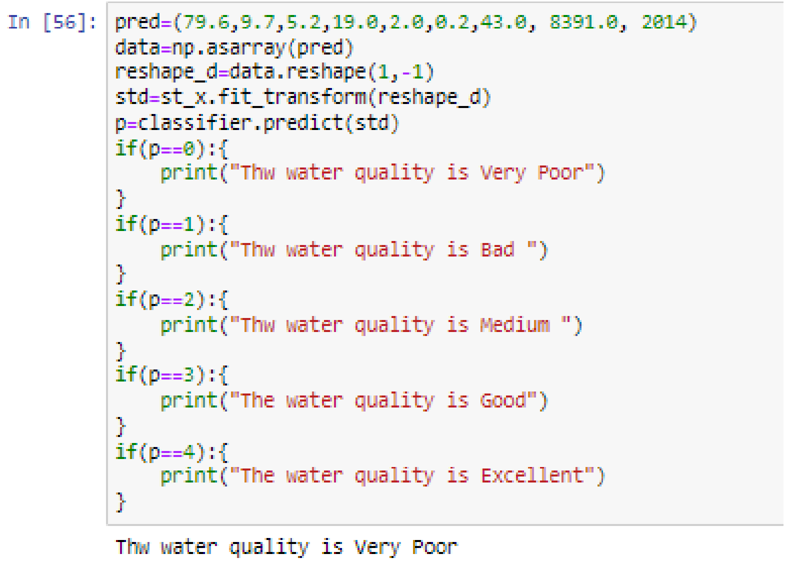
```
In [53]: #Creating the Confusion matrix
         from sklearn.metrics import confusion_matrix
         cm= confusion_matrix(y_test, y_pred)
```

Accuracy score of Test dataset

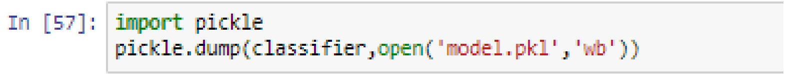


```
In [55]: from sklearn.metrics import accuracy_score, classification_report
         accuracy_score(y_test, y_pred)

Out[55]: 0.8512931034482759
```

Prediction of WQI



```
In [56]: pred=(79.6,9.7,5.2,19.0,2.0,0.2,43.0, 8391.0, 2014)
         data=np.asarray(pred)
         reshape_d=data.reshape(1,-1)
         std=st_x.fit_transform(reshape_d)
         p=classifier.predict(std)
         if(p==0):{
             print("Thw water quality is Very Poor")
         }
         if(p==1):{
             print("Thw water quality is Bad ")
         }
         if(p==2):{
             print("Thw water quality is Medium ")
         }
         if(p==3):{
             print("The water quality is Good")
         }
         if(p==4):{
             print("The water quality is Excellent")
         }

         Thw water quality is Very Poor
```

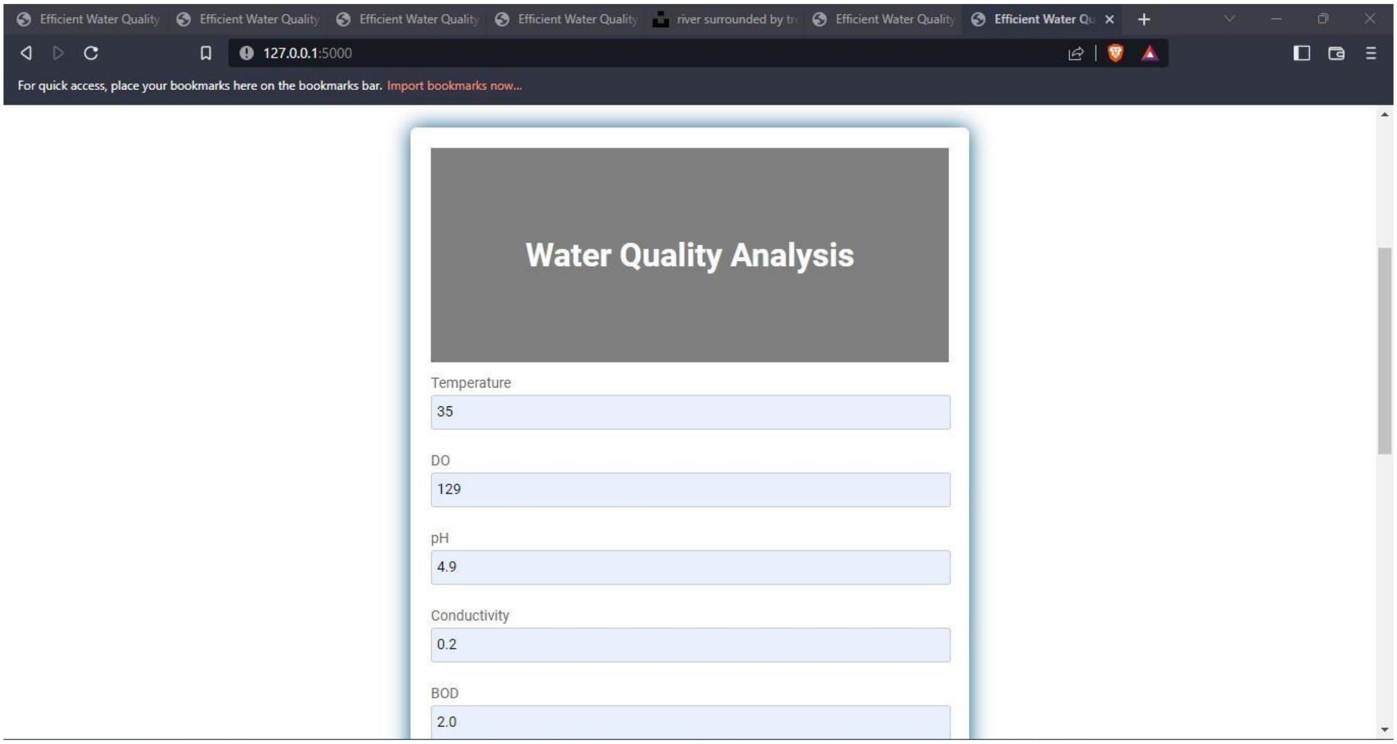


```
In [57]: import pickle
         pickle.dump(classifier,open('model.pkl','wb'))
```

CHAPTER 8

TESTING

8.1 TEST CASES (To predict water quality for input parameters – SUCCESS)



Water Quality Analysis

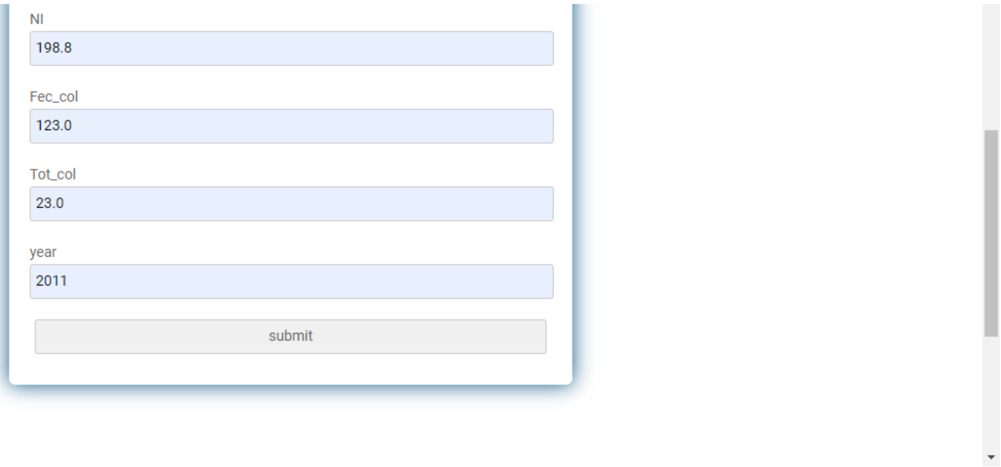
Temperature
35

DO
129

pH
4.9

Conductivity
0.2

BOD
2.0

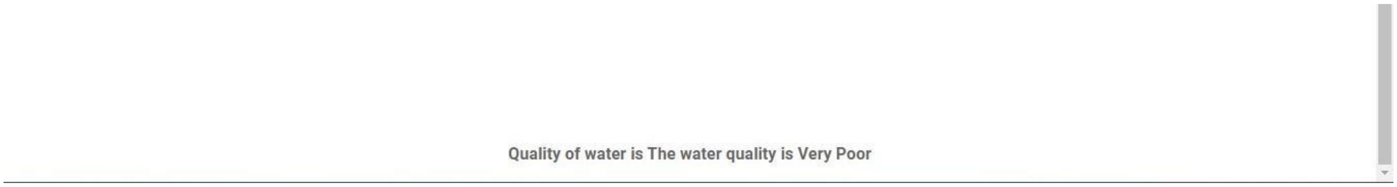


NI
198.8

Fec_col
123.0

Tot_col
23.0

year
2011



Quality of water is The water quality is Very Poor

8.2 USER ACCEPTANCE TESTING

Purpose

The purpose of this document is to briefly explain the test coverage and open issues of the Efficient Water Quality Analysis and Prediction using Machine Learning project at the time of the release to User Acceptance Testing (UAT).

8.2.1 DEFECT ANALYSIS

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	9	3	2	1	15
Duplicate	1	0	0	0	1
External	2	1	0	1	4
Fixed	6	5	4	25	40
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	3	2	1	6
Totals	18	12	10	29	69

8.2.2 TEST CASE ANALYSIS

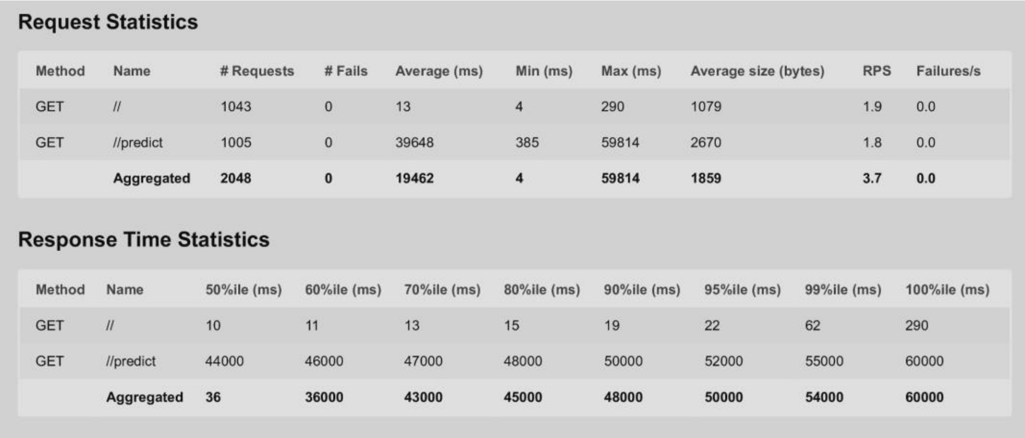
This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	48	0	0	48
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	5	0	0	5
Final Report Output	4	0	0	4
Version Control	2	0	0	2

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS

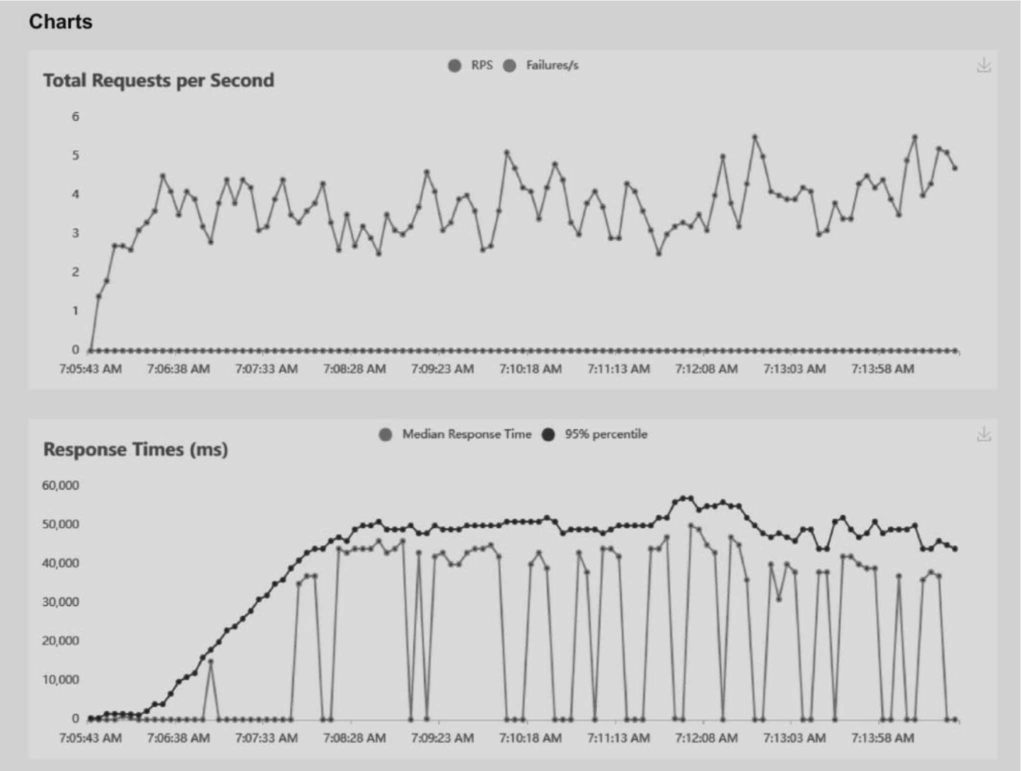


Request Statistics

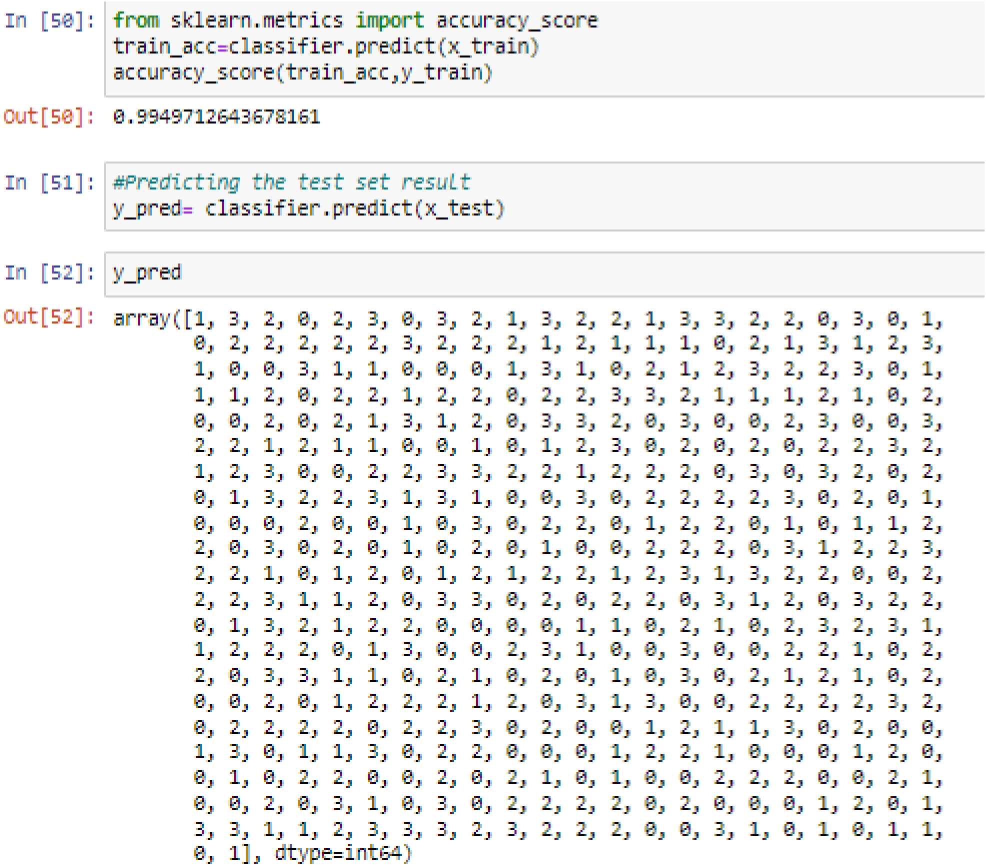
Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	//	1043	0	13	4	290	1079	1.9	0.0
GET	//predict	1005	0	39648	385	59814	2670	1.8	0.0
	Aggregated	**2048**	**0**	**19462**	**4**	**59814**	**1859**	**3.7**	**0.0**

Response Time Statistics

Method	Name	50%ile (ms)	60%ile (ms)	70%ile (ms)	80%ile (ms)	90%ile (ms)	95%ile (ms)	99%ile (ms)	100%ile (ms)
GET	//	10	11	13	15	19	22	62	290
GET	//predict	44000	46000	47000	48000	50000	52000	55000	60000
	Aggregated	**36**	**36000**	**43000**	**45000**	**48000**	**50000**	**54000**	**60000**

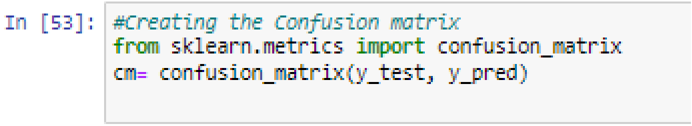


Accuracy score of training dataset



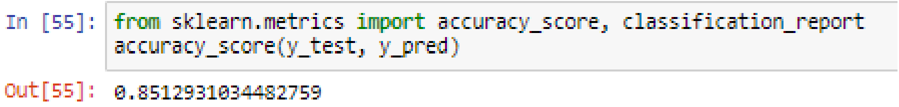
```
In [50]: from sklearn.metrics import accuracy_score
         train_acc=classifier.predict(x_train)
         accuracy_score(train_acc,y_train)

Out[50]: 0.9949712643678161

In [51]: #Predicting the test set result
         y_pred= classifier.predict(x_test)

In [52]: y_pred

Out[52]: array([1, 3, 2, 0, 2, 3, 0, 3, 2, 1, 3, 2, 2, 1, 3, 3, 2, 2, 0, 3, 0, 1,
                0, 2, 2, 2, 2, 2, 3, 2, 2, 2, 1, 2, 1, 1, 1, 0, 2, 1, 3, 1, 2, 3,
                1, 0, 0, 3, 1, 1, 0, 0, 0, 1, 3, 1, 0, 2, 1, 2, 3, 2, 2, 3, 0, 1,
                1, 1, 2, 0, 2, 2, 1, 2, 2, 0, 2, 2, 3, 3, 2, 1, 1, 1, 2, 1, 0, 2,
                0, 0, 2, 0, 2, 1, 3, 1, 2, 0, 3, 3, 2, 0, 3, 0, 0, 2, 3, 0, 0, 3,
                2, 2, 1, 2, 1, 1, 0, 0, 1, 0, 1, 2, 3, 0, 2, 0, 2, 0, 2, 2, 3, 2,
                1, 2, 3, 0, 0, 2, 2, 3, 3, 2, 2, 1, 2, 2, 2, 0, 3, 0, 3, 2, 0, 2,
                0, 1, 3, 2, 2, 3, 1, 3, 1, 0, 0, 3, 0, 2, 2, 2, 2, 3, 0, 2, 0, 1,
                0, 0, 0, 2, 0, 0, 1, 0, 3, 0, 2, 2, 0, 1, 2, 2, 0, 1, 0, 1, 1, 2,
                2, 0, 3, 0, 2, 0, 1, 0, 2, 0, 1, 0, 0, 2, 2, 2, 0, 3, 1, 2, 2, 3,
                2, 2, 1, 0, 1, 2, 0, 1, 2, 1, 2, 2, 1, 2, 3, 1, 3, 2, 2, 0, 0, 2,
                2, 2, 3, 1, 1, 2, 0, 3, 3, 0, 2, 0, 2, 2, 0, 3, 1, 2, 0, 3, 2, 2,
                0, 1, 3, 2, 1, 2, 2, 0, 0, 0, 0, 1, 1, 0, 2, 1, 0, 2, 3, 2, 3, 1,
                1, 2, 2, 2, 0, 1, 3, 0, 0, 2, 3, 1, 0, 0, 3, 0, 0, 2, 2, 1, 0, 2,
                2, 0, 3, 3, 1, 1, 0, 2, 1, 0, 2, 0, 1, 0, 3, 0, 2, 1, 2, 1, 0, 2,
                0, 0, 2, 0, 1, 2, 2, 2, 1, 2, 0, 3, 1, 3, 0, 0, 2, 2, 2, 2, 3, 2,
                0, 2, 2, 2, 2, 0, 2, 2, 3, 0, 2, 0, 0, 1, 2, 1, 1, 3, 0, 2, 0, 0,
                1, 3, 0, 1, 1, 3, 0, 2, 2, 0, 0, 0, 1, 2, 2, 1, 0, 0, 0, 1, 2, 0,
                0, 1, 0, 2, 2, 0, 0, 2, 0, 2, 1, 0, 1, 0, 0, 2, 2, 2, 0, 0, 2, 1,
                0, 0, 2, 0, 3, 1, 0, 3, 0, 2, 2, 2, 2, 0, 2, 0, 0, 0, 1, 2, 0, 1,
                3, 3, 1, 1, 2, 3, 3, 3, 2, 3, 2, 2, 2, 0, 0, 3, 1, 0, 1, 0, 1, 1,
                0, 1], dtype=int64)
```

Confusion matrix



```
In [53]: #Creating the Confusion matrix
         from sklearn.metrics import confusion_matrix
         cm= confusion_matrix(y_test, y_pred)
```

Accuracy score of Test dataset



```
In [55]: from sklearn.metrics import accuracy_score, classification_report
         accuracy_score(y_test, y_pred)

Out[55]: 0.8512931034482759
```

CHAPTER 10
ADVANTAGES & DISADVANTAGES

ADVANTAGES

- water quality prediction helps in controlling Water Pollution
- To predict the water is safe or not
- Predicting potable water quality for water management and water pollution prevention.
- Water quality prediction convey the health of ecosystems, safety of human contact, extend of water pollution and condition of drinking water

DISADVANTAGES

- Training necessary Somewhat difficult to manage over time and with large data sets
- Requires manual operation to submit data, some configuration required
- Costly, usually only feasible under Exchange Network grants Technical expertise and network server required
- Requires manual operation to submit data Cannot respond to data queries from other nodes, and therefore cannot interact with the Exchange Network Technical expertise and network server required

CHAPTER 11

CONCLUSION

The water quality is monitored and managed effectively because of the importance of drinking water. Water has a direct effect on our health. This adds more reason to test the quality of drinking water. Several boards of committees and protocols are established to check the quality of water. The assessment of water quality differs from origin to origin. Using machine learning techniques the water quality is tested without any regular laboratory tests. By using Random Forest algorithm, we can evaluate the quality of water based on the attributes such as pH, BOD, DO, minerals and coliform in the water.This model can be used for predicting the quality of water and can monitor the potability of the water. This model acts as a prototype for the IoT sensors and can make the model even more efficient to predict the quality of water and potability of water. Data cleaning and processing, missing value analysis, exploratory analysis, and model creation and evaluation were all part of the analytical process. The best accuracy on a public test set will be discovered, as will the highest accuracy score. This application can assist in determining the current state of water quality.

CHAPTER 12

FUTURE SCOPE

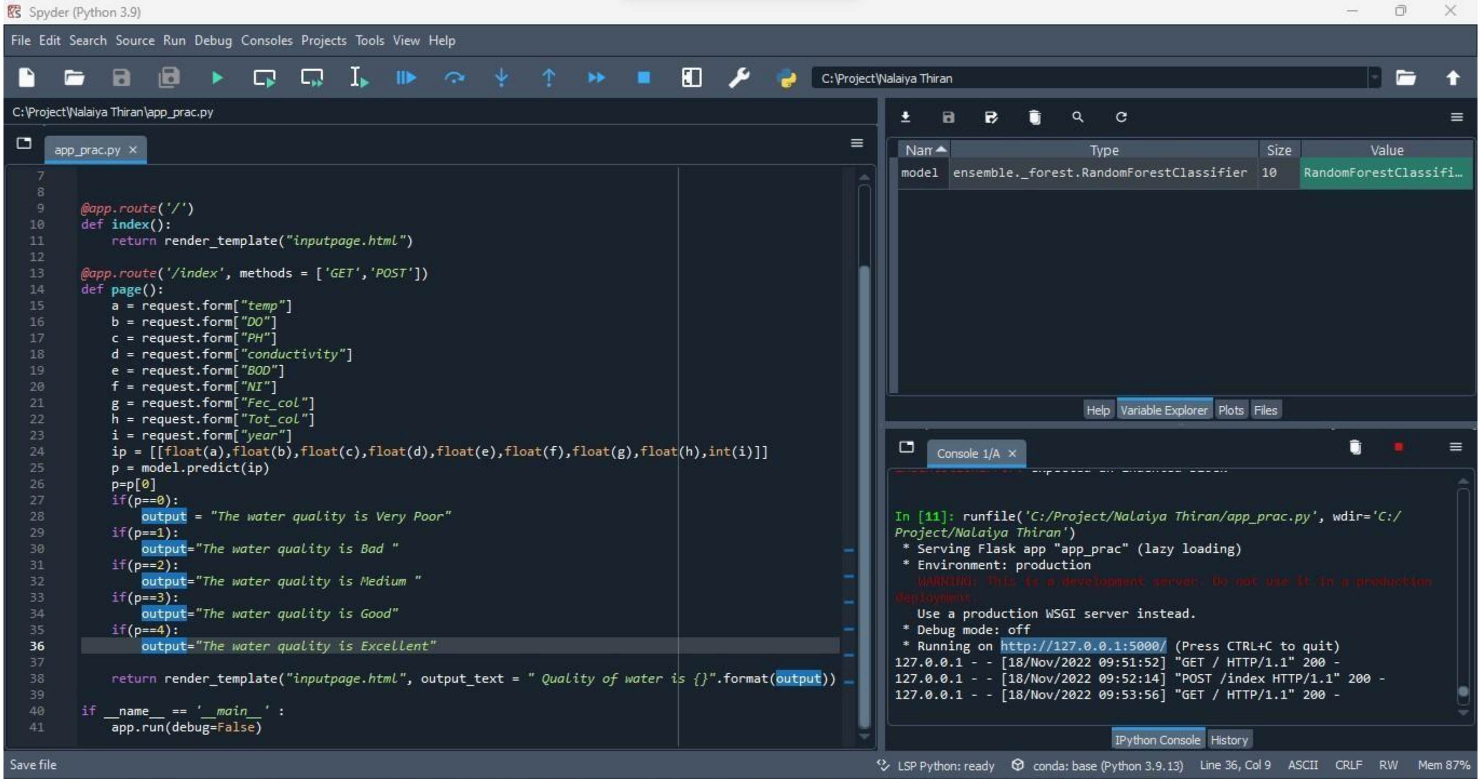
In future works, we propose integrating the findings of this research in a large-scale IoT-based online monitoring system using only the sensors of the required parameters. The tested algorithms would predict the water quality immediately based on the real-time data fed from the IoT system.The proposed IoT system would employ the parameter sensors of pH, turbidity, temperature and TDS for parameter readings and communicate those readings using an Arduino microcontroller. 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 makers.

APPENDIX

REQUIREMENTS

```
Flask == 2.2.2
numpy == 1.23.4
pandas == 1.5.1
scikit-learn == 1.1.3
matplotlib == 3.6.2
seaborn == 0.12.1
```

FLASK APP

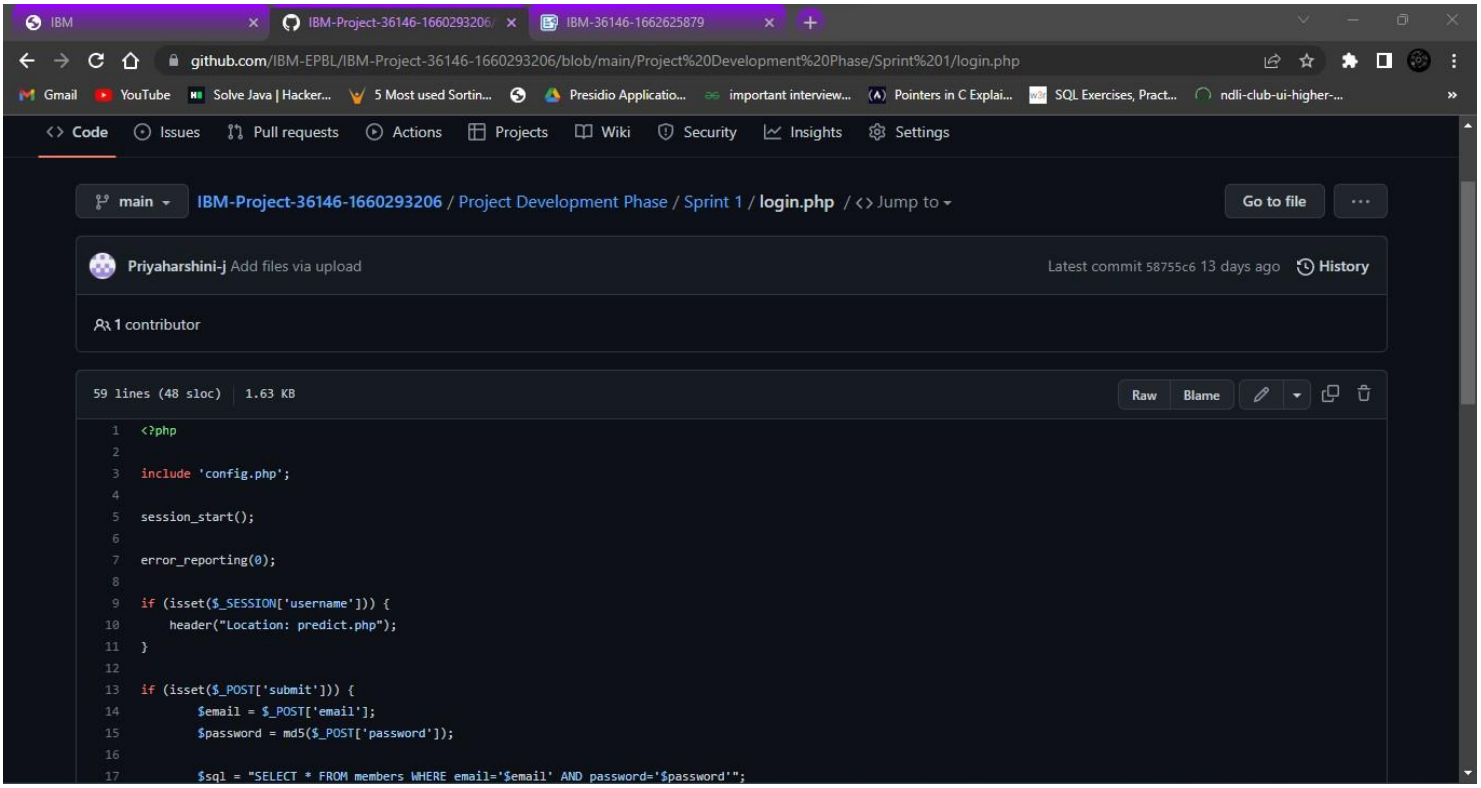

Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Project\Nalaiya Thiran
C:\Project\Nalaiya Thiran\app_prac.py
app_prac.py
Help
Variable Explorer
Plots
Files
Console 1/A
IPython Console
History
Save file
LSP Python: ready
conda: base (Python 3.9.13)
Line 36, Col 9
ASCII
CRLF
RW
Mem 87%

```python


@app.route('/')
def index():
    return render_template("inputpage.html")

@app.route('/index', methods = ['GET','POST'])
def page():
    a = request.form["temp"]
    b = request.form["DO"]
    c = request.form["PH"]
    d = request.form["conductivity"]
    e = request.form["BOD"]
    f = request.form["NI"]
    g = request.form["Fec_col"]
    h = request.form["Tot_col"]
    i = request.form["year"]
    ip = [[float(a),float(b),float(c),float(d),float(e),float(f),float(g),float(h),int(i)]]
    p = model.predict(ip)
    p=p[0]
    if(p==0):
        output = "The water quality is Very Poor"
    if(p==1):
        output="The water quality is Bad "
    if(p==2):
        output="The water quality is Medium "
    if(p==3):
        output="The water quality is Good"
    if(p==4):
        output="The water quality is Excellent"

    return render_template("inputpage.html", output_text = " Quality of water is {}".format(output))

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

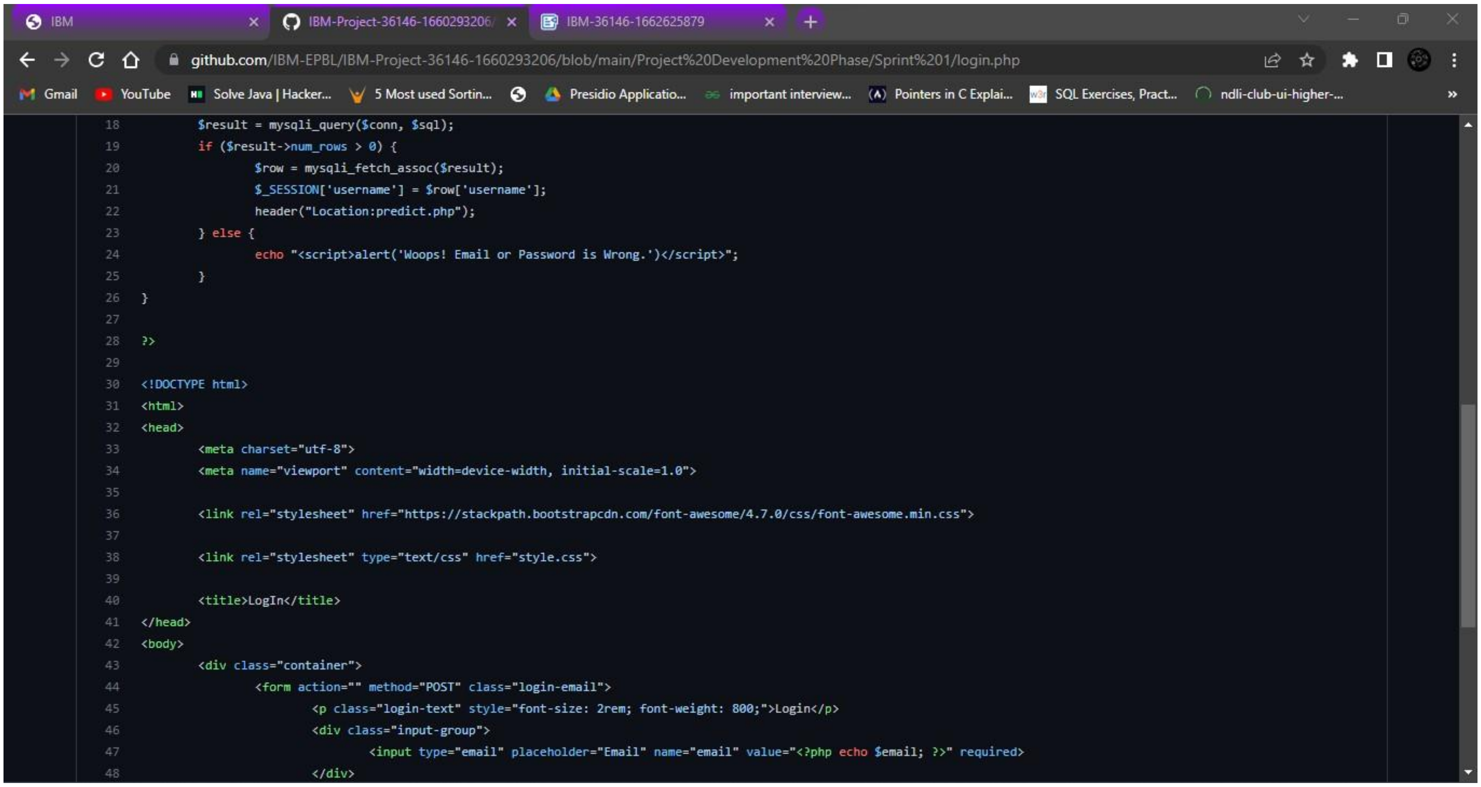
Name	Type	Size	Value
model	ensemble._forest.RandomForestClassifier	10	RandomForestClassifi…

```
In [11]: runfile('C:/Project/Nalaiya Thiran/app_prac.py', wdir='C:/
Project/Nalaiya Thiran')
 * Serving Flask app "app_prac" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production
deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [18/Nov/2022 09:51:52] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [18/Nov/2022 09:52:14] "POST /index HTTP/1.1" 200 -
127.0.0.1 - - [18/Nov/2022 09:53:56] "GET / HTTP/1.1" 200 -
```

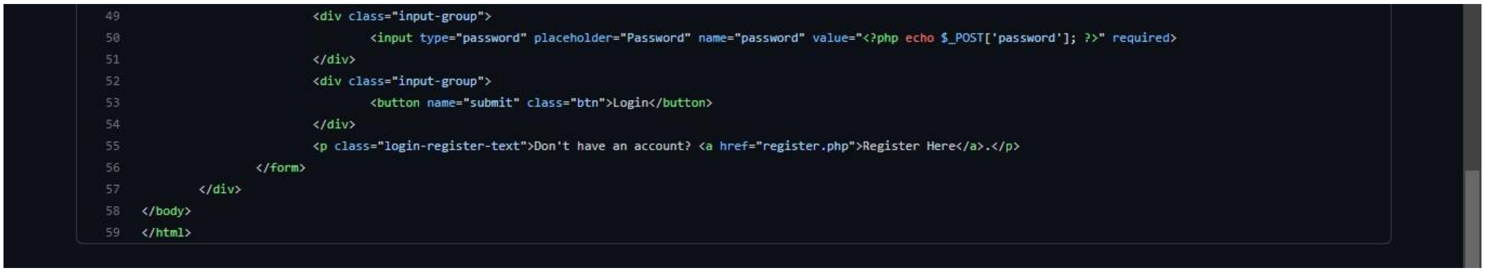
LOGIN PAGE


IBM
IBM-Project-36146-1660293206/
IBM-36146-1662625879
github.com/IBM-EPBL/IBM-Project-36146-1660293206/blob/main/Project%20Development%20Phase/Sprint%201/login.php
Gmail
YouTube
Solve Java | Hacker...
5 Most used Sortin...
Presidio Applicatio...
important interview...
Pointers in C Explai...
SQL Exercises, Pract...
ndli-club-ui-higher-...
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
main
IBM-Project-36146-1660293206 / Project Development Phase / Sprint 1 / login.php / Jump to
Go to file
Priyaharshini-j Add files via upload
Latest commit 58755c6 13 days ago
History
1 contributor
59 lines (48 sloc) | 1.63 KB
Raw
Blame

```php
<?php

include 'config.php';

session_start();

error_reporting(0);

if (isset($_SESSION['username'])) {
    header("Location: predict.php");
}

if (isset($_POST['submit'])) {
        $email = $_POST['email'];
        $password = md5($_POST['password']);

        $sql = "SELECT * FROM members WHERE email='$email' AND password='$password'";
```

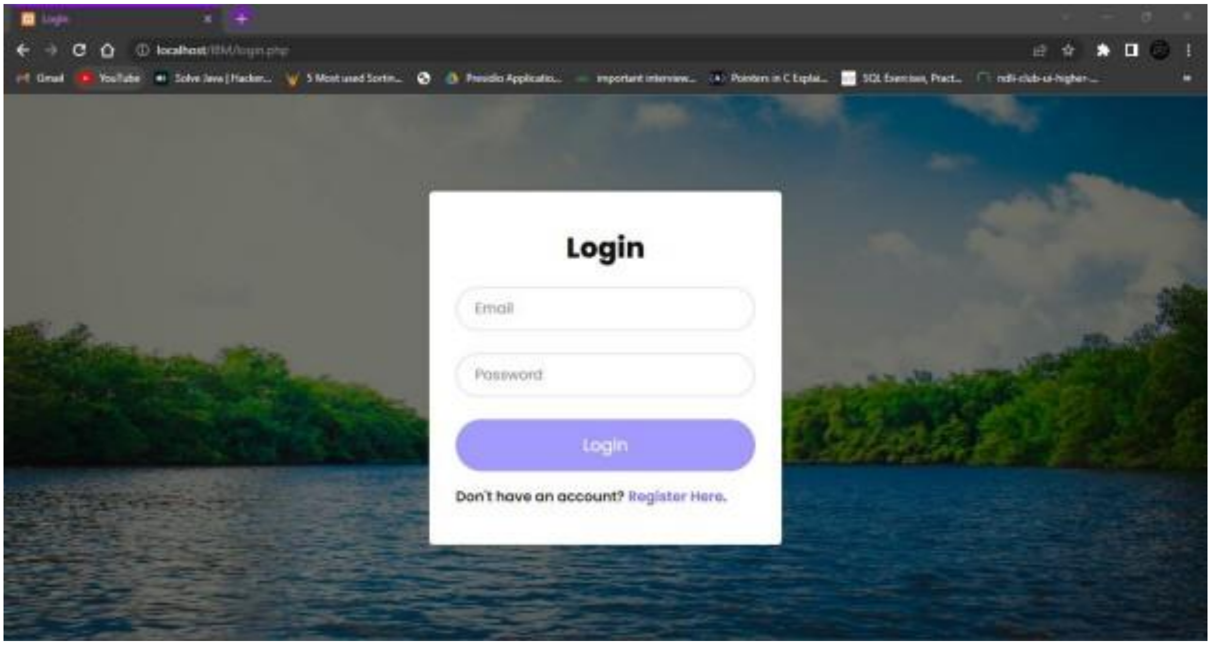


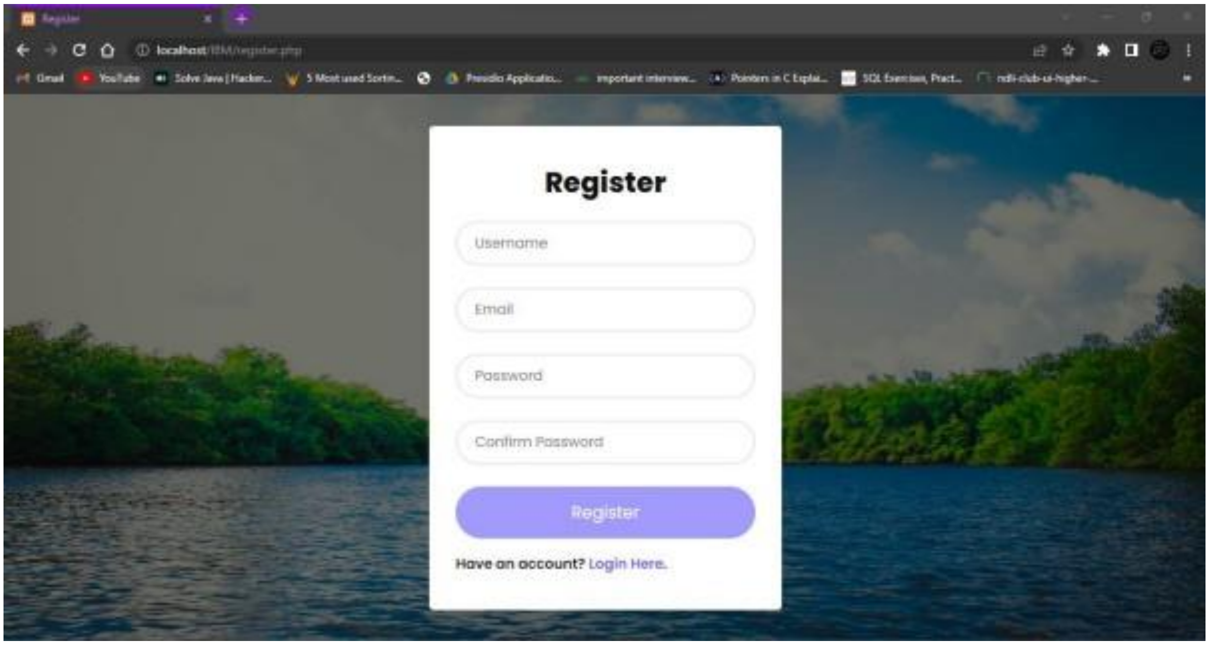
```
        $result = mysqli_query($conn, $sql);
        if ($result->num_rows > 0) {
                $row = mysqli_fetch_assoc($result);
                $_SESSION['username'] = $row['username'];
                header("Location:predict.php");
        } else {
                echo "<script>alert('Woops! Email or Password is Wrong.')</script>";
        }
}

?>

<!DOCTYPE html>
<html>
<head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">

        <link rel="stylesheet" type="text/css" href="style.css">

        <title>LogIn</title>
</head>
<body>
        <div class="container">
                <form action="" method="POST" class="login-email">
                        <p class="login-text" style="font-size: 2rem; font-weight: 800;">Login</p>
                        <div class="input-group">
                                <input type="email" placeholder="Email" name="email" value="<?php echo $email; ?>" required>
                        </div>
```



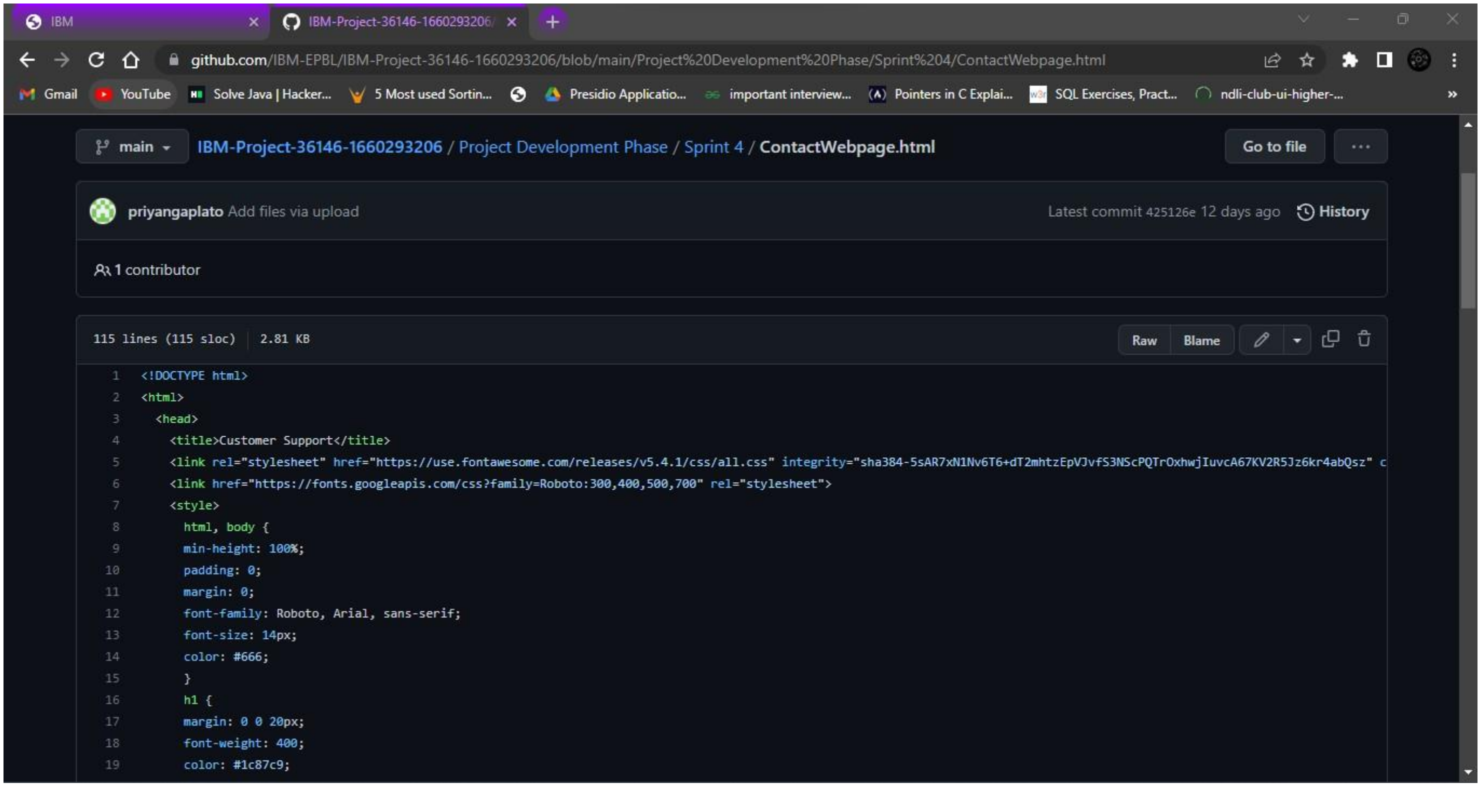
```
                        <div class="input-group">
                                <input type="password" placeholder="Password" name="password" value="<?php echo $_POST['password']; ?>" required>
                        </div>
                        <div class="input-group">
                                <button name="submit" class="btn">Login</button>
                        </div>
                        <p class="login-register-text">Don't have an account? <a href="register.php">Register Here</a>.</p>
                </form>
        </div>
</body>
</html>
```

LOGIN(OUTPUT)

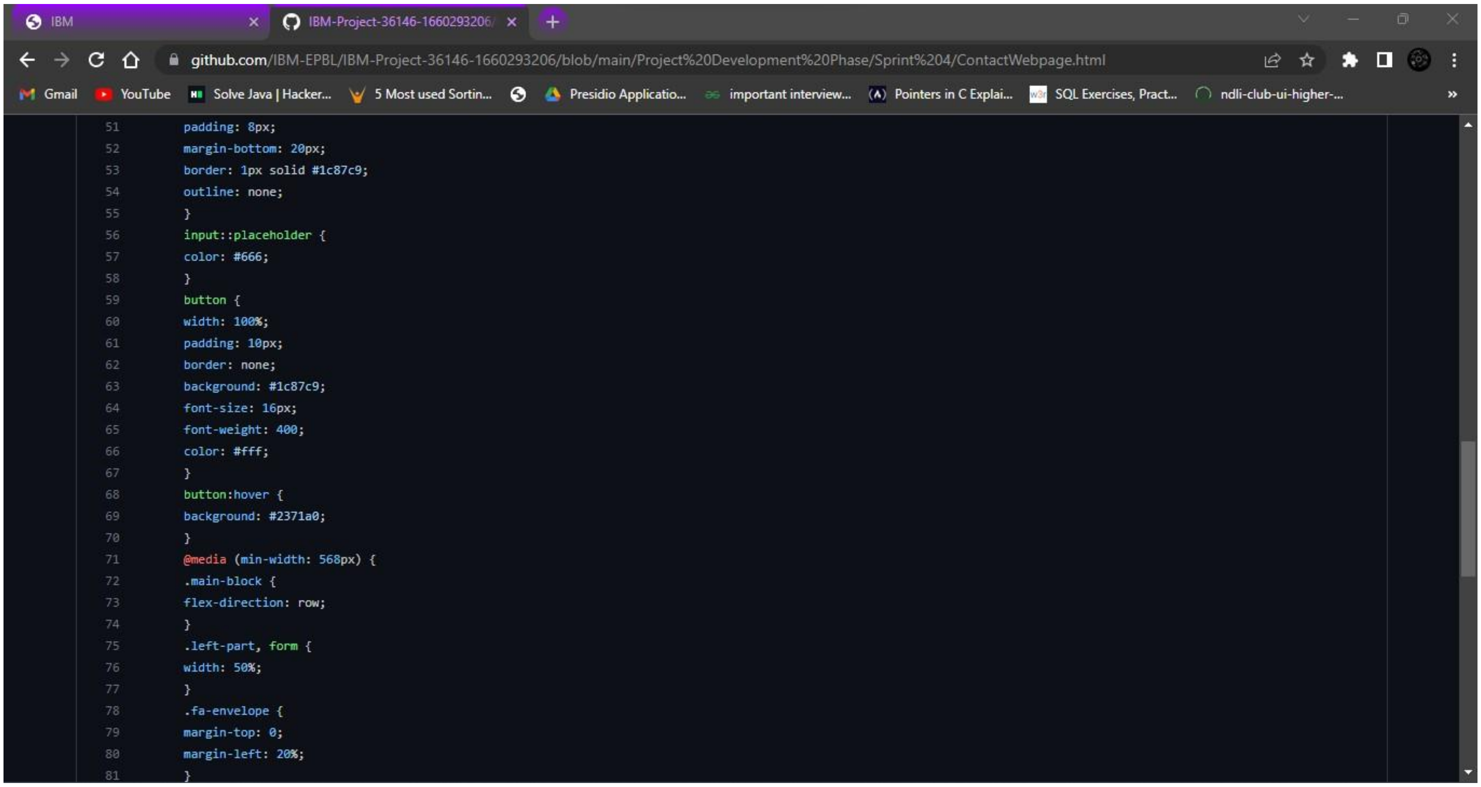




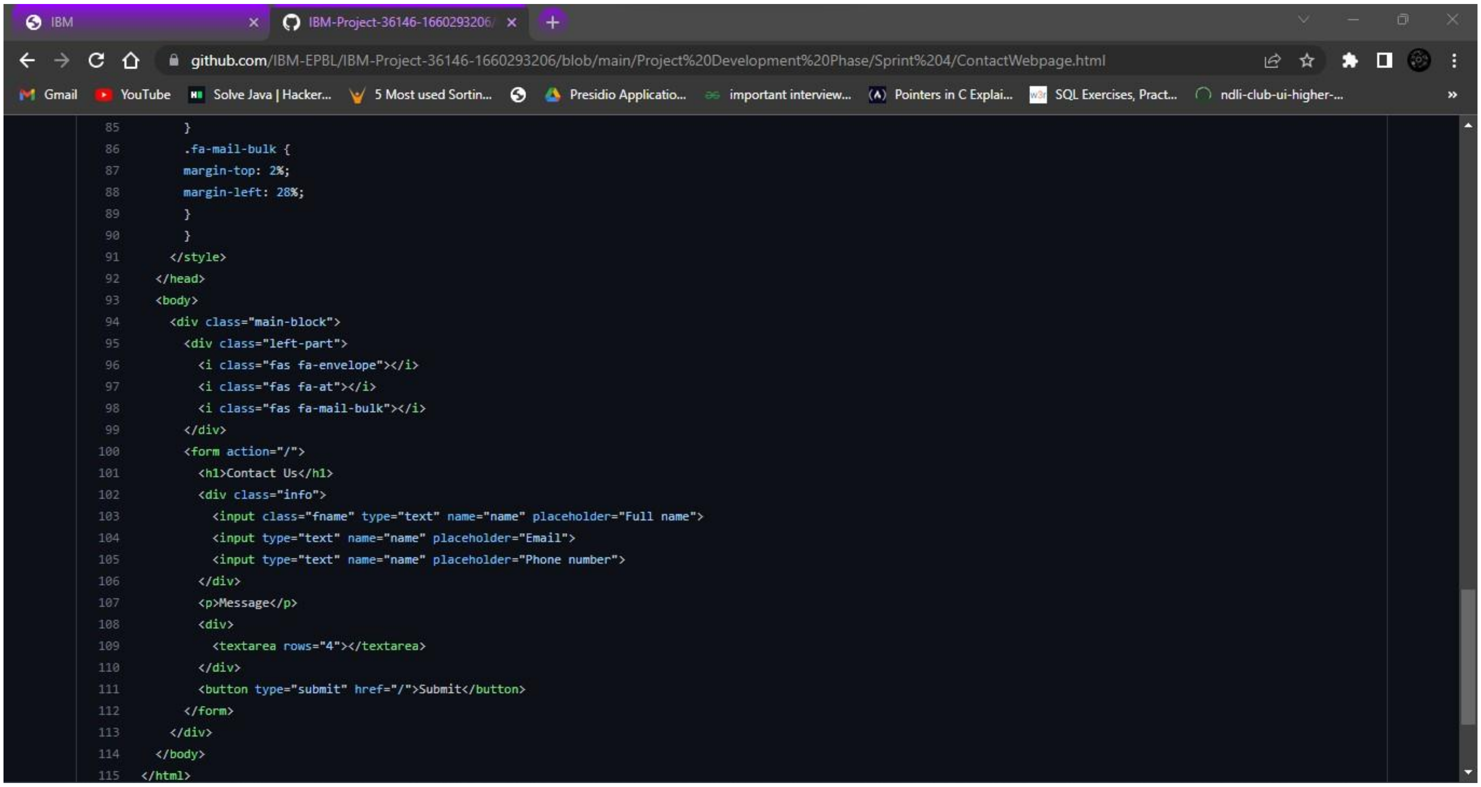
CONTACT PAGE (HTML)


IBM
IBM-Project-36146-1660293206
github.com/IBM-EPBL/IBM-Project-36146-1660293206/blob/main/Project%20Development%20Phase/Sprint%204/ContactWebpage.html
Gmail
YouTube
Solve Java | Hacker...
5 Most used Sortin...
Presidio Applicatio...
important interview...
Pointers in C Explai...
SQL Exercises, Pract...
ndli-club-ui-higher-...
main
IBM-Project-36146-1660293206 / Project Development Phase / Sprint 4 / ContactWebpage.html
Go to file
priyangaplato Add files via upload
Latest commit 425126e 12 days ago
History
1 contributor
115 lines (115 sloc) | 2.81 KB
Raw
Blame

```
<!DOCTYPE html>
<html>
  <head>
    <title>Customer Support</title>
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.4.1/css/all.css" integrity="sha384-5sAR7xN1Nv6T6+dT2mhtzEpVJvfS3NScPQTrOxhwjIuvcA67KV2R5Jz6kr4abQsz" c
    <link href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700" rel="stylesheet">
    <style>
      html, body {
      min-height: 100%;
      padding: 0;
      margin: 0;
      font-family: Roboto, Arial, sans-serif;
      font-size: 14px;
      color: #666;
      }
      h1 {
      margin: 0 0 20px;
      font-weight: 400;
      color: #1c87c9;
```

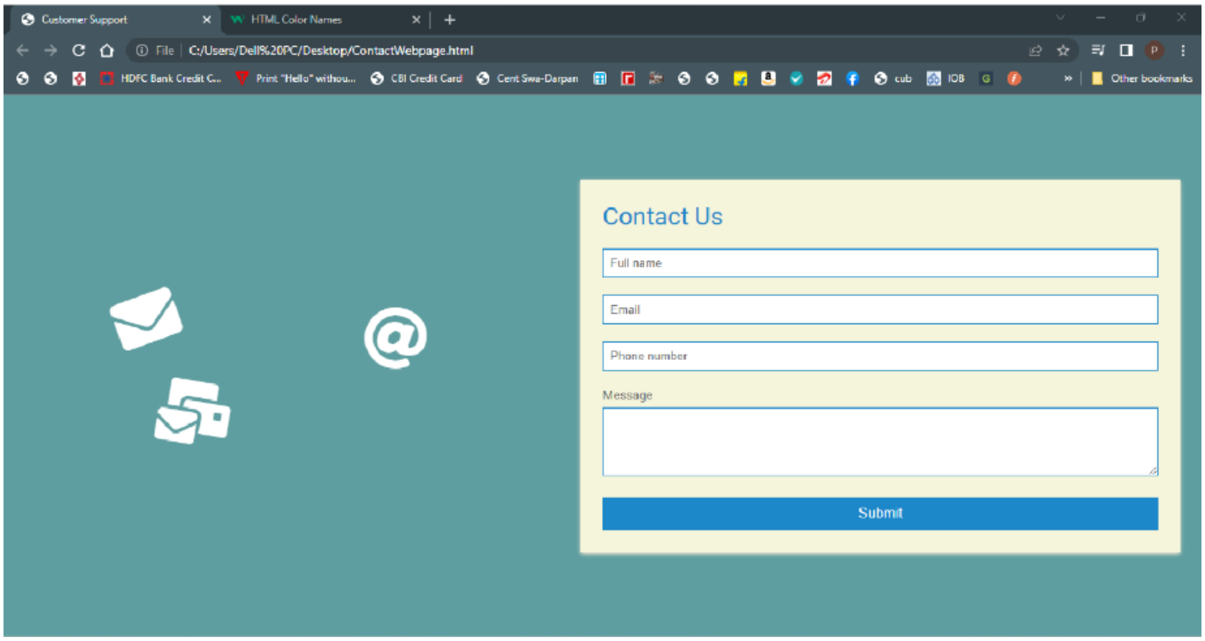


```
      padding: 8px;
      margin-bottom: 20px;
      border: 1px solid #1c87c9;
      outline: none;
      }
      input::placeholder {
      color: #666;
      }
      button {
      width: 100%;
      padding: 10px;
      border: none;
      background: #1c87c9;
      font-size: 16px;
      font-weight: 400;
      color: #fff;
      }
      button:hover {
      background: #2371a0;
      }
      @media (min-width: 568px) {
      .main-block {
      flex-direction: row;
      }
      .left-part, form {
      width: 50%;
      }
      .fa-envelope {
      margin-top: 0;
      margin-left: 20%;
      }
```



```
      }
      .fa-mail-bulk {
      margin-top: 2%;
      margin-left: 28%;
      }
      }
    </style>
  </head>
  <body>
    <div class="main-block">
      <div class="left-part">
        <i class="fas fa-envelope"></i>
        <i class="fas fa-at"></i>
        <i class="fas fa-mail-bulk"></i>
      </div>
      <form action="/">
        <h1>Contact Us</h1>
        <div class="info">
          <input class="fname" type="text" name="name" placeholder="Full name">
          <input type="text" name="name" placeholder="Email">
          <input type="text" name="name" placeholder="Phone number">
        </div>
        <p>Message</p>
        <div>
          <textarea rows="4"></textarea>
        </div>
        <button type="submit" href="/">Submit</button>
      </form>
    </div>
  </body>
</html>
```

CONTACT PAGE (OUTPUT)



Contact Us

Full name

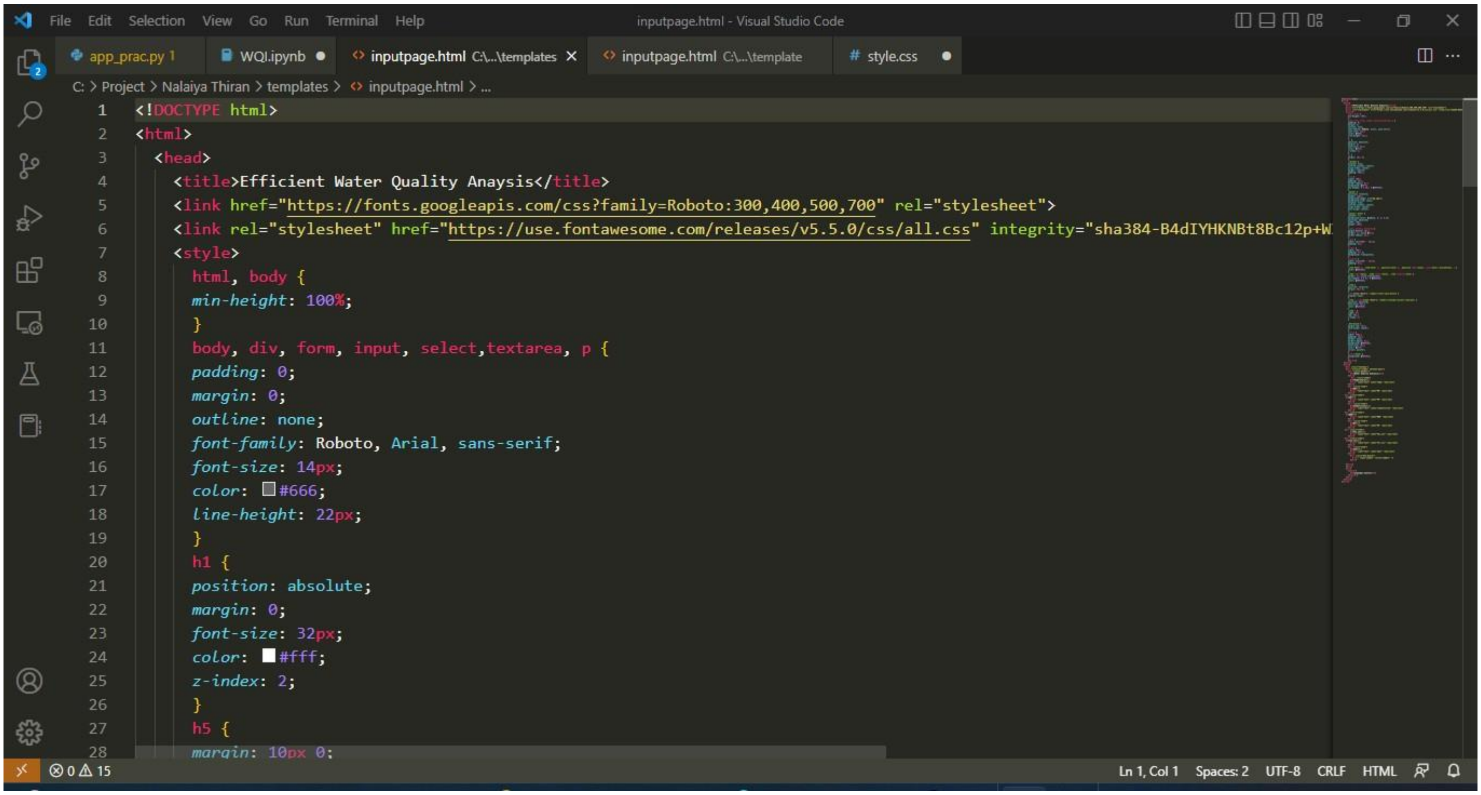
Email

Phone number

Message

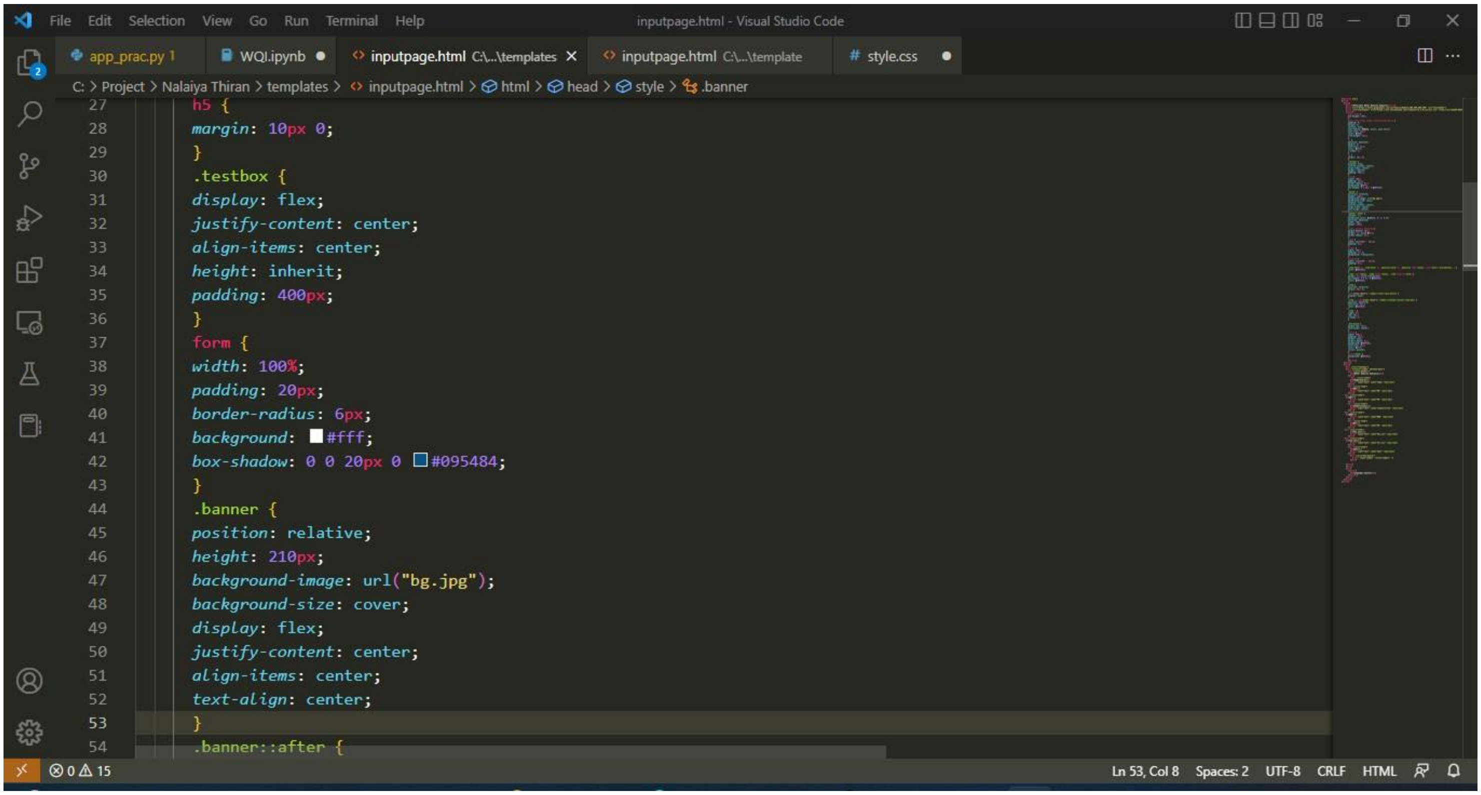
Submit

PREDICT PAGE (HTML)

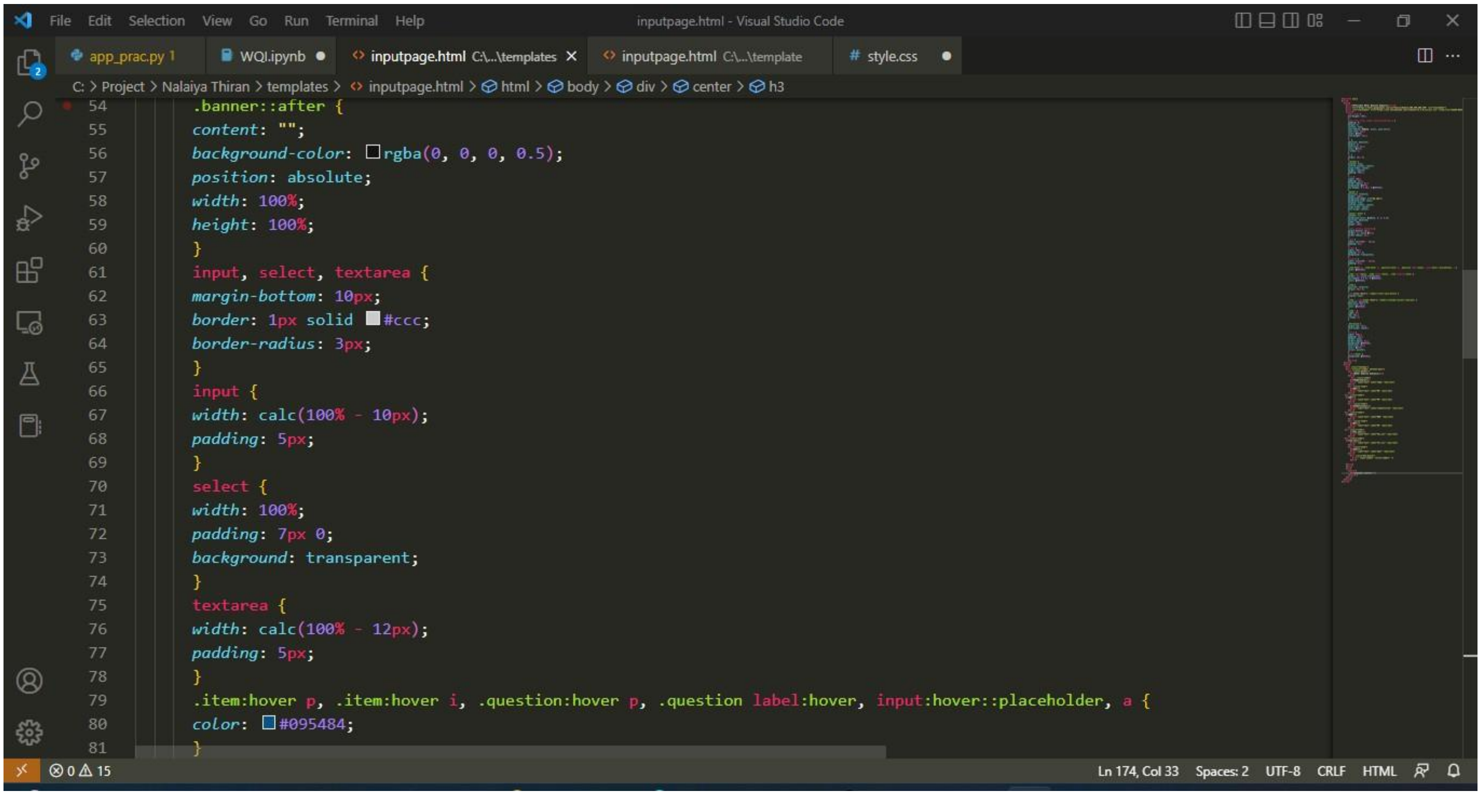


File
Edit
Selection
View
Go
Run
Terminal
Help
inputpage.html - Visual Studio Code
app_prac.py 1
WQI.ipynb
inputpage.html C:\...\templates
inputpage.html C:\...\template
style.css
C: > Project > Nalaiya Thiran > templates > inputpage.html > ...

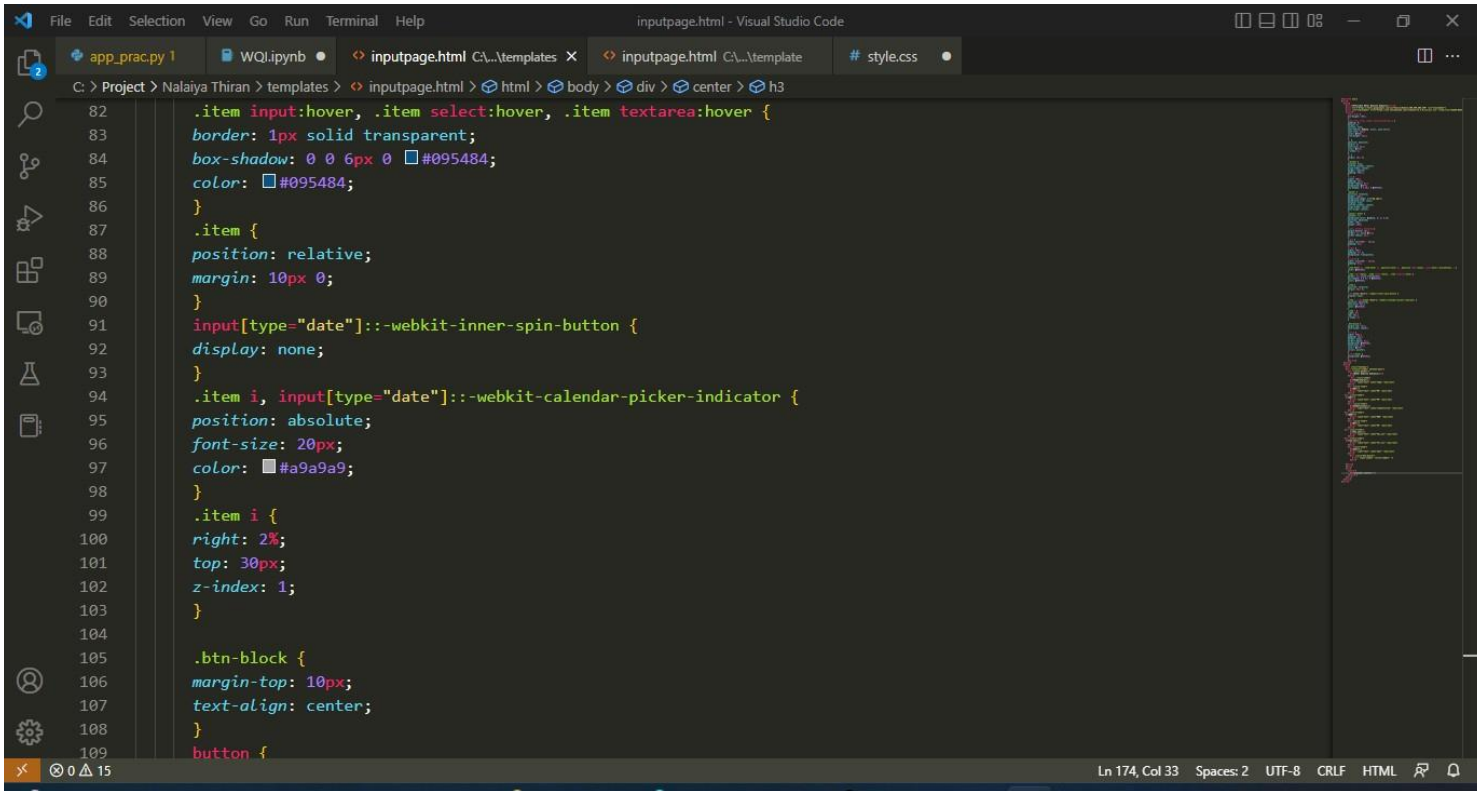
```
<!DOCTYPE html>
<html>
  <head>
    <title>Efficient Water Quality Anaysis</title>
    <link href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700" rel="stylesheet">
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.5.0/css/all.css" integrity="sha384-B4dIYHKNBt8Bc12p+W
    <style>
      html, body {
      min-height: 100%;
      }
      body, div, form, input, select,textarea, p {
      padding: 0;
      margin: 0;
      outline: none;
      font-family: Roboto, Arial, sans-serif;
      font-size: 14px;
      color: #666;
      line-height: 22px;
      }
      h1 {
      position: absolute;
      margin: 0;
      font-size: 32px;
      color: #fff;
      z-index: 2;
      }
      h5 {
      margin: 10px 0;
```


File Edit Selection View Go Run Terminal Help
inputpage.html - Visual Studio Code
app_prac.py 1
WQI.ipynb
inputpage.html C:\...\templates
inputpage.html C:\...\template
style.css
C: > Project > Nalaiya Thiran > templates > inputpage.html > html > head > style > .banner
⊗ 0 ⚠ 15
Ln 53, Col 8 Spaces: 2 UTF-8 CRLF HTML

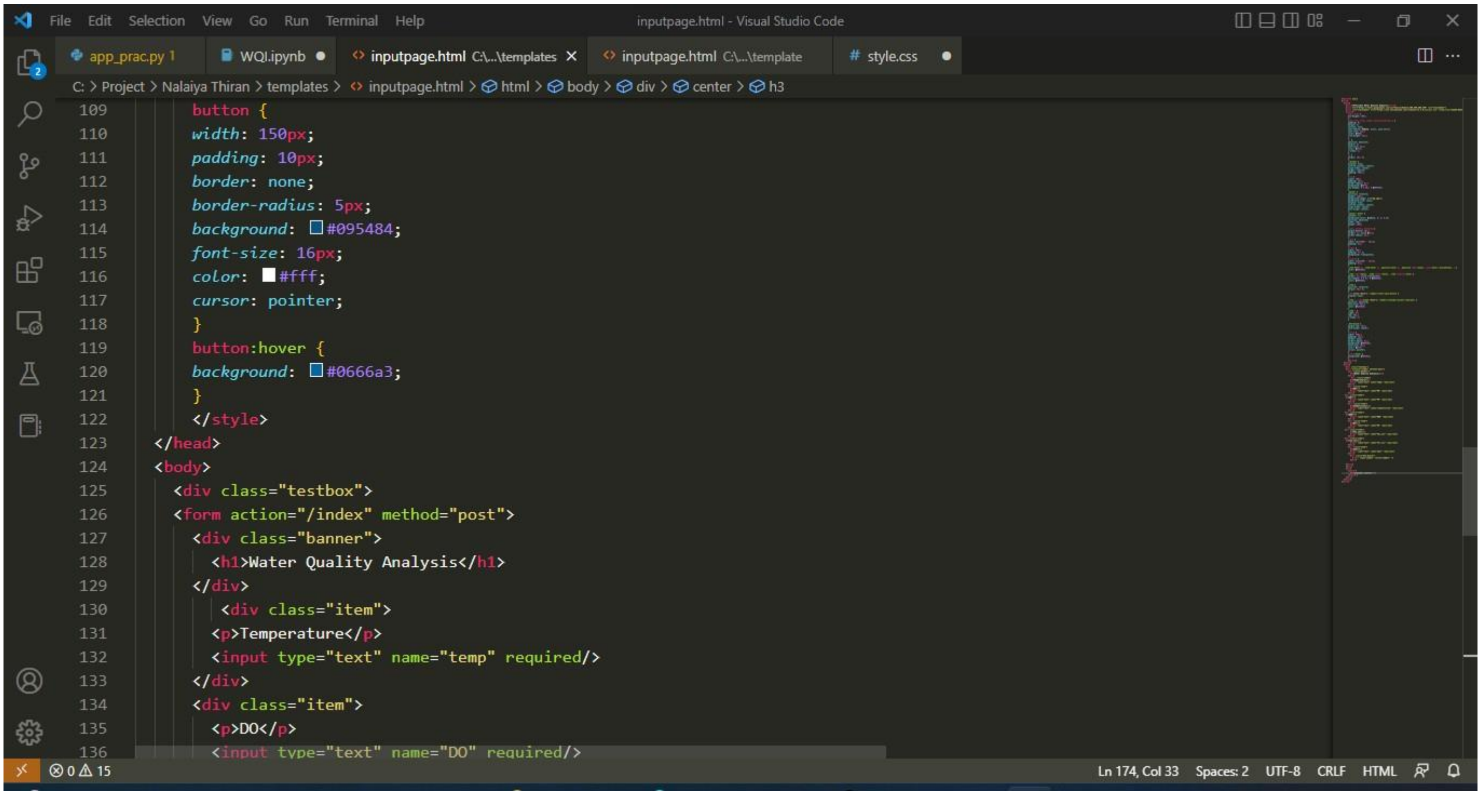
```
h5 {
margin: 10px 0;
}
.testbox {
display: flex;
justify-content: center;
align-items: center;
height: inherit;
padding: 400px;
}
form {
width: 100%;
padding: 20px;
border-radius: 6px;
background: #fff;
box-shadow: 0 0 20px 0 #095484;
}
.banner {
position: relative;
height: 210px;
background-image: url("bg.jpg");
background-size: cover;
display: flex;
justify-content: center;
align-items: center;
text-align: center;
}
.banner::after {
```


File Edit Selection View Go Run Terminal Help
inputpage.html - Visual Studio Code
app_prac.py 1
WQI.ipynb
inputpage.html C:\...\templates
inputpage.html C:\...\template
style.css
C: > Project > Nalaiya Thiran > templates > inputpage.html > html > body > div > center > h3
⊗ 0 ⚠ 15
Ln 174, Col 33 Spaces: 2 UTF-8 CRLF HTML

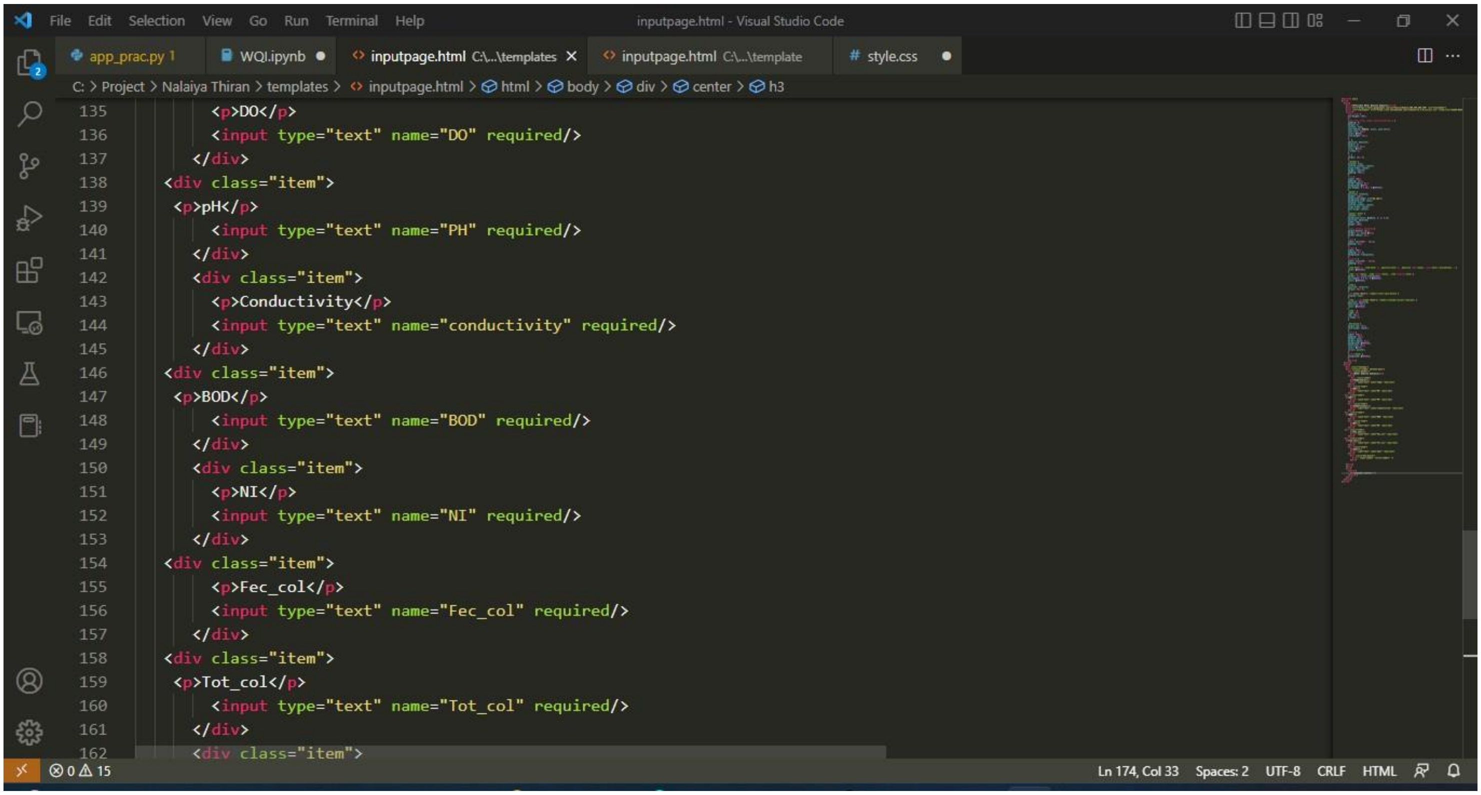
```
.banner::after {
content: "";
background-color: rgba(0, 0, 0, 0.5);
position: absolute;
width: 100%;
height: 100%;
}
input, select, textarea {
margin-bottom: 10px;
border: 1px solid #ccc;
border-radius: 3px;
}
input {
width: calc(100% - 10px);
padding: 5px;
}
select {
width: 100%;
padding: 7px 0;
background: transparent;
}
textarea {
width: calc(100% - 12px);
padding: 5px;
}
.item:hover p, .item:hover i, .question:hover p, .question label:hover, input:hover::placeholder, a {
color: #095484;
}
```


File Edit Selection View Go Run Terminal Help
inputpage.html - Visual Studio Code
app_prac.py 1
WQI.ipynb
inputpage.html C:\...\templates
inputpage.html C:\...\template
style.css
C: > Project > Nalaiya Thiran > templates > inputpage.html > html > body > div > center > h3
0 15
Ln 174, Col 33 Spaces: 2 UTF-8 CRLF HTML

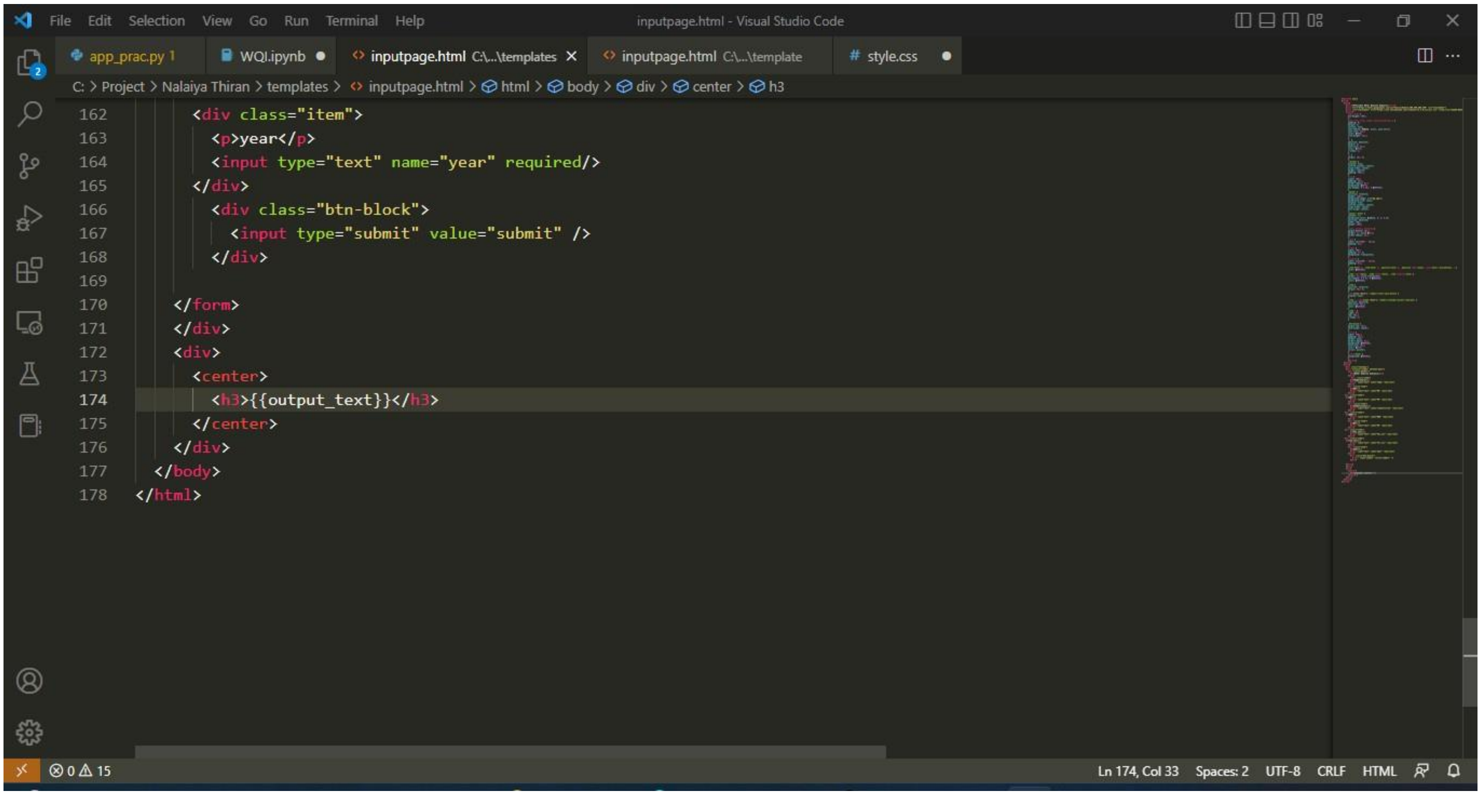
```
        .item input:hover, .item select:hover, .item textarea:hover {
        border: 1px solid transparent;
        box-shadow: 0 0 6px 0 #095484;
        color: #095484;
        }
        .item {
        position: relative;
        margin: 10px 0;
        }
        input[type="date"]::-webkit-inner-spin-button {
        display: none;
        }
        .item i, input[type="date"]::-webkit-calendar-picker-indicator {
        position: absolute;
        font-size: 20px;
        color: #a9a9a9;
        }
        .item i {
        right: 2%;
        top: 30px;
        z-index: 1;
        }

        .btn-block {
        margin-top: 10px;
        text-align: center;
        }
        button {
```


File Edit Selection View Go Run Terminal Help
inputpage.html - Visual Studio Code
app_prac.py 1
WQI.ipynb
inputpage.html C:\...\templates
inputpage.html C:\...\template
style.css
C: > Project > Nalaiya Thiran > templates > inputpage.html > html > body > div > center > h3
0 15
Ln 174, Col 33 Spaces: 2 UTF-8 CRLF HTML

```
        button {
        width: 150px;
        padding: 10px;
        border: none;
        border-radius: 5px;
        background: #095484;
        font-size: 16px;
        color: #fff;
        cursor: pointer;
        }
        button:hover {
        background: #0666a3;
        }
        </style>
    </head>
    <body>
      <div class="testbox">
      <form action="/index" method="post">
        <div class="banner">
          <h1>Water Quality Analysis</h1>
        </div>
           <div class="item">
          <p>Temperature</p>
          <input type="text" name="temp" required/>
        </div>
        <div class="item">
          <p>DO</p>
          <input type="text" name="DO" required/>
```

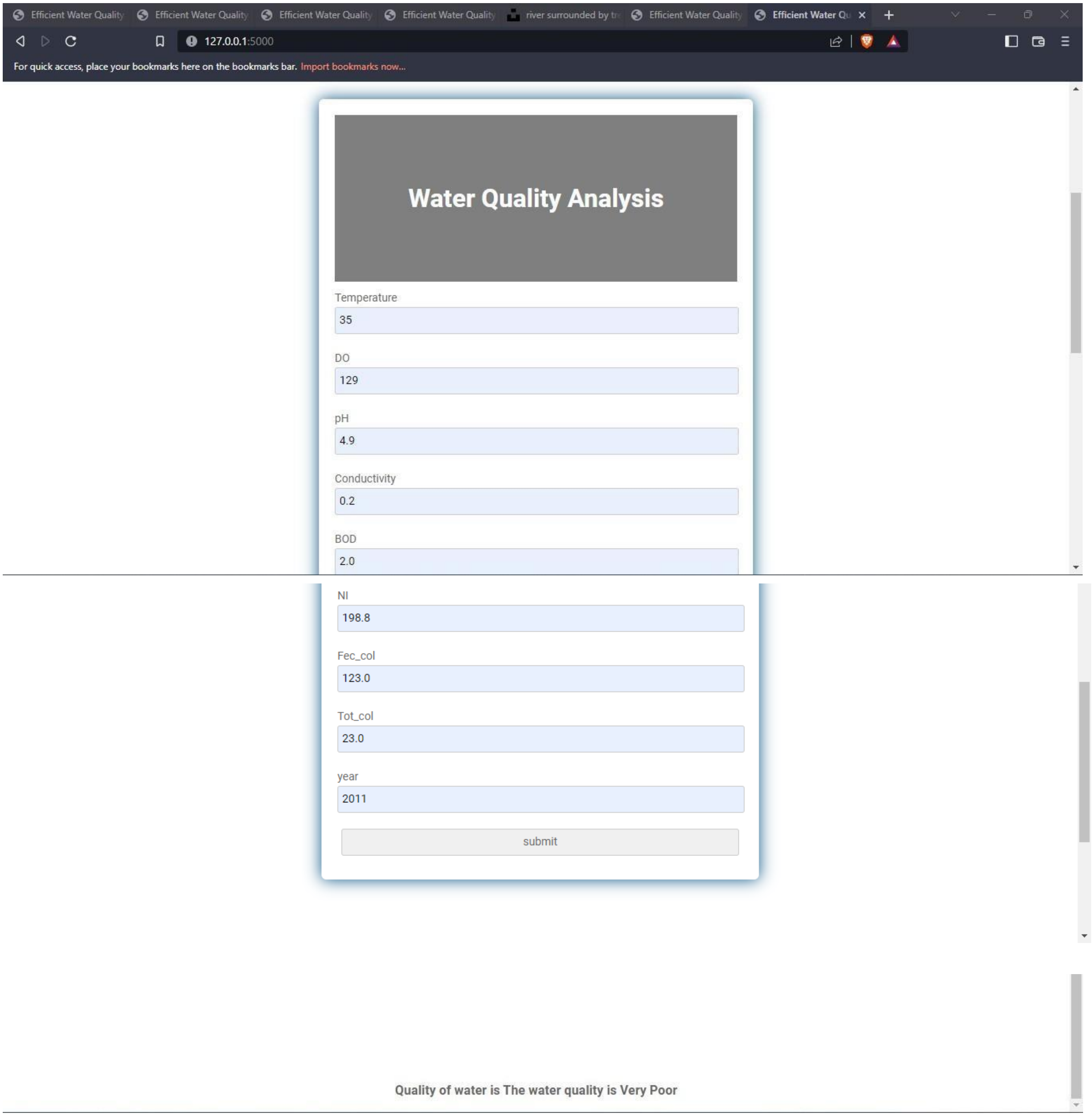

File Edit Selection View Go Run Terminal Help
inputpage.html - Visual Studio Code
app_prac.py 1
WQI.ipynb
inputpage.html C:\...\templates
inputpage.html C:\...\template
style.css
C: > Project > Nalaiya Thiran > templates > inputpage.html > html > body > div > center > h3
⊗ 0 ⚠ 15
Ln 174, Col 33 Spaces: 2 UTF-8 CRLF HTML

```html
        <p>DO</p>
        <input type="text" name="DO" required/>
      </div>
   <div class="item">
    <p>pH</p>
        <input type="text" name="PH" required/>
      </div>
      <div class="item">
        <p>Conductivity</p>
        <input type="text" name="conductivity" required/>
      </div>
   <div class="item">
    <p>BOD</p>
        <input type="text" name="BOD" required/>
      </div>
      <div class="item">
        <p>NI</p>
        <input type="text" name="NI" required/>
      </div>
   <div class="item">
        <p>Fec_col</p>
        <input type="text" name="Fec_col" required/>
      </div>
   <div class="item">
    <p>Tot_col</p>
        <input type="text" name="Tot_col" required/>
      </div>
      <div class="item">
```


File Edit Selection View Go Run Terminal Help
inputpage.html - Visual Studio Code
app_prac.py 1
WQI.ipynb
inputpage.html C:\...\templates
inputpage.html C:\...\template
style.css
C: > Project > Nalaiya Thiran > templates > inputpage.html > html > body > div > center > h3
⊗ 0 ⚠ 15
Ln 174, Col 33 Spaces: 2 UTF-8 CRLF HTML

```html
      <div class="item">
        <p>year</p>
        <input type="text" name="year" required/>
      </div>
        <div class="btn-block">
          <input type="submit" value="submit" />
        </div>

    </form>
    </div>
    <div>
      <center>
        <h3>{{output_text}}</h3>
      </center>
    </div>
  </body>
</html>
```

PREDICT PAGE (OUTPUT)



Water Quality Analysis

Temperature
35

DO
129

pH
4.9

Conductivity
0.2

BOD
2.0

NI
198.8

Fec_col
123.0

Tot_col
23.0

year
2011

submit

Quality of water is The water quality is Very Poor

GITHUB

https://github.com/IBM-EPBL/IBM-Project-36146-1660293206

PROJECT DEMO

DEMO VIDEO