School of Information Technology and Engineering (SITE)

B.Tech (Information Technology)

Project Report

Water Quality, AQI and Temperature Predictor ChatBot

Submitted for the Course

ITE 4003 Internet of Things

Offered during FALL 2020

(Dr. R.K. Nadesh)

By

Chinmay Nrusingh Choudhury	**18BIT0097**
Kuhoo Sharma	**18BIT0101**
Awani Kendurkar	**18BIT0082**

1. Introduction

1.1 Problem Statement

To create a chatbot that will provide weather, AQI and water quality prediction for a specific location.

1.2 Importance

Weather forecasting is important to protect life and property. Forecasting temperature and precipitation also helps in agriculture and traders to predict demands for commodities. AQI predictions help us to understand the effects of human actions on air quality. Air quality analysis also helps in providing a standard for air quality. Similarly water quality analysis also helps in providing insights on the water pollution and future impacts.

2. Overview and Planning

2.1 Proposed System Overview

We propose to create a ChatBot system with the following functionalities. It will provide three details, weather prediction, air quality index prediction and water quality prediction. The dataset for all this information will be taken from data.gov.in. The website provides real-time dataset through APIs and also dataset in .csv format. We use a cloud/local IoT suite for implementing ML algorithms to predict AQI, water quality and weather. This ChatBot is designed to provide people necessary information before they step out of their houses.

Assuming an IoT level 5 model, the end nodes are equipped with various sensors which send data to the cloud, which is stored in the database. For each group of nodes there is a coordinator node which collects data from the end nodes and sends it to the cloud/server.

For example, all AQI sensors have one main coordinator node that combines and collectively sends the data. Similarly for water quality and weather sensors.

2.2 Challenges

The major challenge is to find a valid dataset for the project. It is also required to clean the data to ensure consistency of data, to ensure analytics is not implemented on missing values, and removing unnecessary data. Also it is important to remove unnecessary columns from the dataset to reduce computation time and the data which is to be computed on.

2.3 Assumptions

We assume to have placed various groups of sensors in various places throughout the city for data collection, following the IoT level 5 module.

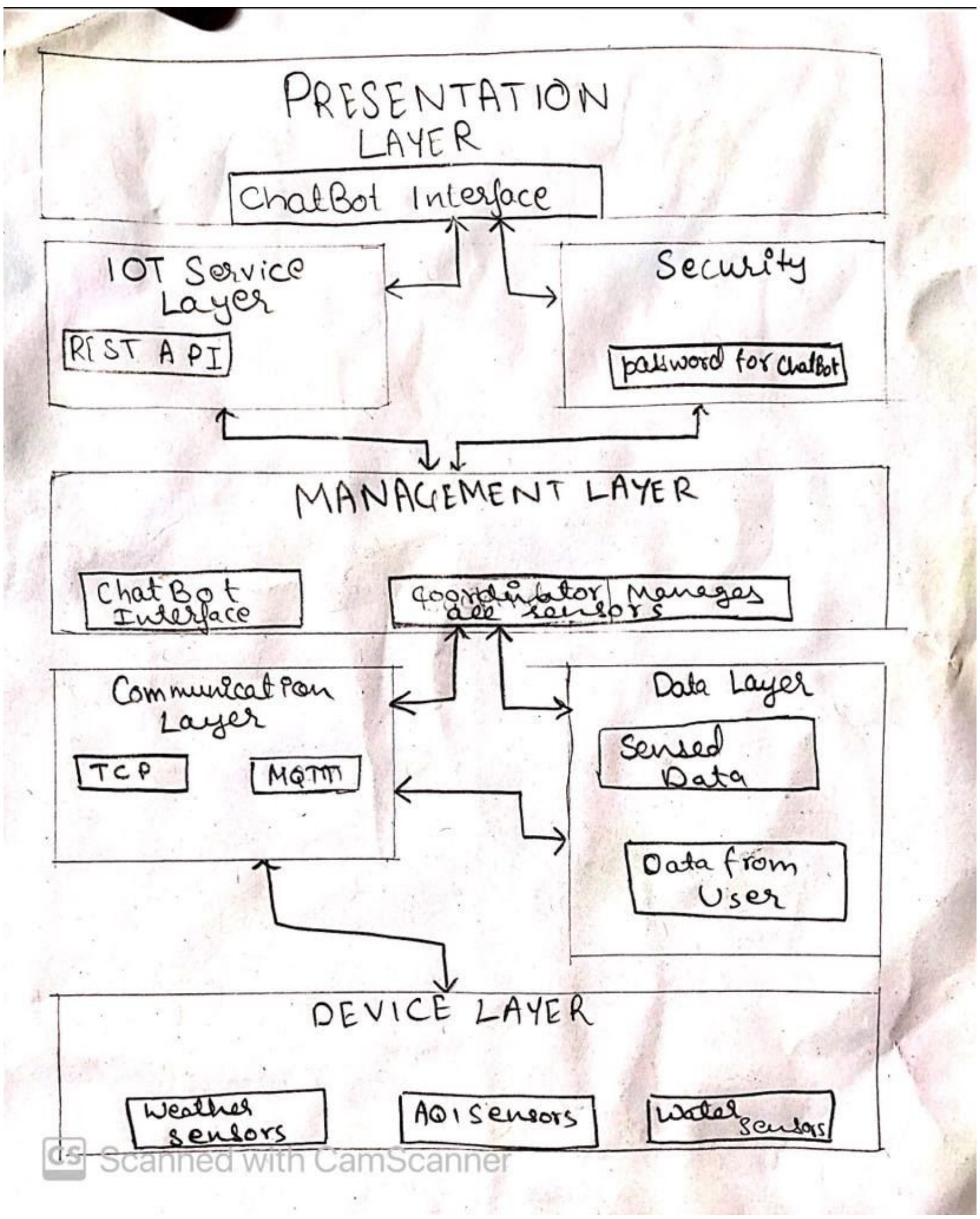
The data from all these sensors is assumed to have been collected and passed on to the cloud server by the coordinator sensor using the internet.

We start our analysis and prediction from this dataset. (However for the purpose of this project we use liable datasets available on the internet.)

2.4 Architecture Specification

Our system will be based on IoT level five. Our model will be processing on 3 types of datasets namely - temperature dataset, AQI dataset as well as water quality dataset. The processed output is stored for the user interface where the user inputs data and we give the predicted outputs to him.

A general architecture model is given :



2.5 Hardware Requirements

- A smart phone or a computer to use the ChatBot
- DHT-11 Digital Temperature And Humidity Sensor as a weather monitoring sensor
- Turbidity Sensor with Module for water quality prediction
- MQ135 - Air Quality Gas Sensor Module for air quality prediction

2.6 Software Requirements

- Jupyter Notebook
- R Studio
- Python 4

3. IoT Design Methodologies

Purpose & Requirements Specification

- ### Purpose

  A monitoring and analysis system which predicts weather, AQI and water quality and uses a ChatBot system as an interface.

- ### Behaviour

  The ChatBot will understand certain textual commands from the user to interact and provide relevant information. For example:

  “Pollution today ” ⇒ gives predicted ppm levels

  “Weather today” ⇒ gives weather prediction

  “Water quality” ⇒ gives water quality information

- ### System Management Requirements

  We collect and analyse data from the assumed physical sensors planted at various locations in an area. The user requires a smartphone or a computer system to access the ChatBot.

- ### Data Analysis Requirements

  The system will perform analysis and prediction of the data over the cloud.

- ### Application Deployment Requirement

  The application is deployed on a cloud server and accessible to the user through a ChatBot interface.

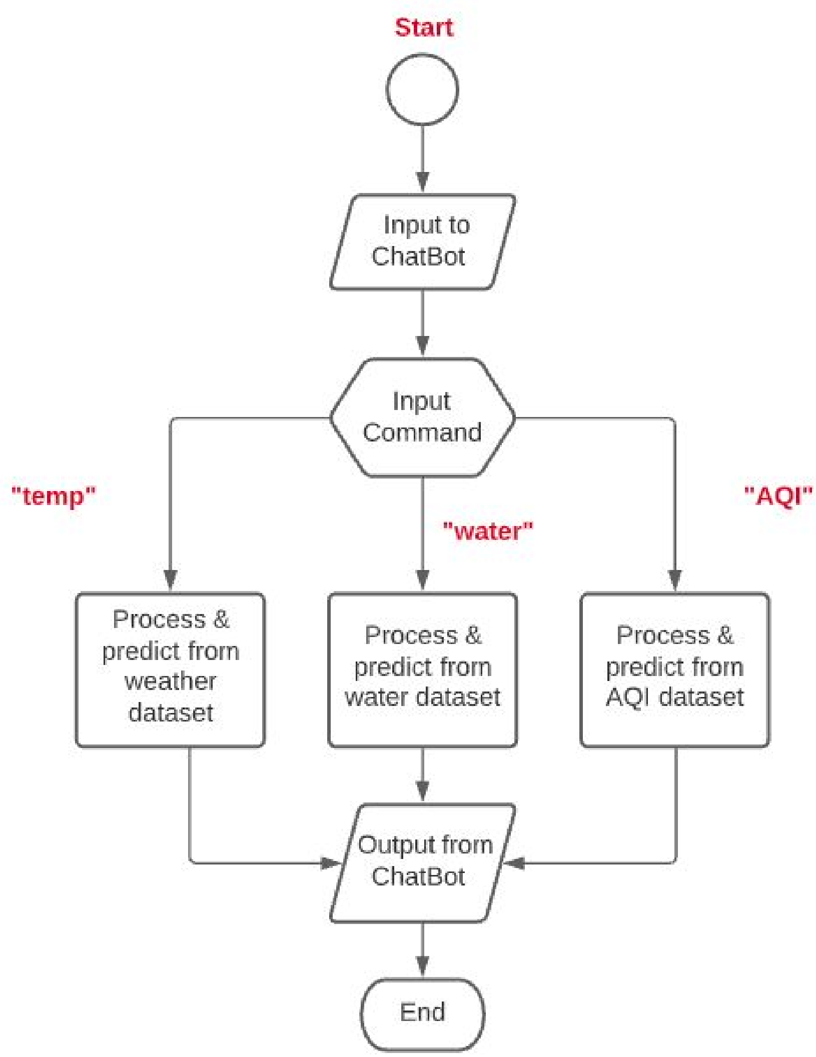
- ### Security Requirements

  The system shall try implementing security measures to prevent alteration/theft of data (hashing database keys). User authentication using login can also be performed.

Process Specification

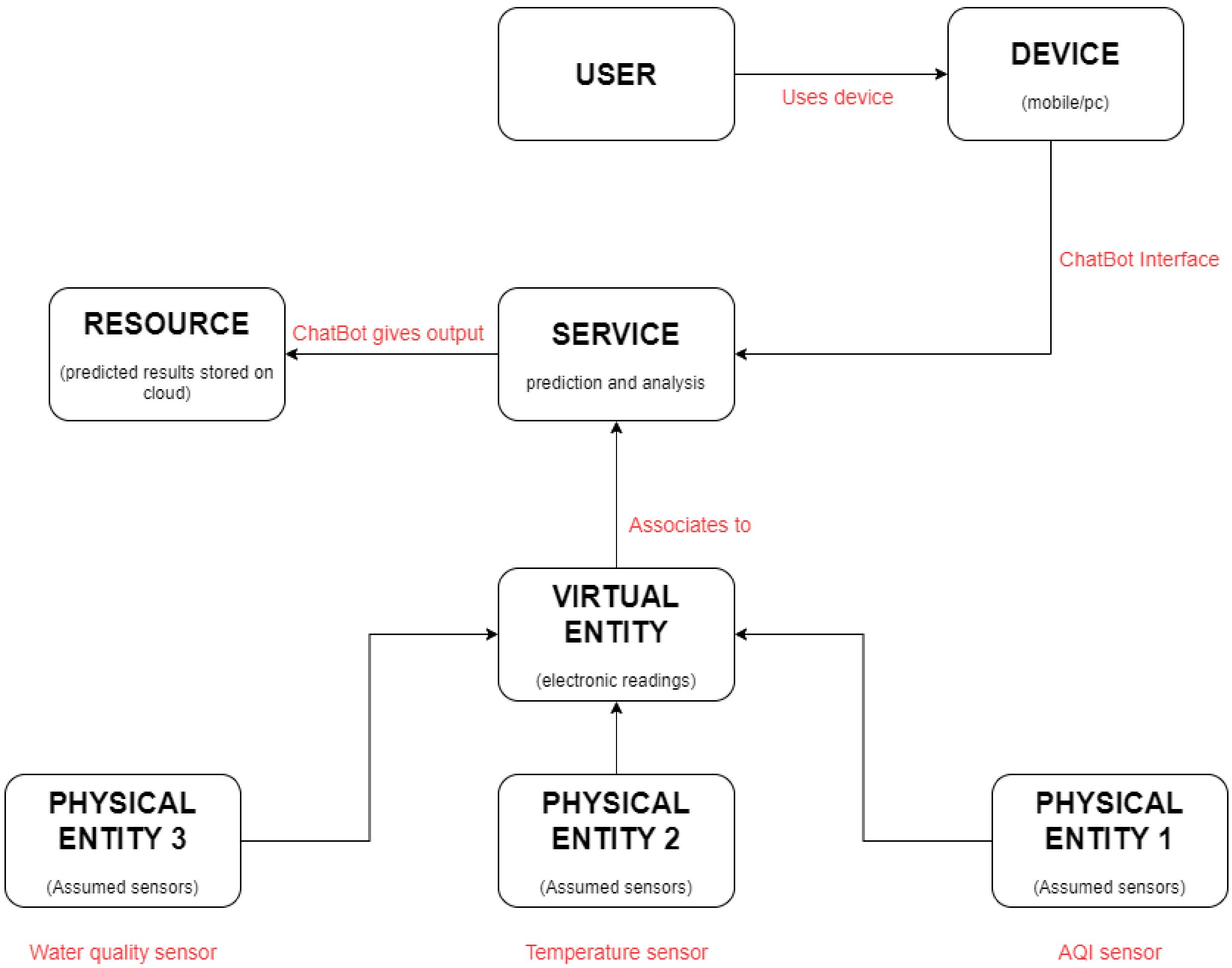
We have assumed to have the data set being periodically recorded (sensed using sensors) and stored on our database on the server. We make use of this database to analyse and therefore predict our results.

Process Specification Diagram:



Domain Model Specification

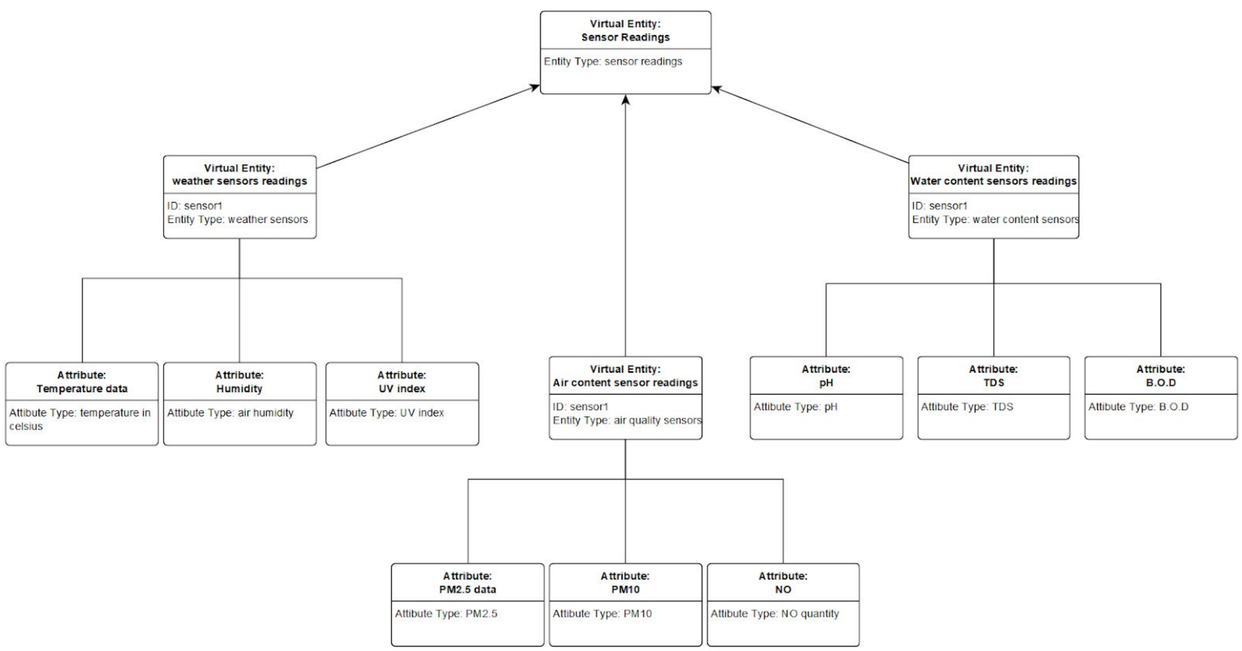
The domain model describes the resource, service, entities and devices. The domain includes physical entities or the assumed sensors installed at various specific locations, and the device or the interface through which a user can interact with our system is a ChatBot on a mobile phone or a computer.



Information Model Specification

This step in the IoT design methodology is to define the information model. It defines the structure of all the information in the IoT system, for example, attributes of virtual entities, relations, etc. It does not describe the specifics of how this information is represented or stored. To define the information model, we first list the virtual entities defined in the domain model.

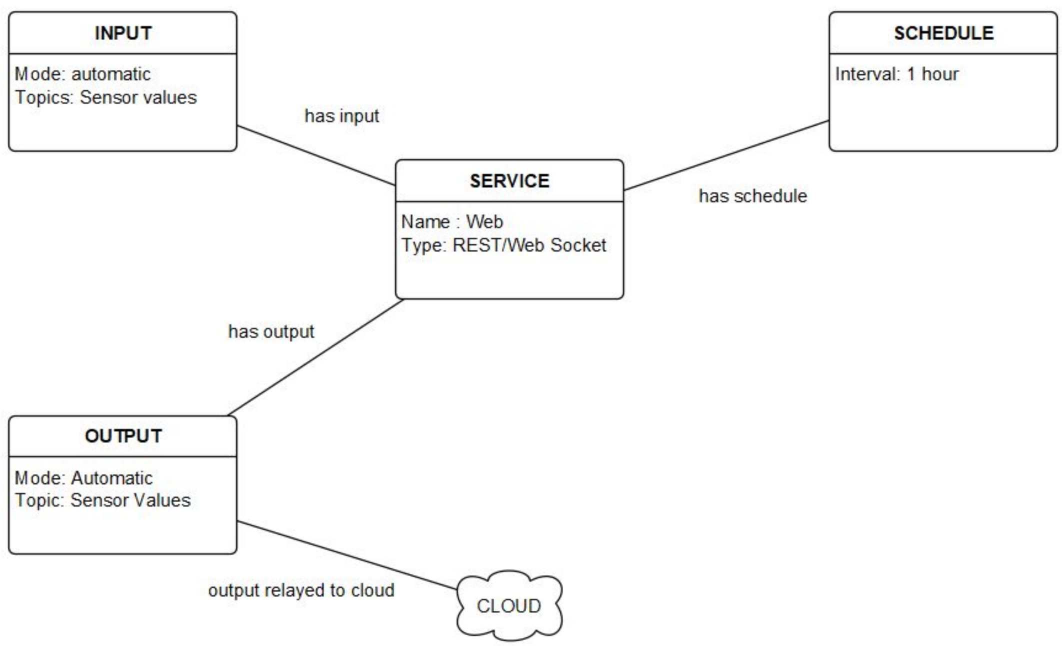
Information model adds more details to the virtual entities by defining their attributes and relations.



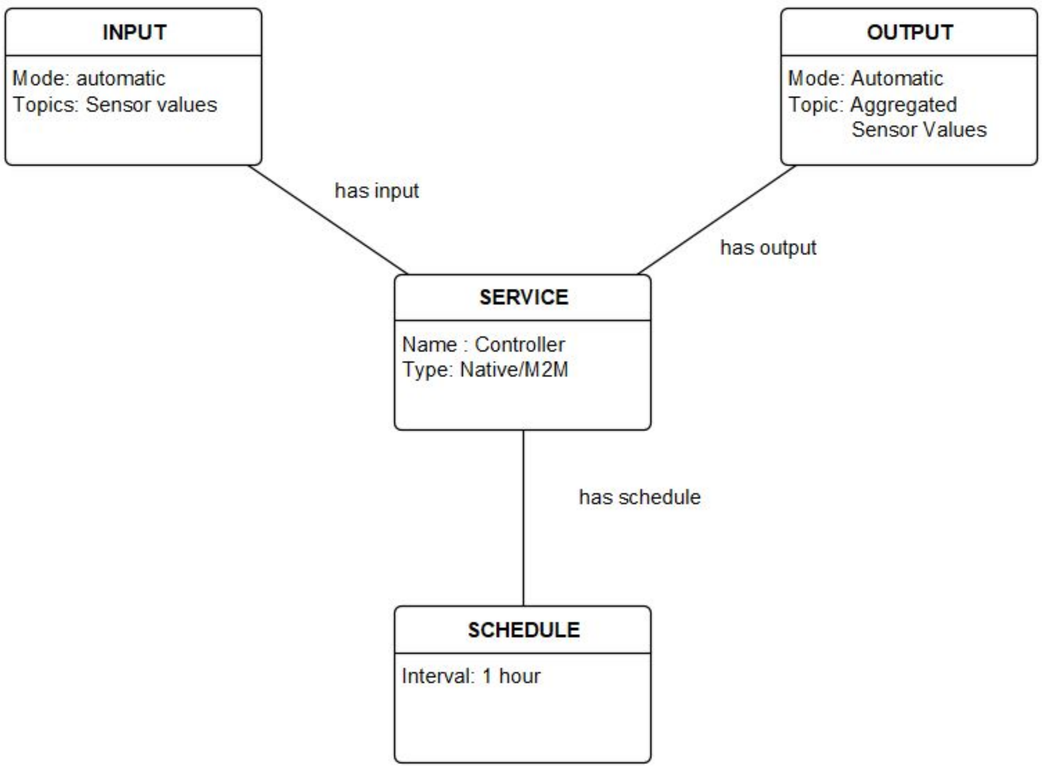
Service Specifications

The services specified will be used to collect data from the sensors and to process in the cloud. The purpose of the system is to collect environmental data such as temperature, humidity, pressure, air particle content and water particle contents.

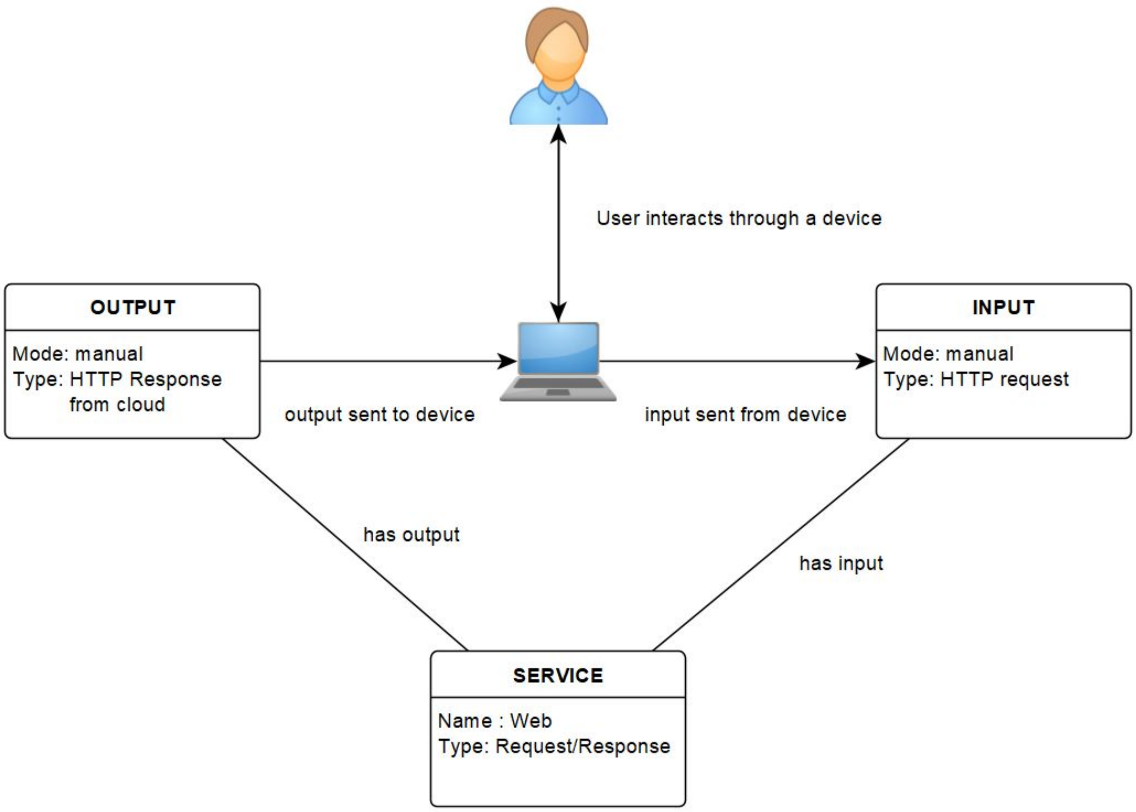
The following shows the service used for the communication between coordinator and the cloud. A REST/Web socket is used to pass the data to the cloud every 1 hour.



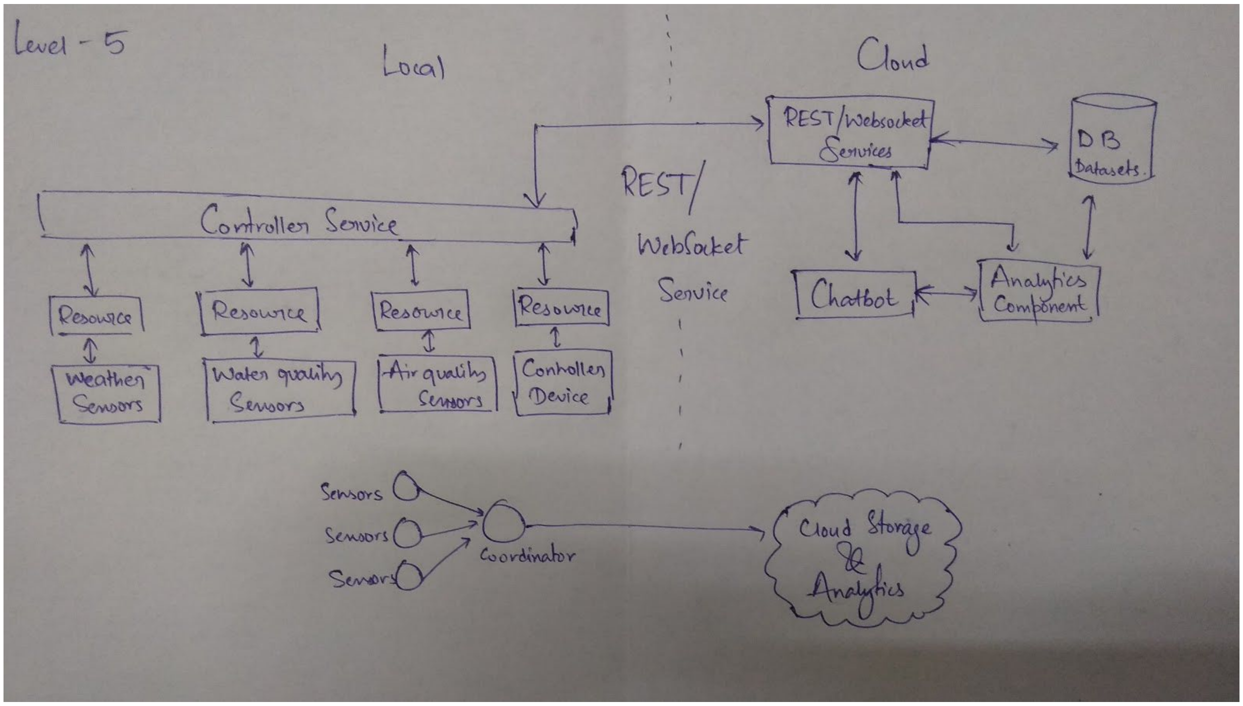
The following shows the service used by sensor devices and the coordinator for communication. The service input is the data generated from the sensors and output is aggregation of the data by coordinator. Further the coordinator also sends commands to the sensors via M2M (MQTT) connections.



Another service that is required is for establishing a communication channel between a user and the cloud. The user will send a request to the cloud server with the chatbot as interface and the cloud server will send a response back to the user using the same interface on the same device.



IoT Level Specification



The deployment level is level-5. There are multiple endpoints present that are monitored by a coordinator.

End points are all the environment sensors.

Coordinator is assumed to be a Raspberry Pi device.

The data from end points is sent to the coordinator via M2M connections. The coordinator further passes the data to the cloud via REST/web socket method.

Functional View Specification

The IoT Functional Model identifies groups of functionalities, out of which most are grounded in key concepts of the IoT Domain Model. The functional view defines the functions of the IoT systems, grouped into various functional groups. A number of these Functional Groups (FG) build on each other, following the relations identified in the IoT Domain Model. The Functionality Groups provide the functionalities for interacting with the instances of these concepts or managing the information related to the concepts, e.g. information about virtual entities or descriptions of IoT services. The functionalities of the FGs that manage information use the IoT Information Model as the basis for structuring their information.

<u>Devices</u>: DHT-11 Digital Temperature And Humidity Sensor for weather monitoring, Turbidity sensor with module for water quality prediction, MQ135 - Air Quality Gas Sensor Module for air quality prediction.

<u>Communication</u>: This block handles the communication for the IoT system. The protocols allow the devices to exchange data over the network.

802.11 wifi protocol - To determine how the data is physically sent over the network's physical layer.

IPv4 protocol - To transmit IP datagrams from the source network to the destination network.

TCP protocol - To determine communication over the transport layer.

MQTT protocol - A lightweight messaging protocol to publish data to the applications.

<u>Services</u>: This functional group includes the various services offered by the IoT system, such as weather monitoring plus water quality and AQI monitoring. This includes the web services as well.

<u>Application</u>: The application functional group provides the interface to the user to control and monitor the various aspects of this IoT system. The interface provided to our users for this system will be a ChatBot.

<u>Security</u>: In order to ensure basic security, user authentication will be performed when logging in.

Operational View Specification

Operational view specification is very important to address how the actual system can be realised by selecting technologies and making them communicate and operate in a comprehensive way.

Mapping for our IoT system:

Devices => A smartphone to access the interface, DHT-11 Digital Temperature And Humidity Sensor, Turbidity sensor with module, MQ135 - Air Quality Gas Sensor Module.

Communication => 802.11 wifi, IPv4, MQTT protocols

Services => Monitoring services, web services, prediction services

Management => Prediction in the form of interaction with the ChatBot

Application => ChatBot interface

Device & Component Integration

This step involves the integration and connection of the devices and the components. In our IoT system we assume 5-6 sensors placed in various parts of the city. These sensors are: DHT-11 Digital Temperature And Humidity Sensor, Turbidity sensor with module, MQ135 - Air Quality Gas Sensor Module.

Application Development

Development includes implementing the regression model on the dataset and producing output. The output would be accessed from a chatbot application with following commands:

“Temp” => access the temperature data.

“AQI” => access the air quality data.

“Water” => access the water quality data.

The chatbot is hosted on Telegram and the model is implemented on Google Colaboratory.

4. System Implementation

Cleaning of data :

The dataset used is from

https://www.kaggle.com/hiteshsoneji/historical-weather-data-for-indian-cities?select=bombay.csv

This is a huge and detailed dataset so we manually reduce the dataset for faster processing as well as remove the non useful attributes/columns which are not needed for implementation of the model.

The final cleaned dataset is has the following attributes and data for each hour from 1st Jan 2019 to 31st Jan 2020.

date_time	pressure	humidity	uvIndex	visibility	windspeedKmph	tempC
2019-01-01 00:00:00	1018	42	7	10	11	28
2019-01-01 01:00:00	1018	40	7	10	10	28
2019-01-01 02:00:00	1017	38	7	10	10	27
2019-01-01 03:00:00	1017	36	7	10	9	27
2019-01-01 04:00:00	1017	34	7	10	9	26
2019-01-01 05:00:00	1018	33	7	10	9	26
2019-01-01 06:00:00	1018	32	7	10	9	25
2019-01-01 07:00:00	1018	33	7	10	8	25
2019-01-01 08:00:00	1019	34	7	10	8	25
2019-01-01 09:00:00	1020	35	7	10	7	26
2019-01-01 10:00:00	1019	37	7	10	7	27
2019-01-01 11:00:00	1019	39	7	10	6	28
2019-01-01 12:00:00	1018	40	7	10	6	29
2019-01-01 13:00:00	1018	39	7	10	8	30
2019-01-01 14:00:00	1017	38	7	10	10	30
2019-01-01 15:00:00	1017	37	7	10	13	31
2019-01-01 16:00:00	1017	40	7	10	13	30
2019-01-01 17:00:00	1017	43	7	10	14	30
2019-01-01 18:00:00	1017	46	7	10	14	29

4.1 Module Development - Code

The following code is to read and import the datasets as numpy arrays and store them. They are then encoded and then split into training and test set and processed. The trained model is stored in trained_model.sav file.

In the following code we are importing necessary dependencies and packages

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import pickle
```

To upload the file from our local computer to Google Collab we use the following snippet.

```
from google.colab import files
uploaded = files.upload()
```

After this code runs a prompt comes up asking for file upload. Then we upload the respective dataset file depending on the value we are implementing for.

The dataset files are “tempdata.csv”, “aqidataset.csv” and “waterdata.csv” for temperature, AQI and water quality respectively.

Now we read the dataset as numpy arrays and store them in a variable

For predicting temperature we the dataset for temperature

```
dataset = pd.read_csv('tempdata.csv')
```

For predicting AQI we read the dataset for AQI.

```
dataset = pd.read_csv('aqidataset.csv')
```

For predicting water quality (BOD) we read the dataset for the same.

```
dataset = pd.read_csv('waterdata.csv')
```

X is all of the rows and columns. Y is the column corresponding to the target value. X1 is a copy of X which will be used ahead again.

```
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 6].values
X1=dataset.iloc[:,:-1].values
```

To train the model we first import related dependencies

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```

Now we separate our training set and testing set. Both of them will be taken out of the original dataset.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.99,
random_state = 0)
X_test1=X_test
```

We encode the values from the dataset to perform regression.

```
labelencoder_X = LabelEncoder()
X[:, 5] = labelencoder_X.fit_transform(X[:, 5])
onehotencoder = OneHotEncoder()
X = onehotencoder.fit_transform(X).toarray()
X = X[:, 1:]
```

Now we train the model based on regression

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.99,
random_state = 0)

from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

The regressor is saved as a .sav file

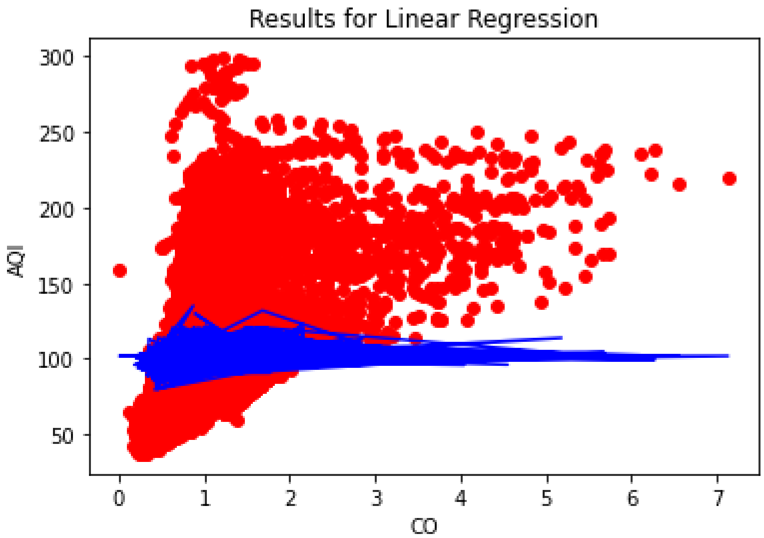
```
filename = 'new_model.sav'
pickle.dump(regressor , open(filename, 'wb'))
```

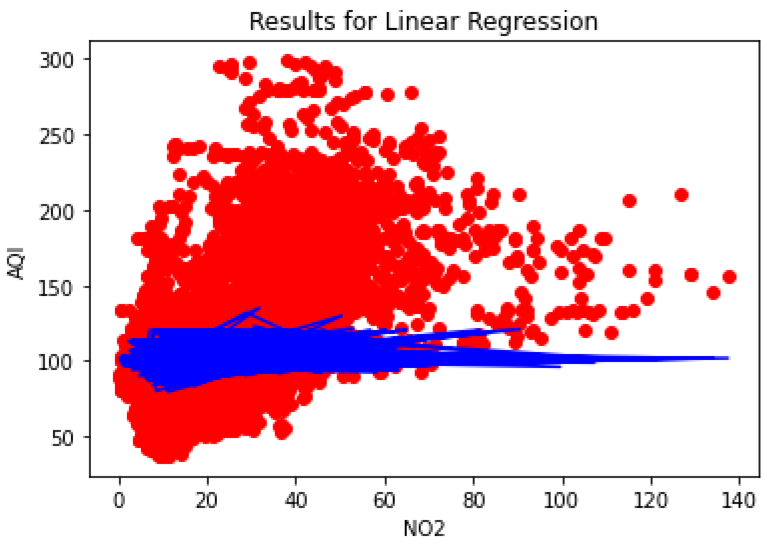
Now data will be predicted using the regressor and then we create a separated csv file named “result.csv” for predicted values.

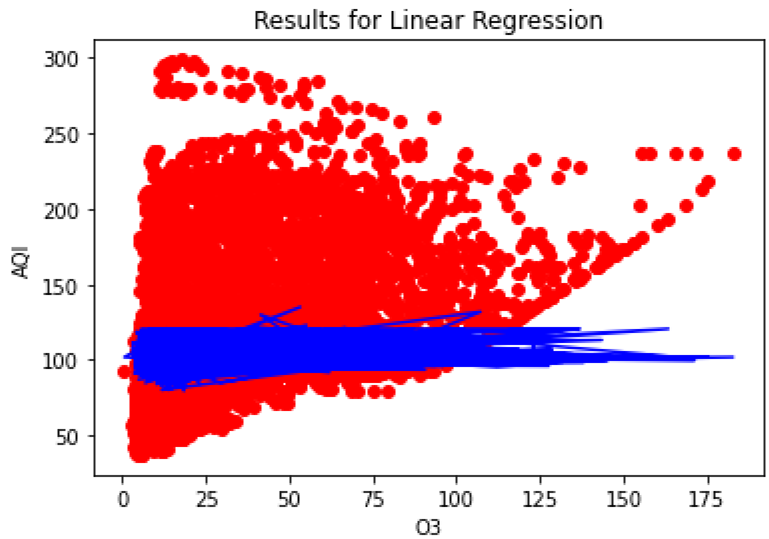
```
y_pred = regressor.predict(X_test)

for i in range(y_pred.size):
 data = pd.DataFrame([[X_test1[i][0],y_pred[i]]], columns=['Trained
model','temperature'])
 data.to_csv('result.csv',mode='a', header=False)
```

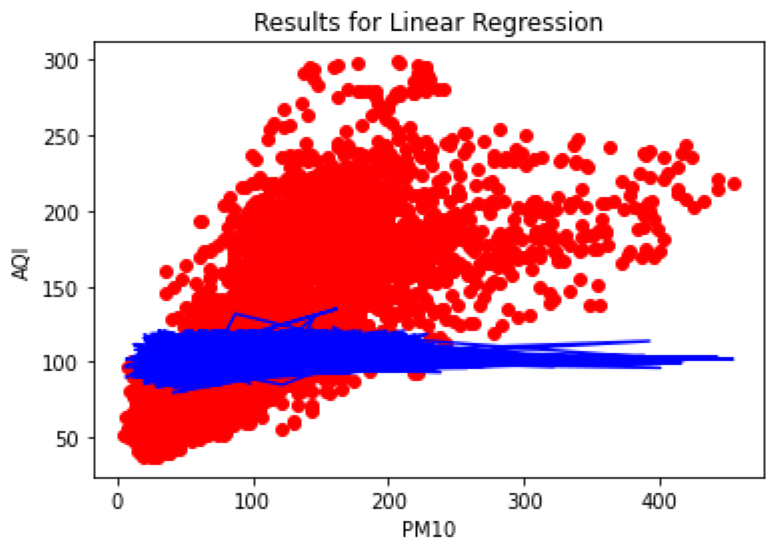
4.2 Output/Results

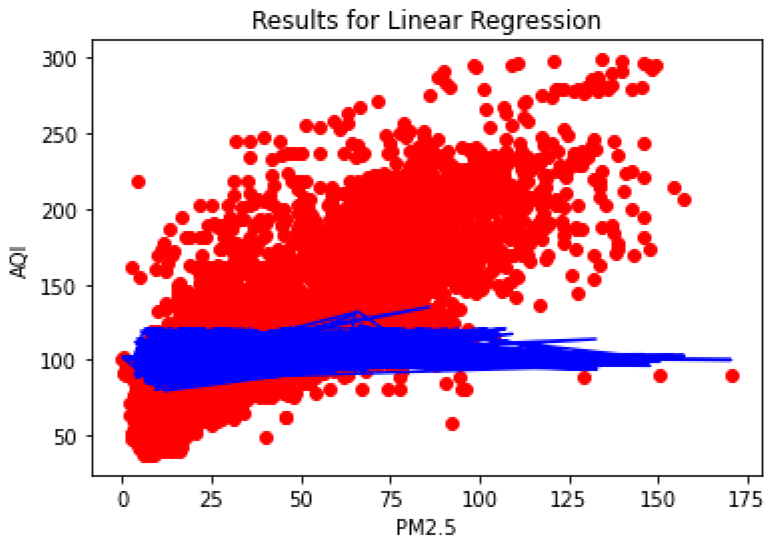
Air:

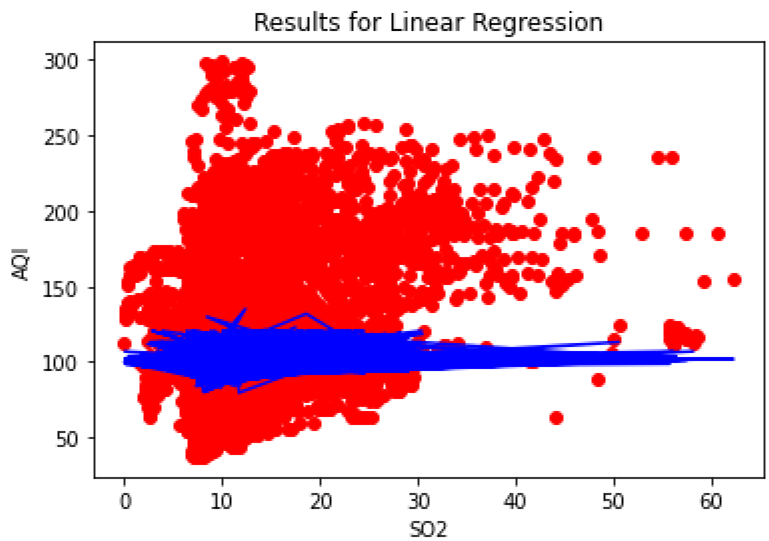




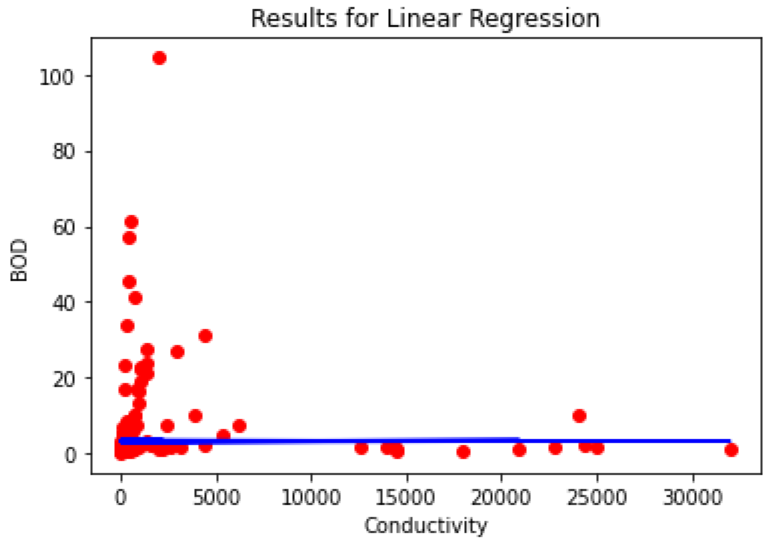

Results for Linear Regression
AQI
O3
300
250
200
150
100
50
0
25
50
75
100
125
150
175

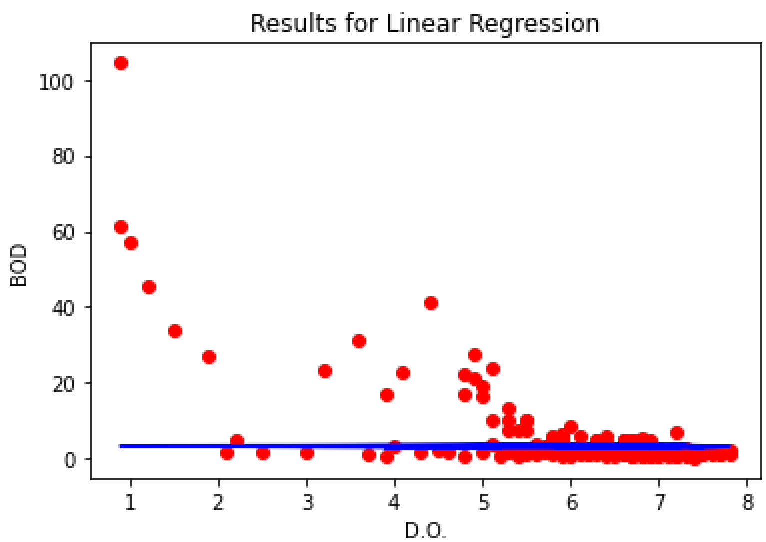

Results for Linear Regression
AQI
PM10
300
250
200
150
100
50
0
100
200
300
400

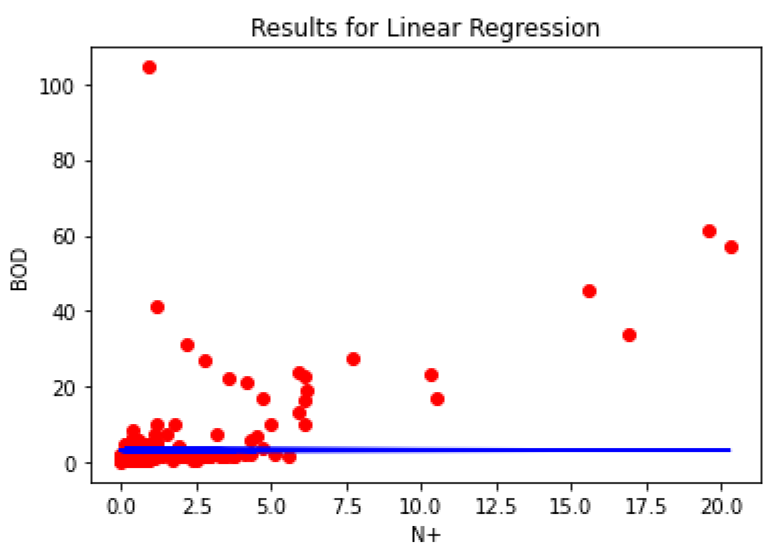

Results for Linear Regression
300
250
200
150
100
50
AQI
0
25
50
75
100
125
150
175
PM2.5

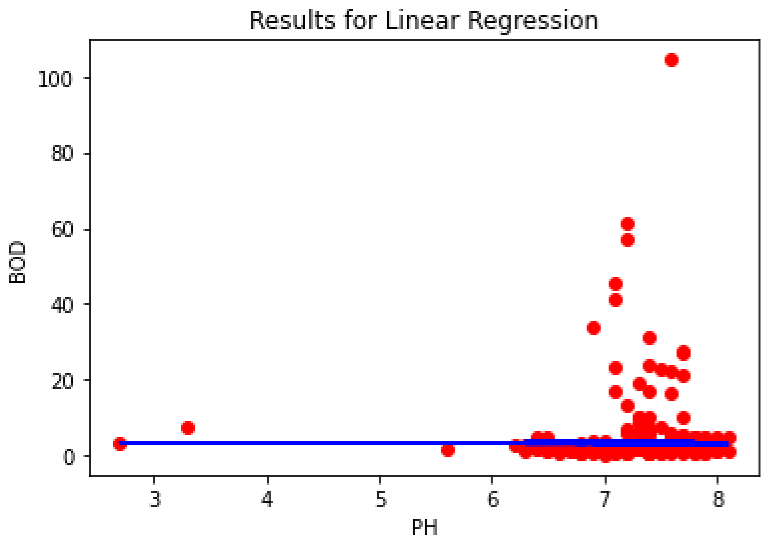

Results for Linear Regression
300
250
200
150
100
50
AQI
0
10
20
30
40
50
60
SO2

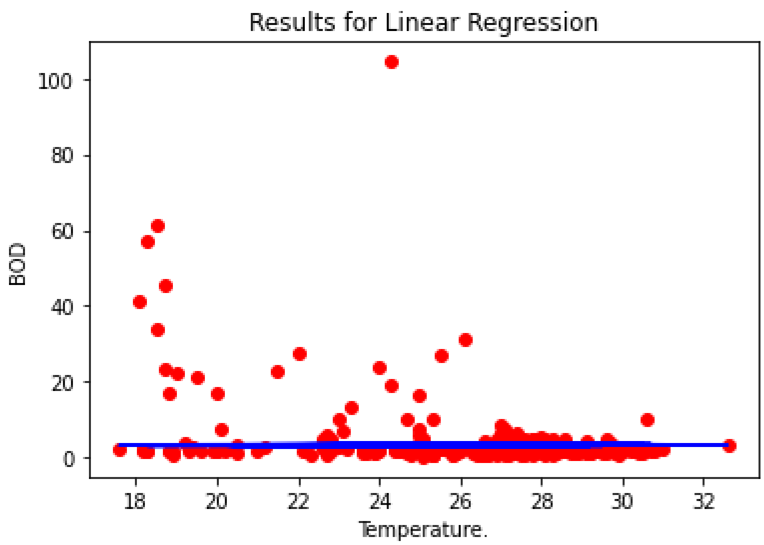
Water:


Results for Linear Regression
BOD
Conductivity
0
20
40
60
80
100
0
5000
10000
15000
20000
25000
30000

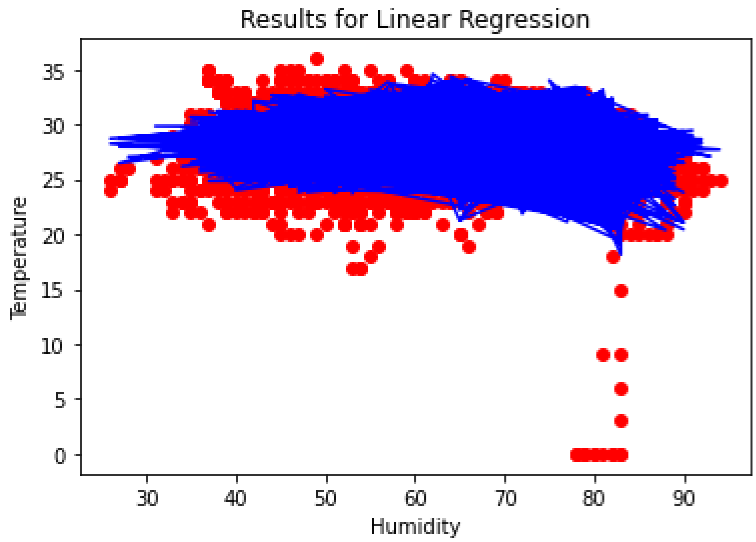

Results for Linear Regression
BOD
D.O.
0
20
40
60
80
100
1
2
3
4
5
6
7
8

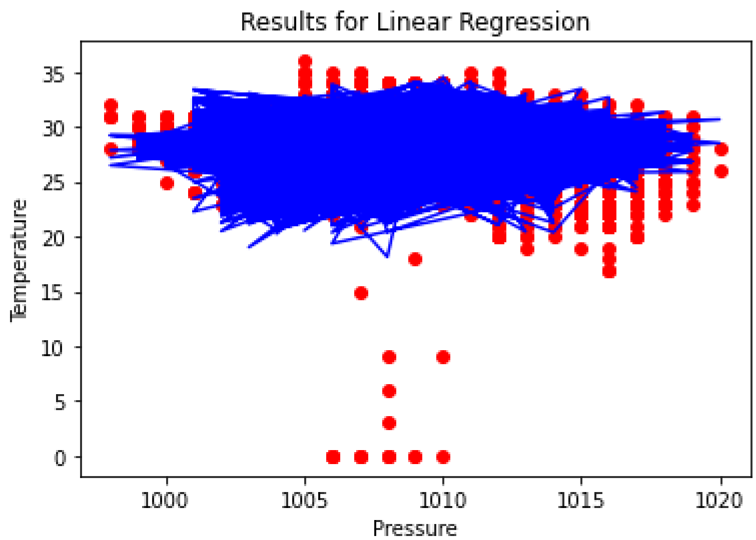

Results for Linear Regression
BOD
N+
0
20
40
60
80
100
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

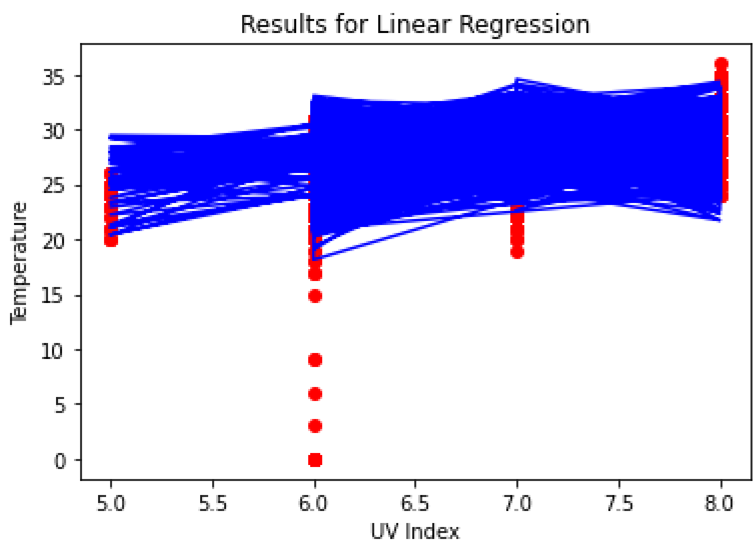

Results for Linear Regression
BOD
PH
0
20
40
60
80
100
3
4
5
6
7
8

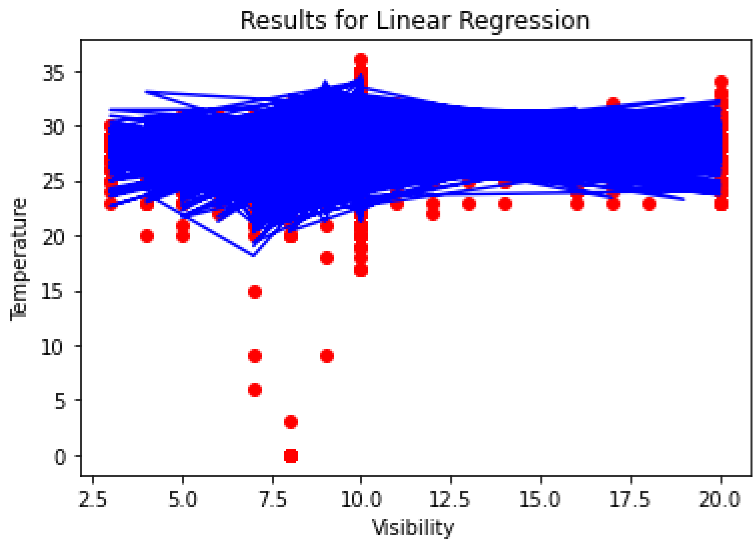

Results for Linear Regression
BOD
Temperature.
0
20
40
60
80
100
18
20
22
24
26
28
30
32

Temperature:

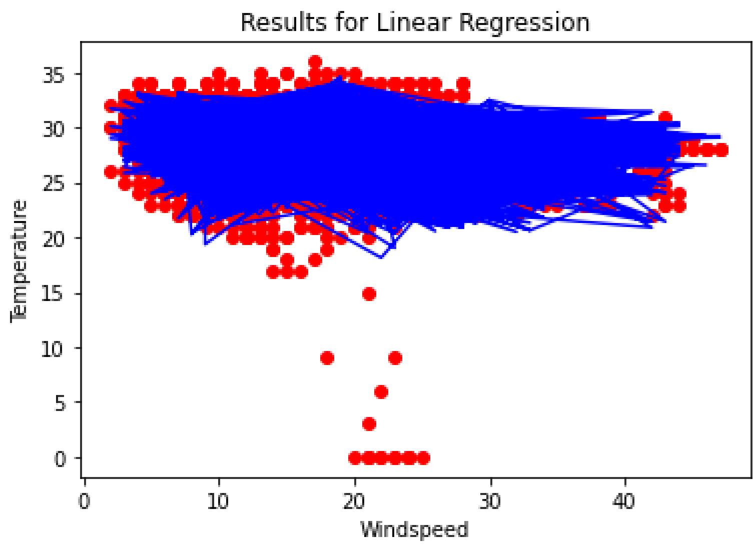






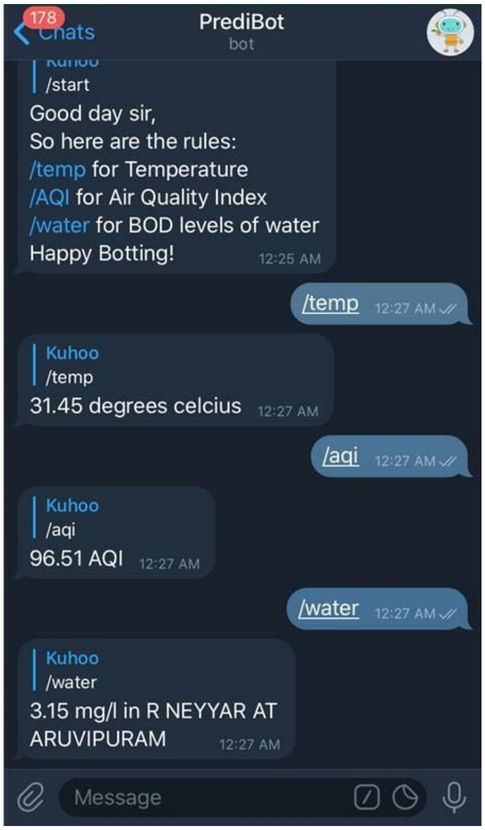


Results for Linear Regression
Temperature
Visibility



Results for Linear Regression
Temperature
Windspeed

ChatBot:



4.3 Discussion

Through the course of this project we came to understand the need for predictive analysis as well as the importance of IoT as an integral part of our lives. Predictive analytics plays a central role in deriving value from the real-time data of IoT devices. Many companies rely on this predictive analysis of their assets to get a sustainable market edge. Chatbots are of the more advantageous applications of IoT. While being complex and incorporating AI, they also give the ease of implementation and a great user experience.

5. Conclusion and Future Development

Weather forecasting is important to protect life and property. Forecasting temperature and precipitation also helps in agriculture and traders to predict demands for commodities. AQI predictions help us to understand the effects of human actions on air quality. Air quality analysis also helps in providing a standard for air quality. Similarly water quality analysis also helps in providing insights on the water pollution and future impacts.

The scope of this project can further be extended to including mobile app development, wherein the app is specifically designed to cater to the needs of the user in a more secluded fashion, as opposed to using a bot API for an already existing app.

Furthermore, we can perform location detection within the app so that the ChatBot gives relevant data persisting to nearby locations. For example, the /water command can be made to give the BOD (Biological Oxygen Demand) of the nearest water body.

For the purposes of this project demonstration, the datasets used were comparatively smaller and resulted in lesser accuracy. Moving the project to cloud (like Microsoft Azure suite) can provide us better accuracy on using a larger dataset.

6. References

1. Haupt, Sue Ellen, et al. "Machine learning for applied weather prediction." *2018 IEEE 14th International Conference on e-Science (e-Science)*. IEEE, 2018.
2. Chen, Sheng, et al. "Investigating China's Urban Air Quality Using Big Data, Information Theory, and Machine Learning." *Polish Journal of Environmental Studies* 27.2 (2018).
3. Kang, Gaganjot Kaur, et al. "Air quality prediction: Big data and machine learning approaches." *International Journal of Environmental Science and Development* 9.1 (2018): 8-16.
4. Mahalingam, Usha, et al. "A Machine Learning Model for Air Quality Prediction for Smart Cities." *2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*. IEEE, 2019.
5. Haghiabi, Amir Hamzeh, Ali Heidar Nasrolahi, and Abbas Parsaie. "Water quality prediction using machine learning methods." *Water Quality Research Journal* 53.1 (2018): 3-13.
6. https://medium.com/@mayankshah1607/machine-learning-feature-selection-with-backward-elimination-955894654026
7. https://www.kaggle.com/hiteshsoneji/historical-weather-data-for-indian-cities

<table>
<tr><td></td><td colspan="4">School of Information Technology & Engineering
B.Tech (Information Technology)
ITE 4003 Internet of Things –
FALL 2020
Course Project- Implementation Review(Final)
Evaluation Sheet
(Review Date 24.10.2020)</td></tr>
<tr><td>Title:</td><td colspan="4">Water Quality, AQI and Temperature Predictor ChatBot</td></tr>
<tr><td>Team Name</td><td colspan="4">ACK</td></tr>
<tr><td colspan="5">Project Team</td></tr>
<tr><td>S.No</td><td>Register Number</td><td>Student Name</td><td>Signature</td><td>Guided By</td></tr>
<tr><td>1</td><td>18BIT0097</td><td>CHINMAY NRUSINGH CHOUDHURY</td><td></td><td rowspan="3"></td></tr>
<tr><td>2</td><td>18BIT0101</td><td>KUHOO SHARMA</td><td></td></tr>
<tr><td>3</td><td>18BIT0082</td><td>AWANI KENDURKAR</td><td></td></tr>
<tr><td colspan="5">Team Member(s) Contribution and Performance Assessment</td></tr>
<tr><td colspan="2">Components</td><td>Student 1</td><td>Student 2</td><td>Student 3</td></tr>
<tr><td colspan="2">Implementation & Results -(30)</td><td></td><td></td><td></td></tr>
</table>

Contributed fair share to the team project -(05)			
Cohesive Presentation -(05)			
Documentation Hard/Soft - (05)			
Q & A - (05)			
Total - 50			

Student Feedback **(Student Experience in this Course Project)**	**Evaluator Comments**

Name & Signature of the Evaluator(s)