A Project Report on

Identifying the Sources of Water Pollution using Open IoT

Submitted in

partial fulfillment of the requirements

for the award of the

INTERNSHIP

in

INTERNET OF THINGS

by

ALBYN BABU

ARUNA RAVEENDRAN

AZEEM K

GOPAL M

LEKSHMI S

International Centre for Free and Open Source Software

Technopark | Trivandrum | Kerala

July 2018

Abstract

Now a days Internet of Things (IoT) is used in different areas of research for monitoring, collecting and analyzing data from remote locations. Due to the vast increase in global industrial output, the quality of water available to people has deteriorated greatly. Water is an essential need for human survival and therefore there must be mechanisms for testing the quality of water that made available for drinking in town and city articulated supplies and as well as the rivers, creeks and shoreline that surround our towns and cities. In this paper, we proposes an IoT system for identifying the sources of water pollution in canals. Our proposed design consist of pH, Turbidity, Dissolved Oxygen, Temperature and Humidity Sensors and that uses LoRa technology for transmission of data. The device is Deployed in different locations of canal Thettiyar near Tecnopark and take real time measurement and there by identifying the sources that cause water pollution in it.

Contents

List of Figures

Listings

Chapter 1

Introduction

Water bodies such as rivers, canals are being polluted by various industrial as well as domestic activities to an extent that aquatic beings can't even survive. Over the years, number of industries increased, that finding the source and extent of pollution became more difficult. Research says that the unchecked urban waste pollution has severely affected the natural ecology of major freshwater sources in the city. This happens due to the source of the pollution is unknown.

At the present, water quality in canal is measured by gathering a sample of canals water, which is done at a selected location on canals surface. The sample will be examined in laboratories to get the result of water quality, which is used as the parameter to decide if the water is polluted. This process, which is taken numerous times for effective measurement, takes relatively long time, also cost a considerable amount of money. To minimize sampling time and cost, the measurement process has to be done by implementing computer and information technology

So we introduce a way to identify the source of pollution by the use of IoT Technology. This can be done by measuring pH, Turbidity as well as the

oxygen level of water. This data can be processed and transmitted to cloud using LoRa Technology and visualized using D3.js
Water quality is identified by using different parameters such as pH, Dissolved Oxygen , Turbidity, Air Temperature and Air Humidity.

pH holds an important part of determining whether aquatic life can use it or not. They greatly influence the availability and solubility of all chemical forms in the lake and may aggravate nutrient problems, the high pH level may be affected from the high algae and water plant growth. The pH scales from 0 to 14, and the normal lake waters hovers between 6.5 to 8.5.

High levels of nutrients fuel algae blooms, which can initially boost dissolved oxygen levels. But more algae mean more plant respiration, drawing on DO, and when the algae die, bacterial decomposition spikes, using up most or all of the dissolved oxygen available. This creates an anoxic, or oxygen-depleted, environment where fish and other organisms cannot survive. Such nutrient levels can occur naturally, but are more often caused by pollution from fertilizer runoff or poorly treated waste water, the DO level of a shallow water in the lake supposed to be around 4-15 mg/L, the higher the better for freshwater fish to survive.

Turbidity is caused by particles suspended or dissolved in water that scatter light making the water appear cloudy or murky. Particulate matter can include sediment - especially clay and silt, fine organic and inorganic matter. High turbidity can significantly reduce the aesthetic quality of lakes and streams, having a harmful impact on recreation and tourism. It can increase the cost of water treatment for drinking and food processing. It can harm fish and other aquatic life .

Temperature is also important because of its influence in water chemistry. An important example of the effects of temperature on water chemistry is its

impact on oxygen. Warm water holds less oxygen than cool water, so it may be saturated with oxygen but still not contain enough for survival of aquatic life.

The proposed method help to analyze the source of pollution through an effective way.

Chapter 2

Literature Survey

In *" Smart Water quality Monitoring " A.N.Prasad, K. A. Mamun, F. R. Islam, H. Haqva* , [3] use GSM Technology for analyse the water quality using pH ,Conductivity, Temperature and Oxidation Reduction Potenial . For Sensing the data they use Libellium Smart Water Device.

"Romi Fadillah Rahmat, Athmanathan, Mohammad Fadly Syahputra, Maya Silvi Lydia " also uses GSM Tecnology for data transfer in *"Real Time Monitoring System for Water Pollution in Lake Toba"* and they use additional parameter Dissolved Oxygen.

Zigbee technology is used by *"Brinda Das and Maneesha V Ramesh"*in*"Real Time Water Quality Monitoring System using IoT " and "Water Quality Monitoring and Waste Management using IoT".*

Chapter 3

Proposed Method

Our proposed design mainly consist of 5 parts they are

- Hardware and Things Network
- Casing
- API Creation
- Data-Base Management
- Visualization

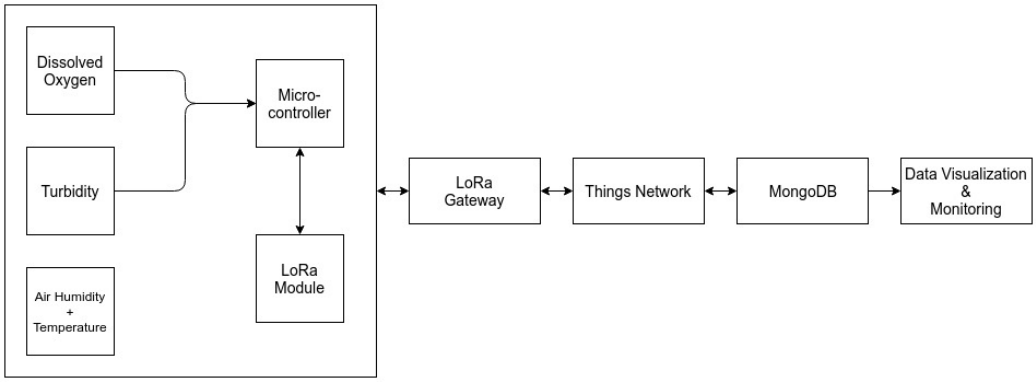

Figure 3.1: System Architecture

3.1 Hardware and Things Network

In this part of project we mainly dealt with designing and implementing dedicated 'Node' for water quality measurement and pushing the data obtained to Things network.

- Node for water quality measurement
- LoRa Gateway
- Things Network

3.1.1 Node For Water Quality Measurement

The Node basically consist of water quality measurement sensors such as Dissolved oxygen sensor, turbidity sensor, and a temperature/humidity sensor which are interfaced with an MCU with Embedded LoRa module. We used LoRa32u4 by BsFrance Inc as the MCU.

LoRa

LoRa is a patented digital wireless data communication IoT technology developed by Cycleo of Grenoble, France, and acquired by Semtech in 2012. LoRa uses license-free sub-gigahertz radio frequency bands like 169 MHz, 433 MHz, 868 MHz (Europe),915 MHz (North America) and 865-867MHz (India). LoRa enables very-long-range transmissions (more than 10 km in rural areas) with low power consumption.

LoRaWAN is the network on which LoRa operates, and can be utilized by IoT for remote and unconnected industries. LoRaWAN is a media access control (MAC) layer protocol for managing communication between LPWAN gateways and end-node devices, maintained by the LoRa Alliance.

LoRaWAN defines the communication protocol and system architecture for the network, while the LoRa physical layer enables the long-range communication link. LoRaWAN is also responsible for managing the communication frequencies, data rate, and power for all devices. As LoRa can be used to achieve large range communication as well as ultra low power consumption, it was the perfect technology for our project. The end device we wanted to install on various parts of thettiyar could use same gateway. Also it could be powered on same battery for years without manual recharging.the drawbacks i learned of LoRa WAN is that it can only be used for low data rate applications, also it requires a LOS for uplinking and downlinking.

Sensors

1. **Dissolved oxygen sensor(SEN0237-A)**

   This is a dissolved oxygen sensor kit by DFrobot, which is compatible with Arduino microcontrollers. This product is used to measure the dissolved oxygen in water, to reflect the water quality. It is widely applied in many water quality applications, such as aquaculture, environment monitoring, natural science and so on.

Figure 3.2: Dissolved oxygen sensor

Theory:

The dissolved oxygen sensors are mainly of three types:

(a) Optical Dissolved Oxygen Sensors

(b) Electrochemical Dissolved Oxygen Sensors

(c) Polarographic Dissolved Oxygen Sensors.

Considering feasibility, availability etc. we choose electrochemical sensors though it requires some maintenance over time. There are two types of electrochemical DO sensors: galvanic and polarographic.our sensor uses a galavanic probe.Both galvanic and polarographic DO sensors use two polarized electrodes, an anode and a cathode, in an electrolyte solution. The electrodes and electrolyte solution are isolated from the sample by a thin, semi-permeable membrane.When taking measurements, dissolved oxygen diffuses across the membrane at a rate proportional to the pressure of oxygen in the water . The dissolved oxygen is then reduced and consumed at the cathode. This reaction produces an electrical current that is directly related to the oxygen concentration . This current is carried by the ions in the electrolyte and runs from the cathode to the anode. As this current is proportional to the partial pressure of oxygen in the sample , it can be calculated by the figure 3.3.

Working:

For a new dissolved oxygen probe, 0.5 mol/L NaOH solution should be added into the membrane cap firstly as the **filling solution**.This solution had to prepared from chemistry lab of CET, by dissolving **2g NaOH crystals in 100ml distilled water.** The procedure to fill the solution is given in figure below. The output signal from the probe

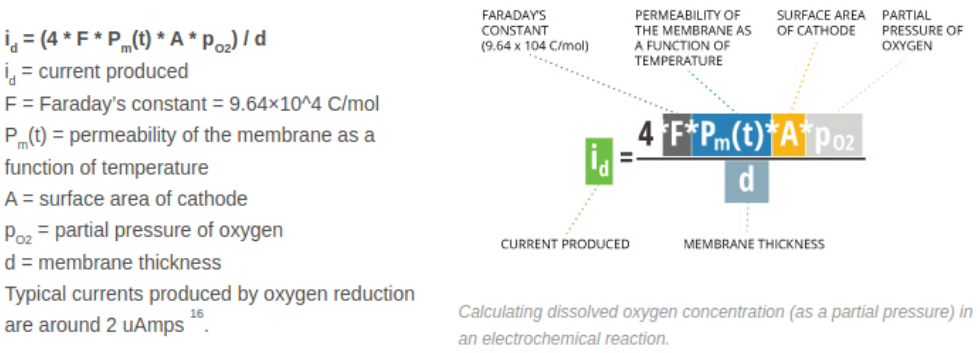

Figure 3.3: Dissoved oxygen current equation

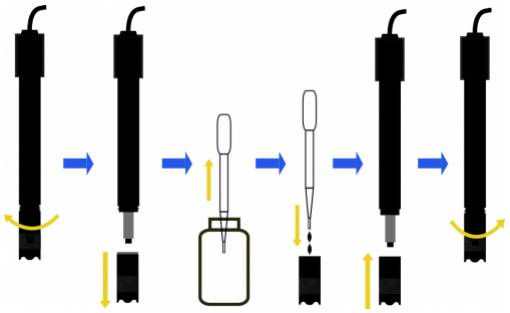

Figure 3.4: Filling the membrane cap

is then converted to voltage range of 0-5 v using a signal converter board.The output analogue signal from signal converter board is fed to the lora32u4. At first, we have to calibrate the sensor in fresh water at constant temperature.As permeability of the galvanic probe also depends upon temperature, the resulting voltage variation may not be that of DO variation. Thus **temperature compensation** must be reduced in the output. This can be done in arduino code.Temperature compensation can be calculated by using a table of saturated DO value at different temperature stored in the **EEPROM**. We used sample code provided by DFrobot.com for calibration and modified it for actual implementation(for memory and performance optimization). dissolved

oxygen value equation:

$$doValue = Voltage / SaturationDoVoltage * SaturationDoValue(with\ temperature\ compensation)$$

The sensor probe can be dipped in any flowing/still waater bodies and do value can be obtained.

2. **Turbidity sensor(SEN0189)**

   The gravity arduino turbidity sensor detects water quality by measuring the levels of turbidity. It uses light to detect suspended particles in water by measuring the light transmittance and scattering rate, which changes with the amount of total suspended solids (TSS) in water. As the TTS increases, the liquid turbidity level increases. Turbidity sensors are used to measure water quality in rivers and streams, wastewater and effluent measurements, control instrumentation for settling ponds, sediment transport research and laboratory measurements. This liquid sensor provides analog and digital signal output modes. we used analogue mode for our application.

   The output voltage signals had to converted to NTU(Nephelometric Turbidity Units) By using following graphical data.

   Firstly we have to convert 0-5v voltage output to 2.5-4.13 v range. secondly,we have to convert the voltage to NTU by implementing the above quadratic equation.

   Due to memory limitation of lora32u4 and for ease of transmitting the data all the above conversions are done in payload formatting of Things network console.

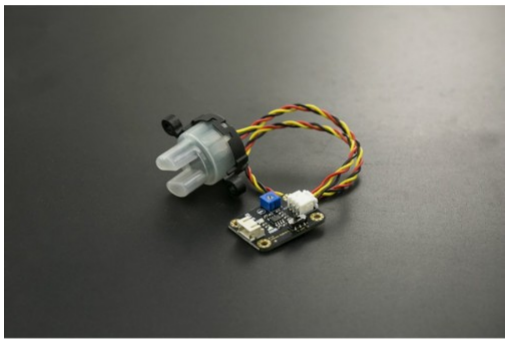

Figure 3.5: Turbidity Sensor

3. **Air temperature and humidity sensor(DHT11)**

   As DO sensor requires temperature value , we used a air temperature moodule to measure the temperature also normalize it to water temperature.(Water temperature sensor is in the proposed model, due to non availability dht11 considered.) Also temp/humidity value helps for proper maintenance of node.

   The DHT11 measures relative humidity. Relative humidity is the amount of water vapor in air vs. the saturation point of water vapor in air. At the saturation point, water vapor starts to condense and accumulate on surfaces forming dew. The saturation point changes with air temperature. Cold air can hold less water vapor before it becomes saturated, and hot air can hold more water vapor before it becomes saturated.

   The DHT11 detects water vapor by measuring the electrical resistance between two electrodes. The humidity sensing component is a moisture holding substrate with electrodes applied to the surface. When water vapor is absorbed by the substrate, ions are released by the substrate

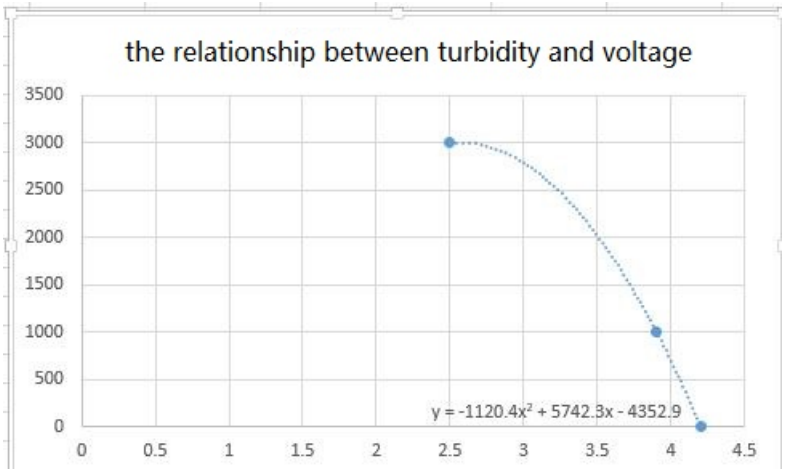

Figure 3.6: Relation between voltage and NTU

```
8  //turbidity
9  vol=  (bytes[6]<<16) | (bytes[5]<<8) |bytes[4];
10 vol1= vol*0.01;decoded.value = vol1;
11 x= (0.327309*vol1)+2.493455;decoded.val=x;
12 tur = (-1120.4*(x*x))+(5742.3*x)-(4352.9);
13 decoded.turbidity = tur ;
```

Figure 3.7: voltage-voltage-NTU conversion

which increases the conductivity between the electrodes. The change in resistance between the two electrodes is proportional to the relative humidity. Higher relative humidity decreases the resistance between the electrodes, while lower relative humidity increases the resistance between the electrodes.

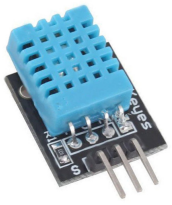

Figure 3.8: DHT11 sensor

The dht11 outputs digital value,hence a library named **dht.h** for mea-

surment.the signal pin of dht11 connected in a digital pin of lora32u4(pin:10) specify the pin, then temperature and humidity can be read by following lines of code in *void Loop()*:

```
1        int chk = DHT.read11(DHT11_PIN);
2        int temp = DHT.temperature;
3        int hum = DHT.humidity;
```

Listing 3.1: Reading Temperature and Humidity from DHT11

LoRa32u4

LoRa32u4 is a light and low consumption board based on the Atmega32u4, a 433MHz LoRA module RA02 from AI-Thinker and an LiPo/Li-ion USB battery charging circuit.The RA02 LoRa module is fitted with an U.FL (IPX) external antenna connector.

The ATmega32u4 is clocked at 8 MHz and 3.3 V. This chip has 32 K of flash, 2 K of RAM and built-in USB to Serial communication allowing debugging and programming capabilities without the need for an external FTDI chip, it can also act as an USB HID device (mouse, keyboard, USB MIDI device, etc).

This board is equipped with a LiPo and Li-ion charging circuit and a standard battery interface. It is fully compatible with Arduino. A white user led is tied to pin 13. An orange LED is showing battery charging status.

A. Specifications:

- ATmega32u4 @ 8MHz with 3.3V logic/power
- 3.3V regulator with 500mA peak current output
- USB native support, comes with USB bootloader and serial port debugging

- Hardware Serial, hardware I2C, hardware SPI support
- SX1278 RA02 LoRa based module with SPI interface
- 8 x PWM pins
- 10 x analog inputs
- +5 to +20 dBm up to 100 mW Power Output Capability (selectable in software)
- 300uA during full sleep
- 120mA peak during +20dBm transmit
- 40mA during active radio listening.
- U.FL (IP) antenna connector

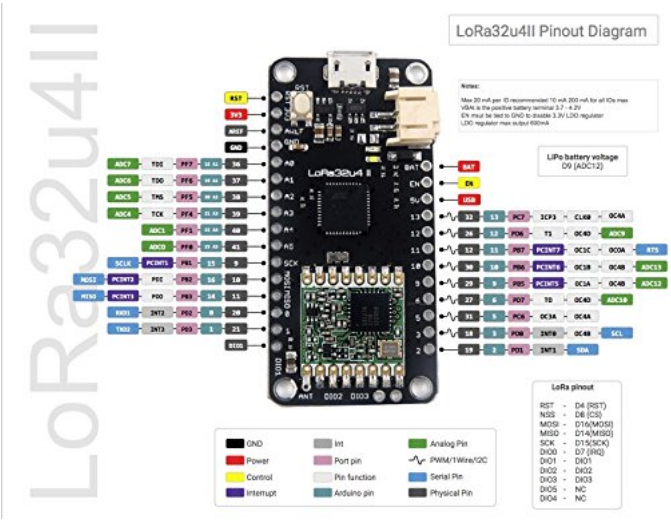

Figure 3.9: LoRa 32u4 II Pinout

Power Supply

Battery used in the project is a 7.4V,2600mAh Li-ion battery, Together with a buck converter for down converting the voltage to 5 v for powering MCU.

- Battery

Figure 3.10: 7.4V 2600mAh battery

- LM2596 DC-DC Buck convertor

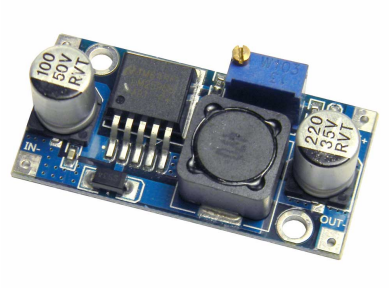

Figure 3.11: LM2596 DC-DC Buck convertor

3.1.2 LoRa Gateway

Gateways form the bridge between devices and The Things Network. Devices use low power networks like LoRaWAN to connect to the Gateway, while the Gateway uses high bandwidth networks like WiFi, Ethernet or Cellular to

connect to The Things Network.

All gateways within reach of a device will receive the devices messages and forward them to The Things Network. The network will reduplicate the messages and select the best gateway to forward any messages queued for downlink. A single gateway can serve thousands of devices. Gateways

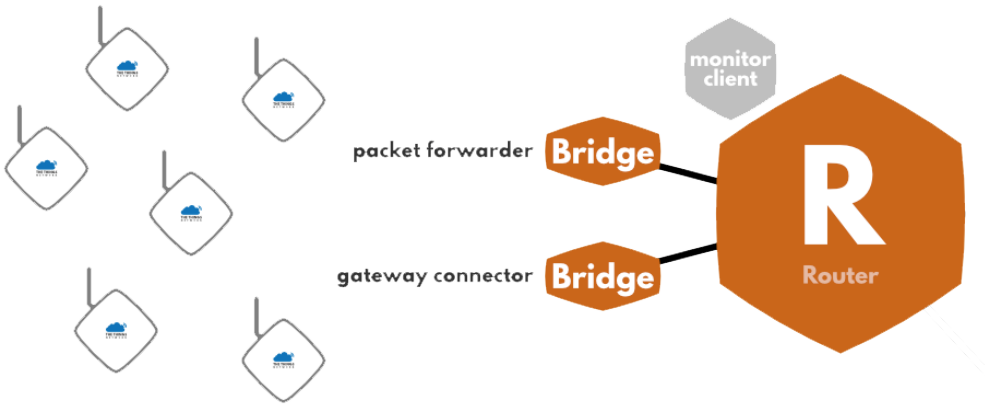

Figure 3.12: Gateway Model

are routers equipped with a LoRa concentrator, allowing them to receive LoRa packets. You can usually find two kinds of gateways:

- Gateways running on a minimal firmware, making them low-cost and easy to use (e.g. The Things Gateway), running only the packet forwarding software.
- Gateways running an operating system, for which the packet forwarding software is run as a background program (e.g. Kerlink IoT Station, Multitech Conduit). This gives more liberty to the gateway administrator to manage their gateway and to install their own software.

3.1.3 The Things Network

The Things Network is building a network for the Internet of Things by creating abundant data connectivity. It uses LoRa WAN technology for achieving this. In our project we used community version of Things network. It provides each user privilege to create their own personal applications as well as to add their own gateway to extend the LoRa network.in each application we can add number of LoRa devices. we can register each device either by ABP method or OTA method.

- **Over-the-Air Activation (OTAA)** is the most secure way to connect with The Things Network. Devices perform a join-procedure with the network, during which a dynamic DevAddr is assigned and security keys are negotiated with the device.

- **Activation by Penalization (ABP):**Using ABP join mode requires the user to define the following values and input them into both the device and Things network: Device Address, Application Session Key,Network Session Key.

The data from each device received as a series of bytes.These bytes can be encoded from the end node. which can be decoded and represented in suitable form by using payload formatting option.The bytes will be represented in hex code,so while decoding each byte has to be left shifted(8 bits) to correctly represent the data.

Things network provides various integrations such as Swagger(NoSql Storage), http integration(To provide a an external api access to the data from the devices.)In our project we designed and implemented an api on our sever and connected it as an integration to fetch the water quality data from the things network.

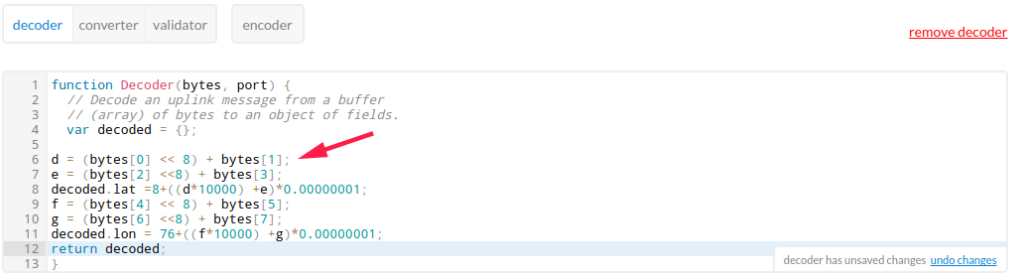

Figure 3.13: Example of payload formatting

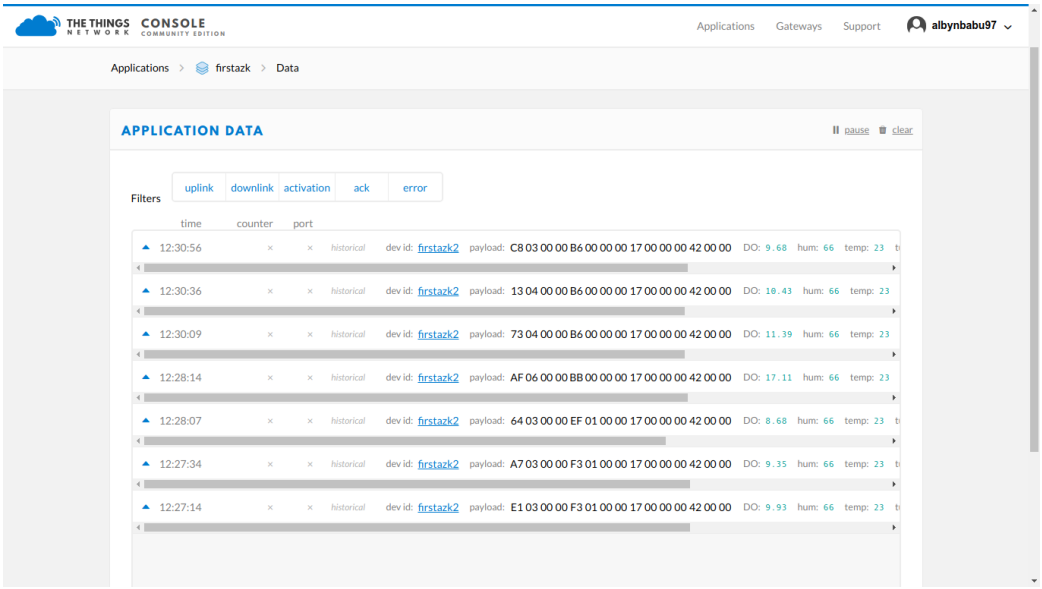

Figure 3.14: TTN Data

Things UNO

It is one of the hardware product of things Network.The Things Uno is based on the Arduino Leonardo (not the Arduino Uno) with an added Microchip LoRaWAN module. It is fully compatible with the Arduino IDE and existing shields.The Things Uno was designed with a low barrier for building proof of concepts for IoT projects using LoRaWAN in mind. Make an existing project wireless with up to 10km range by simply swapping the current Arduino board with the Uno. It is an Arduino Leonardo based embedded

Figure 3.15: Things UNO

SBC solution with an integrated Microchip LoRaWAN (Long Range WAN) RN2483 module and on-board antenna.

HTTP Integration

The data that are collected from the sensors are temporarily stored inside the cache of the TTN, but this temporary data will be vanished after we refresh the current session. So it is as important as getting data from sensors to store it somewhere. Here we use the HTTP Integration feature of TTN to post the data to the given API URI. The TTN will automatically post the updates to the API and in turn will be pushed to the database set inside the API code which will be discussed later.

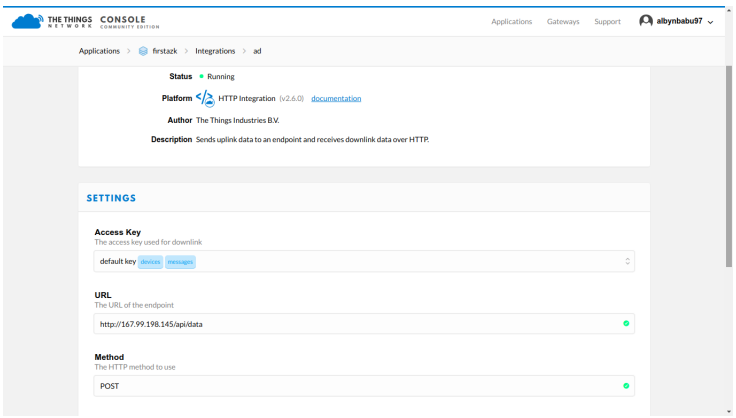

Figure 3.16: TTN HTTP Integration Page

3.2 LoRa32u4 Coding

Lora32u4 comes with a usb connector and programmer.it can be programmed with arduino IDE. LoRa32u4 uses ATmega32u4 board. in order to program it we have to add the **Adafruit feather32u4** board from the Adafruit avr board packages.

It uses a Rfm95 model lora module, so **Imic.h** can be used to program the up linking and down linking of data. we use ABP personalization method so we add dev address, network session key and app key to link the device to lora. We can define it as static global variables of the code.

imic.h works on a OS concept, It waits for a free channel,doing other operations in void loop at the time. It transmits the data when it gets a free channel.it also listens for a downlink.

All of our sensor data read to variables in a struct variable.Then the struct variable is converted to bytes and transmitted using *void dosend()* function.Inorder to avoid any floating/null data all the sensor readings are taken inside *void dosend()* function.

3.3 Casing

The sensors, controller and circuits of the node must be enclosed within a casing to protect from environmental factors such as rain, dust, heat etc. The casing also provides portability to the node which allows us to easily relocate the node from one point of interest to another. The casing for our project was fabricated from acrylic sheet plates that are fixed together. The plates were designed using the Free and Open Source CAD modeler FreeCAD and Inkscape, a Free and Open Source vector graphics editor. The design was cut out from acrylic sheet using a laser cutter. The fabrication work was

done at Fab Lab Kerala - Trivandrum.

3.3.1 Designing Using FreeCAD

The design of the casing was done using FreeCAD, a free and open source CAD modeler. Initially, a rough sketch was drawn to determine the positioning and arrangement of the components of the node, including the microcontroller, sensor modules, battery and charging circuit etc. Once this was determined, an estimate of the dimensions of the box was calculated. Next holes on the panels were drawn for the probes of the sensors, antenna for the LoRa module and the wires from the Solar panel. Using this information, a final rough sketch was drafted. Using this sketch, a design was created on FreeCAD. Each panel was created using the Part Design workbench. First, a sketch was created the panel. The sketch was modelled in the required shape with appropriate constraints for length. The edges of the panels were shaped in a ridge manner to provide an interlocking between the panels. Once the sketch was designed, the padding tool of the Part Design workbench was used to provide the thickness of the plate. Finally each plate was exported as a Flattened SVG (.svg) image.

3.3.2 Finishing Up with Inkscape

The Flattened SVG export of each plate was edited in Inkscape to provide some finishing touches. The FreeCAD export has an error were some lines are having multiple lines on top of each other which can cause problems during cutting. To resolve this, in Inkscape, each plate is selected and the ungroup tool is used to seperate the duplicate lines. The duplicate lines are deleted so that the resulting drawing doesn't have any duplicate lines. Next, the filling

is cleared and the lines are given the color which is required by the cutter to help it determine the lines which are to be cut, which in our case is red.

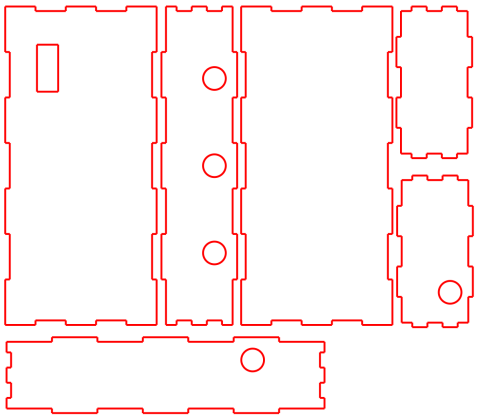

Figure 3.17: Design of the casing plates

3.3.3 Fabrication

The design for our case was cut out from an acrylic sheet using a laser cutter. This fabrication work was done at FabLab Kerala - Trivandrum. The material for our case is 3mm acrylic sheet. The machine used for cutting out the plates is the Speedy 100 by Trotec. First the SVG design was loaded into Inkscape and the line color was changed to red to identify that the design is to be cut, not engraved. Next, the stroke thickness was changes to 0.282 mm which is the diameter of the cutting laser.Finally, we printed the design. The plate is loaded onto the machine and the laser to calibrated to adjust its focus. After that, the print job is started and the cutting is done. The cut plates were then assembled and glued together to form the casing.

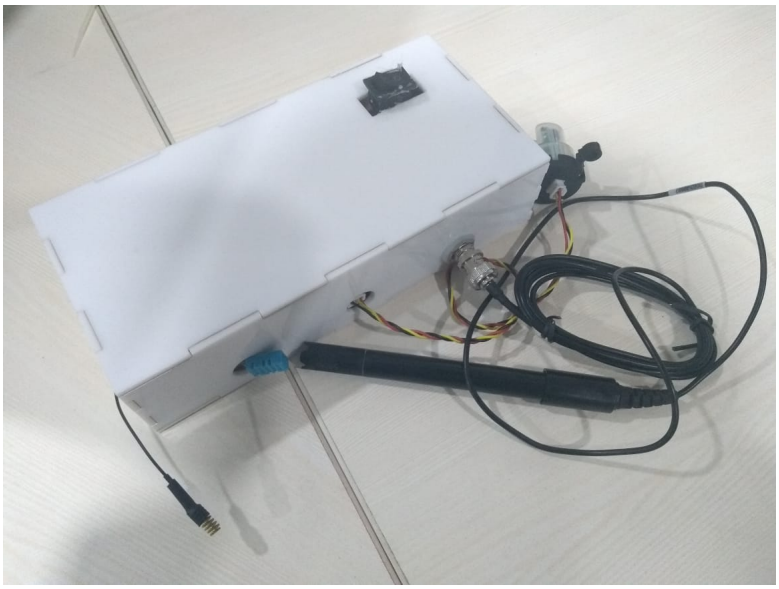

Figure 3.18: Final Case

3.4 Data-Base Management

Data from the TTN network need a permanent storage. We used Mongodb as the database that stores that data. MongoDB is a NoSQL database that stores in document format. Each database contains collections which in turn contains documents. Each document can be different with varying number of fields. The size and content of each document can be different from each other. We used MongoDB compass which is a GUI for MongoDB. Compass will Visually explore our data. In our project we created database called 'water_pollution' and collection name is 'data'. Data are stored in the JSON format.

Here we first refered online sites to learn about the new database MongoDB. After learning its commands and quires we installed the MOngoDB in a local system along with its dependencies and practiced in it. On becomming almost stable in managing the database we installed the database in a

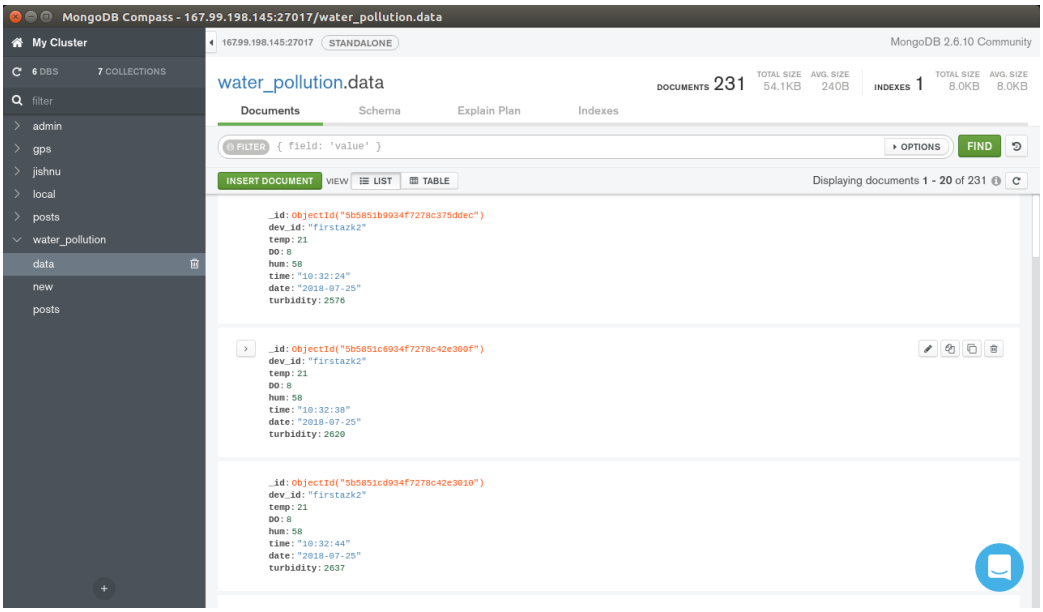

Figure 3.19: MongoDB Compass

server space so that we could access it from any remote point. Very basically we had two operations on MongoDB, namely pushing data into database from TTN and fetching or querying documents from the database for further prioccesing and visualization. We made it realized it by developing an API that pushes the data to and fetches data from database using Python that is discussed later.

```
#Configure DB settings
app.config['MONGO_DBNAME'] = "water_pollution"
app.config['MONGO_URI'] =
    "mongodb://167.99.198.145:27017/water_pollution"
```

Listing 3.2: Configuring DB in API

3.5 API Creation

API is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other. Python Flask framework and the flask-restful package have been used for the creation of the APIs. In our project the API has 2 parts

- POST Request
- GET Request

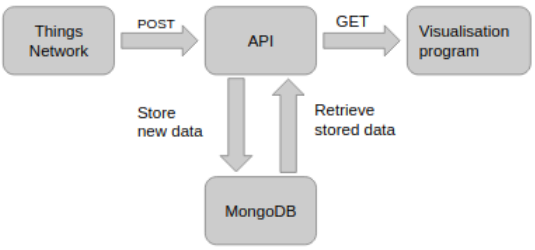

Figure 3.20: Flow Diagram

POST Request is used to store the data from the Things Network to the Database. In the POST method we assigned the values from TTN to specific variables and pushed it to the database as shown in the following code snippet.

```
def post(self):
        request = dataParser.parse_args()
        payload = payloadParser.parse_args(req=request)
        metaData = metadataParser.parse_args(req=request)
        dateTimeStr = metaData['time']
        dateTime = dateTimeStr.split("T")
        date = dateTime[0]
        time = dateTime[1].split(".")[0]
```

```
        new_data = {
            'dev_id' : request['dev_id'],
            'DO' : payload['DO'],
            'hum' : payload['hum'],
            'temp' : payload['temp'],
            'turbidity' : payload['turbidity'],
            'date' : date,
            'time' : time
        }
        data = mongo.db.data
        post_id = data.insert(new_data)
        post_inserted = data.find_one({'_id' : post_id})
        new_data = {
            'dev_id' : post_inserted['dev_id'],
            'DO' : post_inserted['DO'],
            'hum' : post_inserted['hum'],
            'temp' : post_inserted['temp'],
            'turbidity' : post_inserted['turbidity'],
            'date' : post_inserted['date'],
            'time' : post_inserted['time']
        }
        return { "success":True, "data":new_data }, 201
```

Listing 3.3: Python API for POST request

GET Request is used to retrieve the data from the Database and given to the visualization part. For that we took the required fields from the document inside the database using the following code snippet.

```
#Create Resource for data
class waterData(Resource):
    def get(self):
        data = mongo.db.data
        output = []
```

```
6         for q in data.find().sort("_id",-1).limit(15):
7       output.append({
8           'date' : q['date'],
9           'time' : q['time'],
10          'hum' : q['hum'],
11          'temp' : q['temp'],
12          'turbidity' : q['turbidity'],
13          'DO' : q['DO']
14      })
15        return output
```

Listing 3.4: Python API for GET request

3.6 Visualization

All the data will be useless when it cant represent in a proper way and that cannot understand by the user. Data visualization helps to represent our data to the right people, at the right time, enables to gain knowledge in an effective way. Currently visualization solutions have evolved as rapidly. Charts, videos, info graphics and at the cutting edge even virtual reality and augmented reality (VR & AR) presentations offer increasingly engaging and intuitive channels of communication[1]. There are so many tools such as chart.js, Tableau, Qlikview, FusionChart, Highcharts etc. For our project we use d3.js visualization tool.

3.6.1 D3.js

D3 stands for Data-Driven Documents. D3.js is a JavaScript library for manipulating documents based on data. D3.js is a dynamic, interactive, online data visualizations framework used in a large number of websites[2].D3.js is

an open source library allow to integrate our code to the sample code that already existing in the framework.Unique charts will help to develop chart using framework.It help to manipulate any part of DOM and provide flexibility offered by HTML, CSS and SVG. It can handle huge amounts of data.

In this phase of visualization of the data from the database using D3 we used the available script that resides inside the D3 library using Content Delivery Network(CDN). 'd3.json()' function was used to fetch the data as json from the api mentioned and the drawing function that describes how the data has to be represented before users like line or bar or bubble chart is called.

```
1 // Get the data from API
2 d3.json("http://167.99.198.145/api/data", function(error, data)
    {
3         if (error) throw error;
4         // Call function from plotting
5         draw(data);
6     });
```

Listing 3.5: JavaScript code for fetching JSON from API

3.6.2 Data representation using Chart

Here we are using bar chart and line chart for representation of data. Data is stored in the database(MongoDB). From MongoDB data is taken to the api using python flask. In the api data are in json format. D3.js call the json file in api for chart drawing. PH,humidity,dissolved oxygen,temperature are the data from the sensors. As aim of our project is to find the source of the water get polluted, we plot four chart respected to the time and date. Real time data are represented using line chart and historical data are represented

using bar chart.

3.6.3 Dashboard

The dashboard the appealing look for the presentation is designed using bootstrap available over the internet. The bootstrap template was downloaded and was carefully edited in order it could accommodate our JavaScript code that are used for graph creation. Required JavaScript codes and CSS files were attached along with using cdn itself. The scripts were embedded inside the div of the card that we see and an auto refresh of 10 seconds is set using meta in HTML head, which will refreshes the graphs and thus helps in plotting the real time data from the database which is automatically updated when the TTN sends data. The dashboard screenshots and the graphs are plotted herewith in following pages.

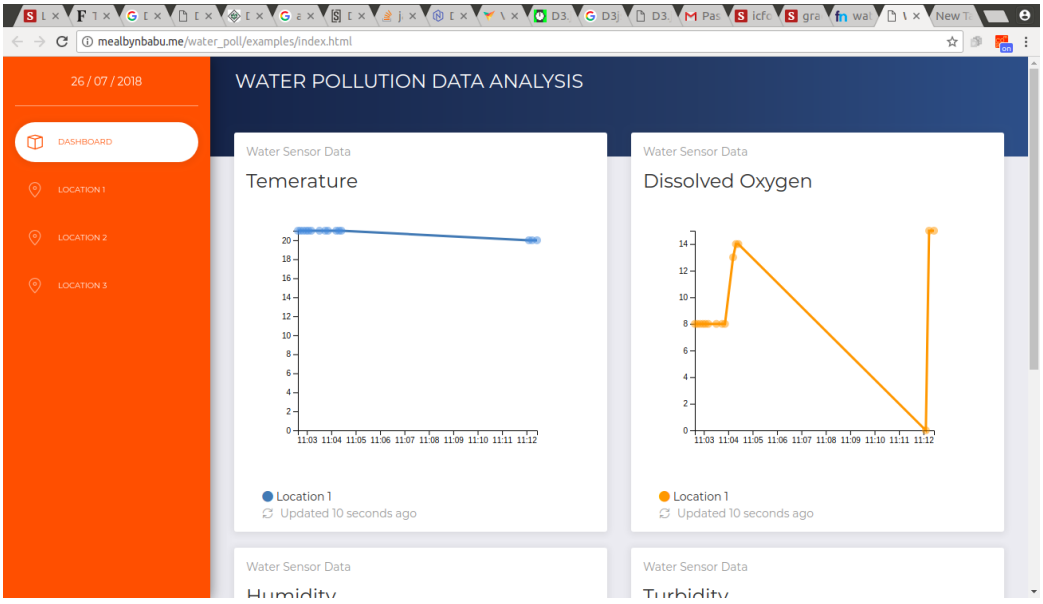

Figure 3.21: Realtime Data (Temp and DO)

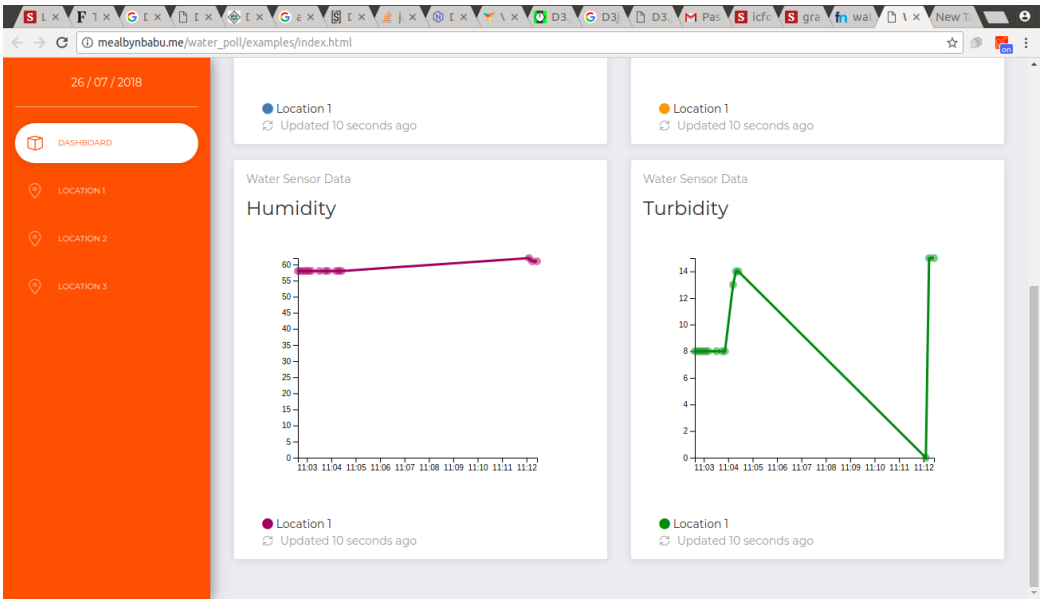

Figure 3.22: Realtime Data (Humidity and Turbidity)

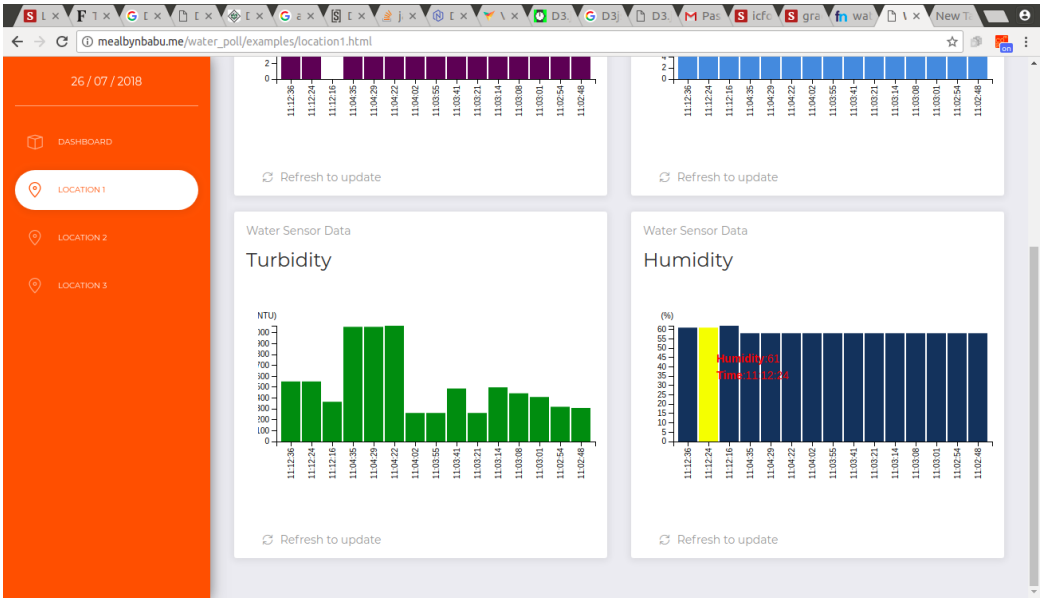

Figure 3.23: Historical data in Location 1

Chapter 4

Conclusion and Future Scope

The focus on our project is to measure the water quality parameters and identify the source for water pollution in canals. We implemented an IoT solution for this. Sensors continuously monitor water polluting factors and it is passed to The Things Network using LoRa, the comparison values can be displayed using D3.js.

The future of our project is vast as it is very economic and is relevant to the society. The casing designed is inefficient for releasing it to the open market and hence has to redesigned. As well as the wiring could be removed and implement it to a printed PCB board. More features like time range facility to view and analyze the graph could be incorporated. The dashboard could be redesigned to fit for responsive features.

We were able to complete the prototype of our project, the full product requires much more efficient power supply system, more senors, more compact casing etc. The product also requires much more research in making it more cost-effective,reliable and durable.

References

[1] "The 7 Best Data Visualization Tools Available Today",Accessed on July 2018[Online], Available:https://www.forbes.com/sites/bernardmarr/2017/07/20/the-7-best-data-visualization-tools-in-2017/5ea960a86c30/

[2] Tutorialspoint, Accessed on July 2018[Online]. Available:https://www.tutorialspoint.com/d3js/

[3] "Smart Water Quality Monitoring System" by A.N.Prasad, K. A. Mamun, F. R. Islam, H. Haqva

[4] "Real-Time Water Quality Monitoring System using Internet of Things Brinda Das, P.C. Jain"

[5] Real Time Monitoring System for Water Pollution in Lake Toba Romi Fadillah Rahmat, Athmanathan, Mohammad Fadly Syahputra, Maya Silvi Lydia Real Time Monitoring System for Water Pollution in Lake Toba Romi Fadillah Rahmat, Athmanathan, Mohammad Fadly Syahputra, Maya Silvi Lydia

[6] Water Quality Monitoring and Waste Management using IoT Maneesha V Ramesh, Nibi K V

[7] Mongodb Reference : https://www.tutorialspoint.com/mongodb/

[8] API Creation Reference : https://www.youtube.com/watch?v=upGiAG7-Sa4