

IoT DESIGN AND INTEGRATED APP DEVELOPMENT FOR WATER QUALITY MONITORING

Final Year Project Report

*submitted in partial fulfillment of the
requirements for the award of the degree
of
Bachelor of Technology
in
COMPUTER SCIENCE & ENGINEERING
BY*

Antothijah Bhoi (B19CS004)

Banteilang Mukhim (B19CS010)



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY MEGHALAYA, INDIA**

May 2023

CERTIFICATE



We hereby certify that the work which is being presented in the B.Tech. Final Year Project Report titled **“IoT Design And Integrated App Development For Water Quality Monitoring”**, in partial fulfillment of the requirements for the award of the Bachelor of technology in Computer Science & Engineering and submitted to the Department of Computer Science & Engineering of National Institute of Technology Meghalaya, India is an authentic record of our own work carried out during a period from 1st August 2022 till 17th May 2023 under the supervision of **Dr. Diptendu Sinha Roy, Associate Professor**.

The matter presented in this report has not been submitted by us for the award of any other degrees elsewhere.

(Signature of Candidate)

Student Name: **Banteilang Mukhim**

Roll no: **B19CS010**

(Signature of Candidate)

Student Name: **Antothijah Bhoi**

Roll no: **B19CS004**

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

(Signature of Supervisor)

Dr.Diptendu Sinha Roy,
Associate Professor
NIT Meghalaya

Head

Department of Computer Science & Engineering, National Institute of
Technology Meghalaya, India

DECLARATION OF ORIGINALITY

We hereby declare that this project work titled **“IoT Design And Integrated App Development For Water Quality Monitoring”** represents our original work carried out as students of the Department of Computer Science & Engineering of National Institute of Technology Meghalaya, India and to the best of our knowledge it contains no material previously published or written by another person unless cited. Any contribution made to this project work by others, with whom we have worked at National Institute of Technology Meghalaya or elsewhere, is explicitly acknowledged.



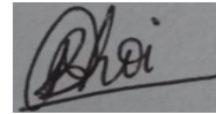
(Signature of Candidate)

Student Name: Banteilang Mukhim

Roll No: B19CS010

Place: Shillong, Meghalaya

Date: 31-05-2023



(Signature of Candidate)

Student Name: Antothijah Bhoi

Roll No: B19CS004

Place: Shillong, Meghalaya

Date: 31-05-2023

ACKNOWLEDGEMENT

First and foremost, we would like to thank **God** for giving us the strength, health and wisdom to perform this project work throughout the whole 2 semesters.

We are immensely grateful to **Dr. Diptendu Sinha Roy**, our project supervisor, for his invaluable guidance and mentorship throughout this endeavor. His insightful feedback, constructive criticism, and continuous encouragement have been pivotal in shaping our work and enhancing the quality of this report.

We extend our sincere thanks to all the other **faculty members** of the **Computer Science Department** who have imparted their knowledge and expertise to us during our academic journey. Their stimulating lectures, thought-provoking discussions, and profound insights have broadened our understanding of the subject matter and enriched our learning experience.

We would also like to thank our **friends** and our **family** who have contributed and helped us in completing this project.

ABSTRACT

This report describes the design and implementation of an IoT device & android application for monitoring water quality, aimed at ensuring safe and healthy consumption of water. The objective of the project is to develop a system that can monitor water quality using sensors and provide real-time information on water parameters through an Android app. The methodology involves using turbidity, pH, and TDS sensors to collect physical and chemical data, which are forwarded to a microcontroller and transmitted & stored in a Google Firebase database.

The Android app allows users, both clients and authorized personnel, to access the water quality information and view the status of various rivers by fetching sensor data from Google Firebase database. Machine learning algorithm (Random Forest) is integrated into the app to classify the collected data as potable or non-potable.

The proposed system enables authorities to monitor and inform people about the quality of water, ensuring safe drinking water and improving living standards. By leveraging IoT technology and data analysis, this project provides an efficient solution for water quality monitoring.

TABLE OF CONTENTS

	Page No.
1. Introduction.....	1-2
1.1. Objective.....	1
1.2. Methodology.....	1
1.3. Proposed System.....	1
2. Different Water Quality Parameter.....	3
3. Water Standards.....	4-6
4. Overall IoT System Design.....	7-8
4.1. IoT System Device.....	7
4.2. Flowchart.....	8
5. Hardware Design & Development.....	9-19
5.1. Hardware Circuit Diagram.....	9
5.2. Components Used.....	10
5.3. WeMosD1 R2 & Specifications.....	10
5.4. CD74HC4067 & its Specifications.....	11
5.5. Turbidity Sensor.....	12
5.6. PH Sensor.....	13
5.7. TDS Sensor.....	14
5.8. Firebase Database Setup.....	14
5.9. Arduino Sketch Algorithm.....	16
5.10. Algorithm Explanation.....	17
5.11. Firebase Result.....	19
6. Machine Learning Development.....	20-25
6.1. Kaggle Water Quality Datasets.....	20
6.2. ML Model Creation.....	21
6.3. ML Model Conversion.....	24

7.	Android Application Development.....	26-31
7.1.	Connect Application With Firebase.....	26
7.2.	App Design.....	26
7.3.	User Backend Credentials Structure.....	28
7.4.	Working of the Application.....	29
7.5.	Listen for Change In Data Value In Real Time.....	30
7.6.	Android Application Algorithm.....	30
8.	ML With Android Integration.....	32-33
9.	Conclusion.....	34
	Reference.....	35-38
	Appendix.....	39-41
	Terms Related to Water Parameters	
	• <i>Physical Parameters</i>	
	• <i>Chemical Parameters</i>	
	• <i>Biological Parameters</i>	

LIST OF TABLES

SI NO.	CHAPTER NO.	TABLE NAME	PAGE NO.
1	3	Different Water Parameters for Potable and Non-Potable	4-6
2	5	Truth Table for Multiplexer Selector Pins	18-19

LIST OF FIGURES

SI NO.	CHAPTER NO.	FIGURE NAME	PAGE NO.
1	4	lot System Device.	7
		Flow Working of the lot System	8
2	5	Circuit Diagram	9
		Physical Connections of Components	
		WeMos D1 R2 Board	10
		CD74HC4076 Analog/Digital Mux	11
		Turbidity Sensor	12
		pH Sensor and Board	13
		TDS Sensor and Board.	14
		Creating Project.	15
		Agreeing to Terms	
		Creating a Real Time Database	16
		Real Time Database Creation Completed	
		Data From Sensors Sent To Firebase	19
3	6	First 16 rows of Dataset Instances	21
		Last 8 rows of Dataset Instances	
		A Histogram of Different Dataset Features Showing Almost Similar Distribution	22
		Normalization of all the Dataset Features	
		Table Showing the Correlation Between Corresponding Features. No Features are Positively & Negatively Related to Each Other	23

		Dataset Division. Assigning 20% to Test Data	
		Creating a Random Forest Model and Training it	24
		Accuracy of Model on Training and Test Data	
		Specifying Input Layer of the Model	25
		Convert Model to tflite	
		Save Model to tflite file.	
4	7	App Icon in Phone	26
		Logic App Design	27
		Push User Credentials to Firebase	28
		User Credentials Structure in Database	
		Use Case Diagram of the App Working and Design	29
		Listen for Change in Data in Realtime	30
5	8	Import Tensorflow Lite File	32
		ML Model in Android.	33

1. INTRODUCTION

1.1 Objective:

The objective of the project is to design a system that can monitor the water quality using an IoT device equipped with sensors. The system should be able to display the status of the water quality (**potable or non-potable**) on an **Android app**, which can be accessed by both clients and authorized personnel. The system should also be able to provide information on the water quality parameters and detect any deviation from the specified standards. The overall goal of the project is to provide a reliable and efficient solution to monitor water quality and ensure safe and healthy consumption of water.

1.2 Methodology:

The project is to be done by using Turbidity, pH and TDS(Total Dissolved Solids) sensors for taking the input of both the physical, chemical parameters of water. The captured sensor values are to be fed into the microcontroller board(WeMos D1 R2) through an Analog/Digital Multiplexer. The data are to be transmitted in real time to a database. In our case, we used Google “Firebase” Database. The stored data is to be fetched from an android app.

Users will log into the app either as an admin or as a general user. He/She will select one of the rivers displayed on the app so that he/she will view the water quality parameters and/or status of it. The app should be trained using ML/DL algorithms so that it will recognise whether the collected water data is potable or not.

1.3 Proposed System:

In this project we are going to monitor the water quality of River/Stream using an IoT based system and integrate it with an android app. This project is done to help the personnel looking after the water quality of

rivers/streams, to ensure the safety of the water for human consumption as the water may have been polluted to the extent that it has to be monitored for the safety of the people. With the help of this project, the authorities will be able to inform the people about the quality of water.

The sensors will collect the data from these waters and with the help of a microcontroller, we will be able to send these data to a database. From the database, the data will be displayed in the android application and these same data will be fed to the ML Model integrated in the application, which will then predict the current status of the water quality.

2. DIFFERENT WATER QUALITY PARAMETERS

Water is one of the most basic needs for any living organism for survival. Water of good quality is important for health for the people consuming it. The quality of water can be defined in terms of its - Physical, chemical and biological properties. To measure the quality of water, we will consider and use these properties or parameters. They are mentioned as follows:

1. **Physical Parameters:** It includes turbidity, temperature, taste, color, solids and odor.
2. **Chemical Parameters:** It includes pH, chlorine, chlorine residue, sulfate, nitrate, fluoride, iron & manganese, hardness, dissolved oxygen, copper & zinc, Biochemical Oxygen Demand (BOD), Chemical Oxygen Demand (COD), Toxic Inorganic/Organic Substances, alkalinity and radioactive substances (alpha emitters and beta emitters).
3. **Biological Parameters:** It includes the count of different microorganisms like bacteria, protozoa, algae, viruses, E.Coli, Fecal Coliform and salmonella.

3. STANDARDS

By standard, we mean an accepted or approved value of a quantity or quality

The standards of water quality for different uses vary from one another. In this project, we are concentrating only on two different uses of water - **human consumption (drinking & cooking) and other domestic uses** which fall under **Potable** Water. If it does not fall, then it is classified as **Non-Potable**.

The different standards for different parameters are as follows:

Table 3.1: Different Water Parameters for Potable and Non-Potable

Parameters	Potable	Non-Potable
Physical Characteristics		
Turbidity	0.1 - 5 NTU	> 5 NTU
Temperature	10° - 40° C	> 40° C
Taste	No Objectionable Taste	No Objectionable Taste
Colour	Clear (with no noticeable color deposits)	Muddy (with noticeable color deposits)
Solids	< 500 ppm	> 500 ppm
Odor	Agreeable	Non-Agreeable
Chemical Characteristics		
pH	6.5 - 8	> 8
Chloride	< 250 ppm	> 250 ppm
Chlorine residual	0.2 - 0.5 ppm	> 0.5 ppm
Sulphate	< 500 ppm	> 500 ppm

Nitrate		< 10 ppm	> 10 ppm
Fluoride		0.5 - 1 ppm	> 1 ppm
Iron & Manganese		< 0.3 ppm	> 0.3 ppm
Hardness		< 100 ppm	> 100 ppm
Iron		0.3 ppm	> 0.3 ppm
Alkalinity		500 ppm	> 500 ppm
Dissolved Oxygen		5 - 6 ppm	> 6 ppm
Copper & Zinc		1.3 - 3 ppm & 5 ppm	> 3 ppm & > 5 ppm
BOD		1 - 2 ppm	> 2 ppm
COD		N/A	N/A
Toxic Inorganic/Organic Substances		N/A	N/A
Manganese		0.4 ppm	> 0.4 ppm
Alkalinity as (CaCO ₃)		200 ppm	> 200 ppm
Radioactive Substances	Alpha Emitters	0.1 Bq/l	> 0.1 Bq/l
	Beta Emitters	1.0 Bq/l	> 1.0 Bq/l
Biological Characteristics			
Bacteria		100 count / ml	> 100 count / ml
Algae		N/A	N/A

Viruses	N/A	N/A
Protozoa	N/A	N/A
E.Coli	0 count/100 ml	> 0 count/100 ml
Fecal Coliform	0 count/100 ml	> 0 count/100 ml
Salmonella	0 count/100 ml	> 0 count/100 ml

4. OVERALL IoT SYSTEM DESIGN

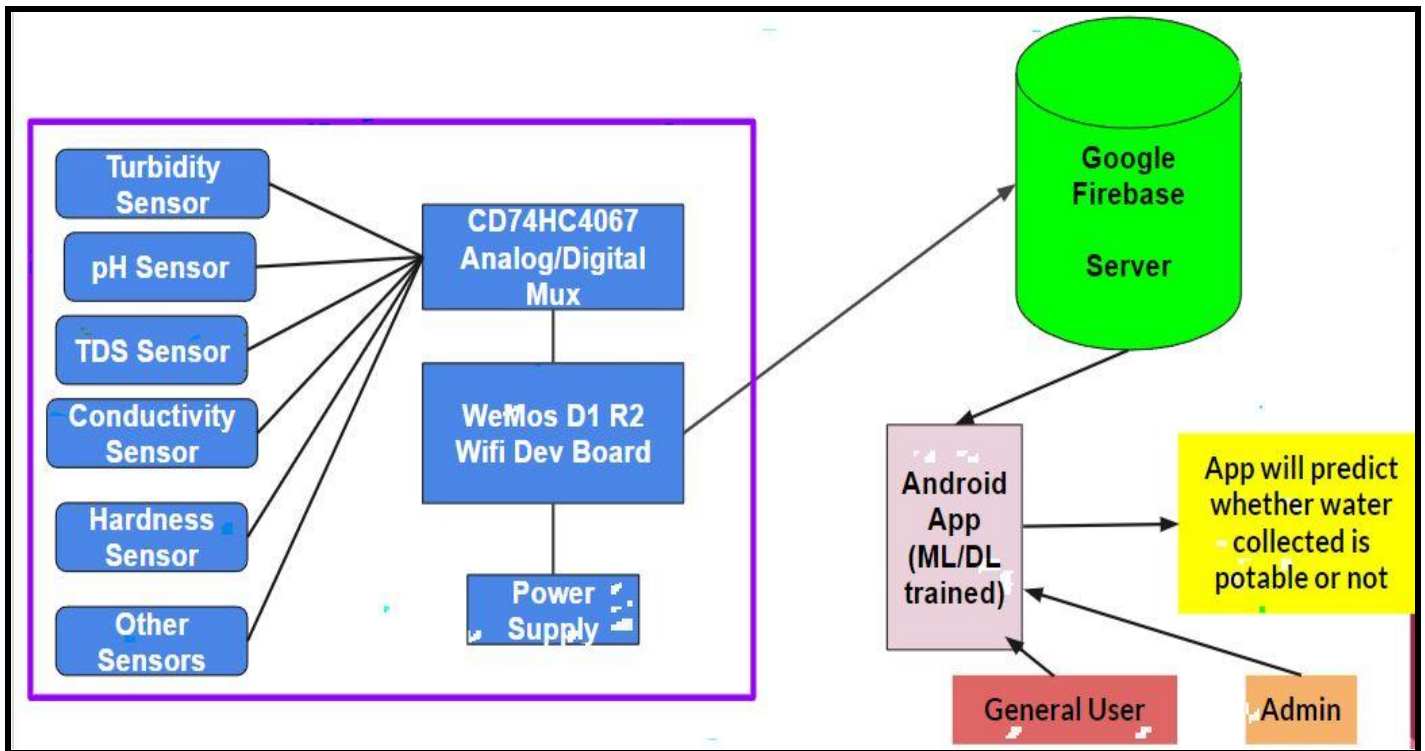


Fig 4.1: Iot System Device.

4.1 Iot System Device:

From the figure given, the different sensors, namely **Turbidity**, **pH**, **TDS**, **Conductivity**, **Hardness** and others, read the voltage values from the water collected and feed them to the microcontroller board (**WeMosD1 R2**) through a multiplexer. The **CD74HC4067 Analog/Digital Mux** will select one of the sensor values at a time and store it in a buffer in the microcontroller. The reason this is done is that the WeMosD1R2 board has only 1 analog pin.

The microcontroller will enable the inbuilt **wifi module(ESP8266)** to transmit the sensor captured values to the Google Firebase in real time.

An **android application** will fetch the “stored data” of all the sensor values in real time. Clients can login the app either as an **administrator** or as a **general user**. Depending on these roles, the app will display the sensor values and/or water status to them.

After the user has logged into the application, he/she will be able to select the **one of the rivers** displayed on the UI page. Based on his/her choice of

the river, all/some water parameters and/or water status of the chosen river will be displayed on the application depending on their Email ID.

The android application has been integrated with an **Random Forest Algorithm (ML) Model** so it can use the “fetched data” from the database as a “test data” and predict the outcome as “Safe” or “Danger” | “Potable” or “Non-Potable”.

4.2 Flowchart:

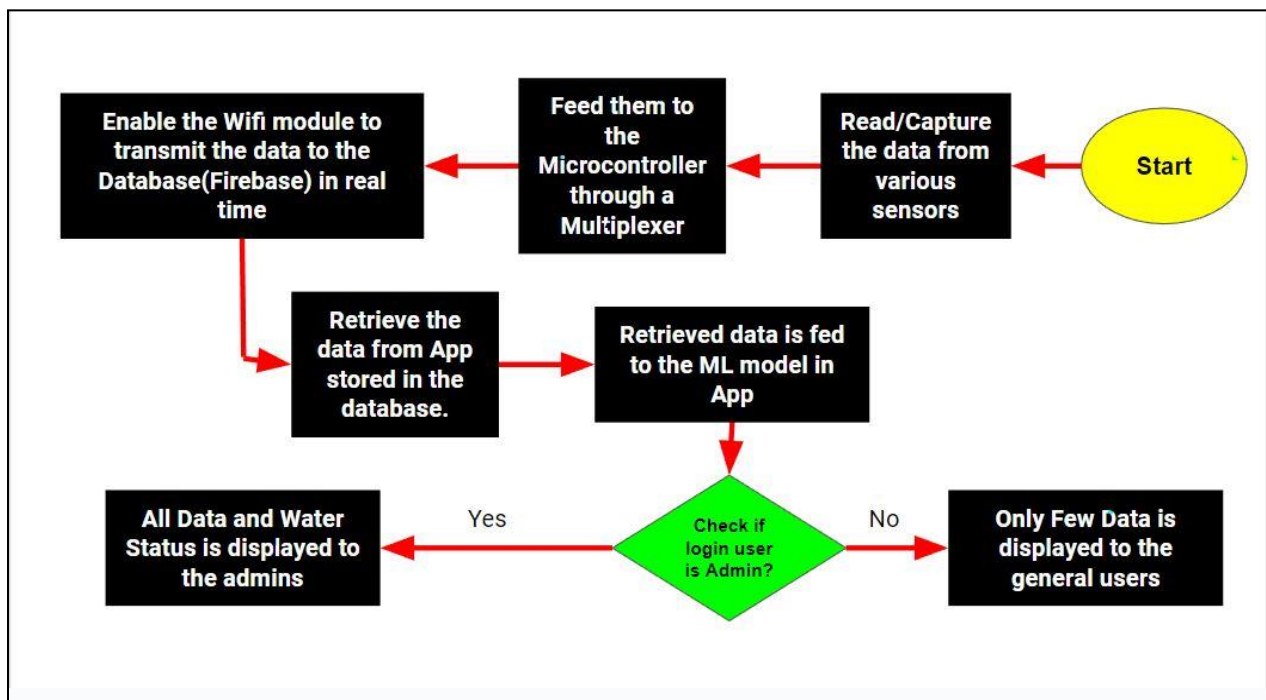


Fig 4.2: Flow Working of the IoT System.

5. HARDWARE DESIGN & DEVELOPMENT

5.1 Hardware Circuit Diagram

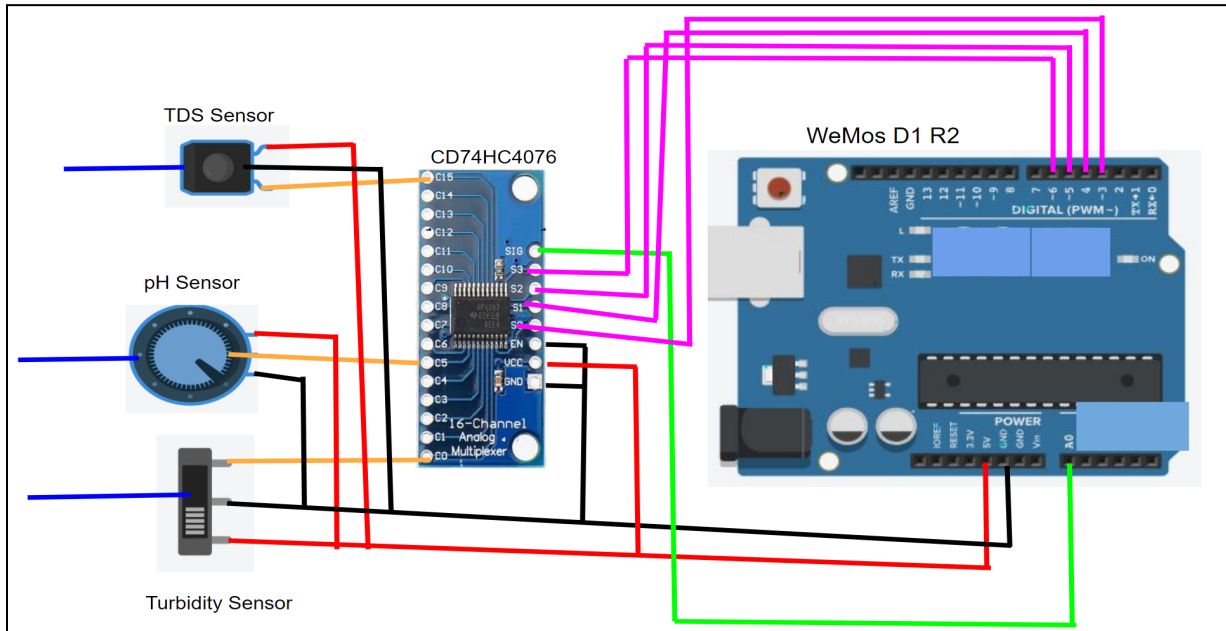


Fig 5.1.1: Circuit Diagram.

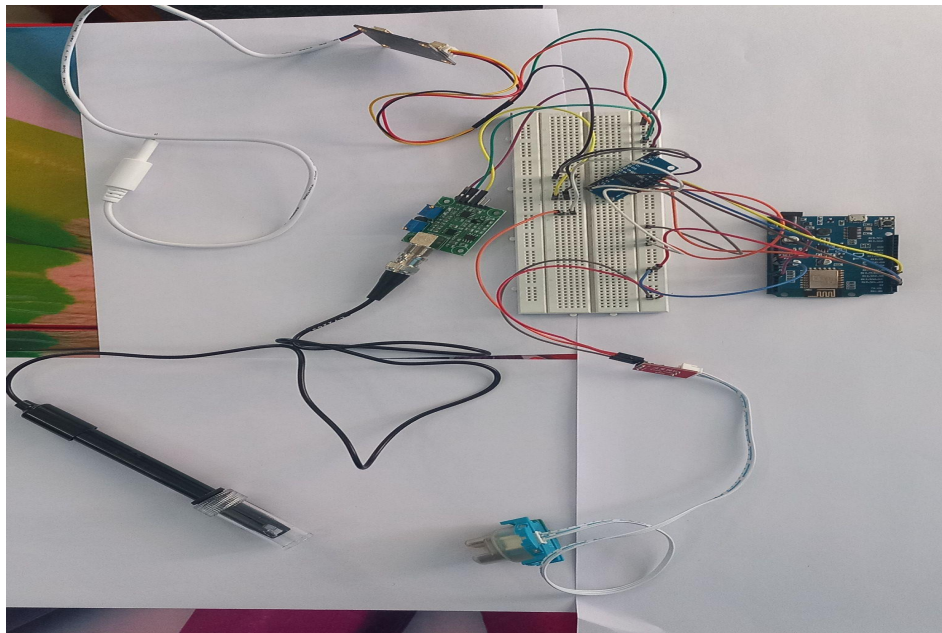


Fig 5.1.2: Physical Connections of Components.

5.2 Components Used:

For the hardware portion, the components which have been used are as follows:

- WeMos D1 R2 Wifi Development Board
- CD74HC4076 Analog/Digital Multiplexer
- Turbidity Sensor
- pH Sensor
- TDS(Total Dissolved Solids) Sensor
- Standard Jumper Wires & Breadboard
- USB(Type B) for powering the WeMos D1 R2 board

Additionally, we used an **Arduino IDE** for interfacing with the WeMosD1R2 and reading the sensor values.

5.3 WEMOS D1 R2:

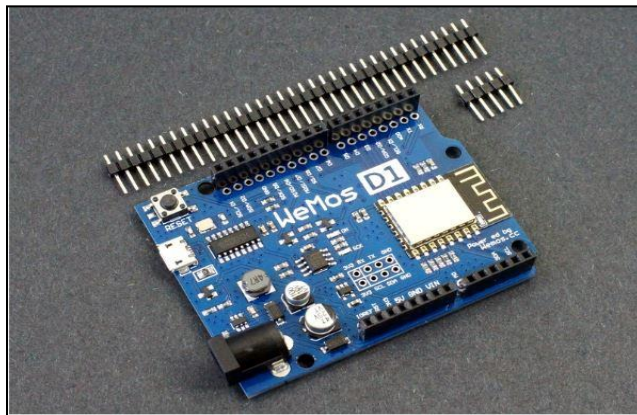


Fig 5.3: WeMos D1 R2 Board.

The ESP8266 D1 R2 WiFi board utilizes the ESP8266 WiFi enabled processor, and puts it onto an Arduino Uno board footprint. It provides a way to work with the ESP8266 in a familiar setup and does not require a breadboard in order to make interconnections since it has the typical on-board female headers. It will also work with some Uno shields that have compatible I/O pin-outs.

With the help of this WiFi Development Board we are able to take the analog data from the sensors and convert them into discrete values that we

can work with.

Specifications

The following are the specifications of the WeMos D1 R2 wifi microcontroller used in this project:

- **Microcontroller:** ESP-8266 32-bit
- **Clock Speed:** 80MHz and up to 160MHz
- **USB Converter:** CH340G
- **Operating Voltage:** 3.3V
- **Flash Memory:** 4MB
- **Digital I/O:** 11
- **Analog Inputs:** 1
- **Communications:** I2C, Serial, SPI
- **WiFi:** Built-in

5.4 CD74HC4076:

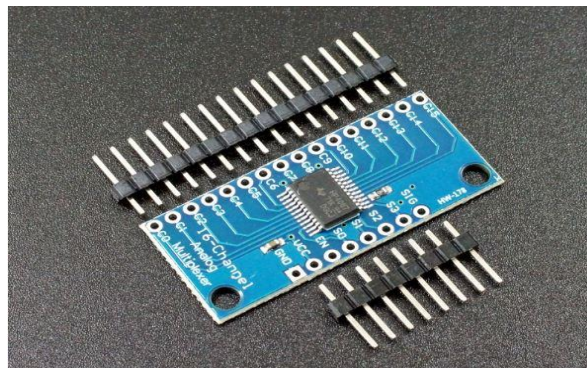


Fig 5.4: CD74HC4076 Analog/Digital Mux.

The CD74HC4067 High-Speed CMOS 16-Channel Analog/Digital Multiplexer Breakout Module is a breakout board for the very handy 16-Channel Analog/Digital Multiplexer/Demultiplexer. This chip is like a rotary switch – it internally routes the common pin (COM in the schematic, SIG on the board) to one of 16 channel pins (CHANxx). It works with both digital and analog signals (the voltage can't be higher than VCC), and the connections function in either direction. To control it, connect 4 digital outputs to the chip's address, select pins (S0-S3), and send it the binary address of the channel you want (see the datasheet for details). This allows you to connect up to 16 sensors to your system using only 5 pins!

Specifications

The following are the specifications of the WeMos D1 R2 wifi microcontroller used in this project:

- **Operating Voltage (VDC):** 2 to 6V
- **Operating Temperature(°C):** -55 to 125
- **PCB Size (L x W) mm:** 40 x 18

5.5 Turbidity Sensor:



Fig 5.5: Turbidity Sensor.

Working: The sensor operates on the principle that when the light is passed through a sample of water, the amount of light transmitted through the sample is dependent on the amount of soil in the water. As the cloudiness level increases, the amount of transmitted light decreases. The turbidity sensor measures the amount of transmitted light to determine the turbidity of the wash water. These turbidity measurements are sent to the microcontroller.

Turbidity Sensor Specifications:

- **Working voltage:** DC 5V
- **Working current:** 30mA [MAX]
- **Response time:** <500 msec
- **Insulation Resistance:** 100M [Min]
- **Operating Temperature (°C)** -30 ~ +80
- **Length (mm):** 33
- **Width (mm):** 20

- Height (mm): 12
- Weight (gm): 55

5.6 PH Sensor:

Working: The Ph Sensor is often made of glass and has a rod-like construction with a bulb at the bottom that holds the sensor. A glass bulb that is specifically made to be selective to hydrogen-ion concentration is present in the glass electrode used to measure pH. Hydrogen ions in the test solution exchange with other positively charged ions on the glass bulb upon immersion in the solution under test, creating an electrochemical potential across the bulb. The electrical potential difference between the two electrodes created during the test is detected by the electronic amplifier, which transforms it into pH units. The Nernst equation states that the electrochemical potential across the glass bulb is linearly linked to pH

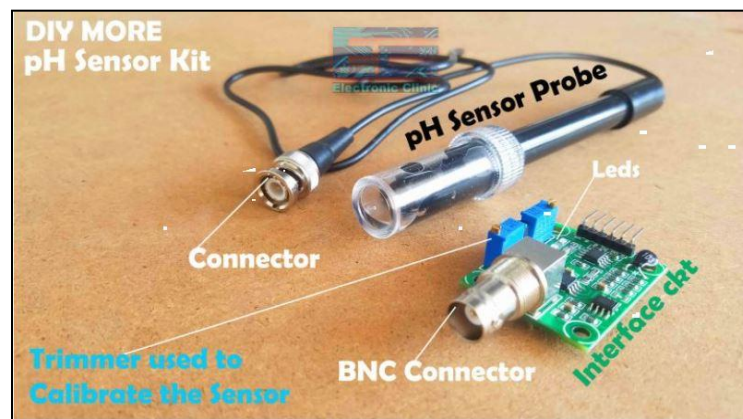


Fig 5.6: pH Sensor and Board.

Specifications:

- pH Range: 0-14 pH
- Operating Temperature (C): 0 to 80
- Zero Point: 70.5 pH
- Alkali Error: 0.2 pH
- Theoretical Percentage Slope: 98.5%
- Internal Resistance: 250M
- Response Time: 1min
- Terminal Blocks: BNC plug
- Total Probe Length (m): Approx 1m
- Weight (gm): 56

5.7 TDS Sensor:

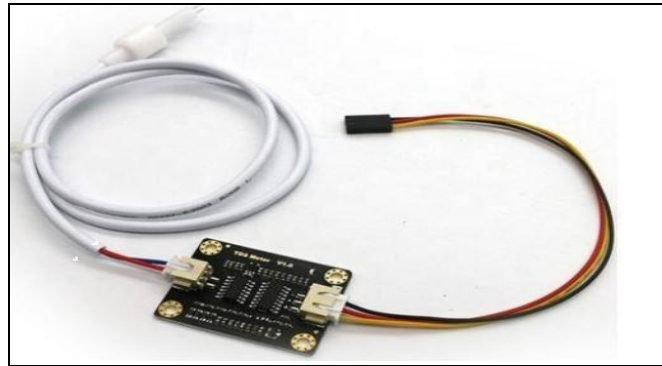


Fig 5.7: TDS Sensor and Board.

Working: A TDS meter is basically an electrical charge (EC) meter whereby two electrodes equally spaced apart are inserted into water and used to measure charge. The result is interpreted by the TDS meter and converted into a ppm figure. If the water contains no soluble materials and is pure, it will not conduct a charge and will, therefore, have a 0 ppm figure. Conversely, if the water is full of dissolved materials, it will conduct a charge, with the resulting ppm figure being proportional to the number of dissolved solids. This is because all dissolved solids have an electrical charge, which allows conduction of electrical charge between the electrodes.

TDS Specifications:

- **Input Voltage:** 3.3 ~ 5.5V
- **Output Voltage:** 0 ~ 2.3V
- **Working Current:** 3 ~ 6mA
- **TDS Measurement Range:** 0 ~ 1000 ppm
- **TDS Measurement Accuracy:** $\pm 10\%$ FS (25 °C)
- **TDS probe with Number of Needle:** 2

5.8 Firebase Setup:

Google Firebase is a comprehensive mobile and web application development platform provided by Google. But before creating a Real-Time Database, **one should have a Google Account.**

To set up the Database for our project, the following steps were followed:

- **Create a Firebase Project:**

- Visit the Firebase Console at console.firebase.google.com.
- Click on the "Add project" button
- A name for the project was provided and desired region was selected
- "Create project" button was clicked to create the Firebase project.

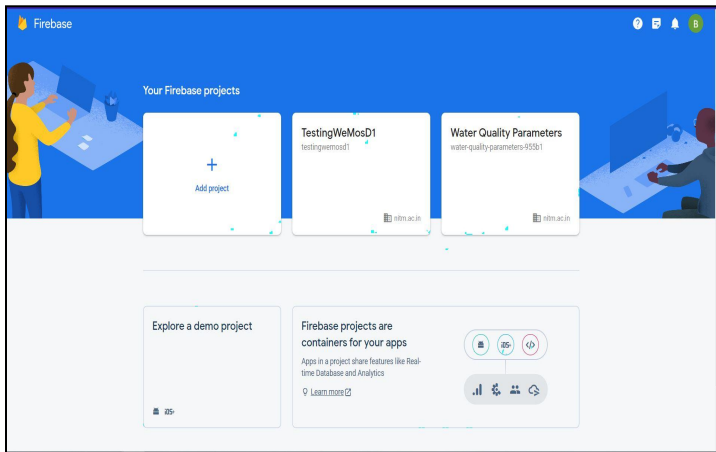


Fig 5.8.1: Creating Project.

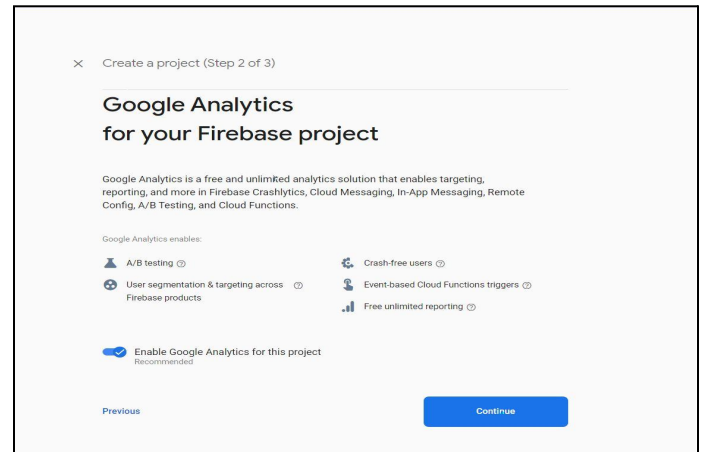


Fig 5.8.2: Agreeing to Terms.

- **Enable Realtime Database:**

- In the Firebase project dashboard, click on the "Develop" option in the left-hand menu.
- From the sub-menu, select "Database" to access the Realtime Database section.
- If prompted, choose the "Realtime Database" option (not Firestore) and click on the "Enable" button.
- At this point, a Realtime Database dashboard was observed where we could manage our database.

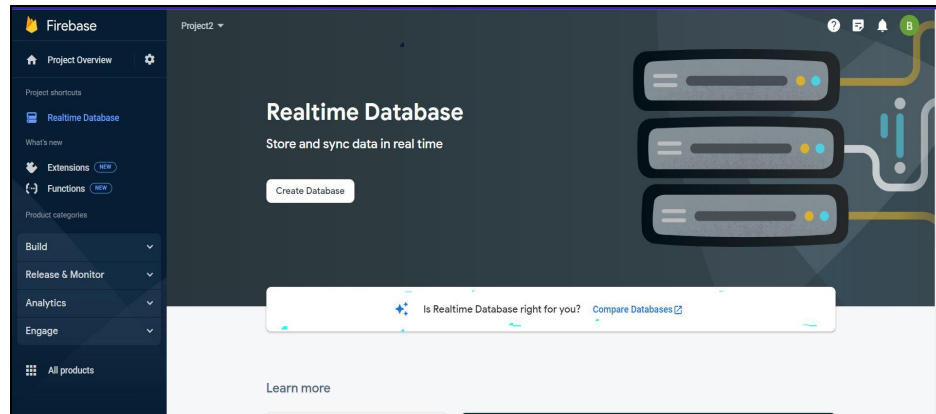


Fig 5.8.3: Creating a Real Time Database.

- **Choosing Database Rules:**
 - In the Realtime Database dashboard -> "Rules" tab, here the rules were updated to 'write' and 'read.'

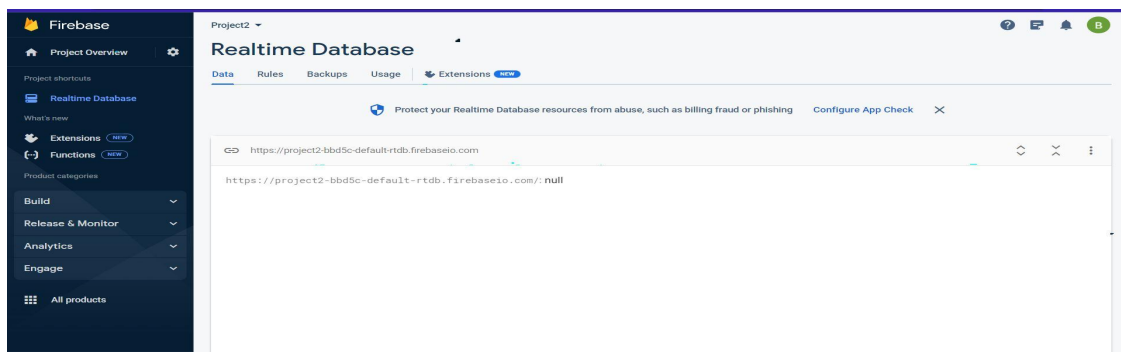


Fig 5.8.4: Real Time Database Creation Completed.

5.9 Arduino Sketch Algorithm:

For reading and transmitting real time sensor values, the following algorithm and codes(arduino sketch) was used and followed:

Algorithm:

- Include libraries for TDS, ESPWifi, EEPROM & Firebase
- Initialise Wifi & Firebase database credentials
- Define Pins for controlling Selector and Output Pins on Multiplexer.

Also declare objects of the GravityTDS Class.

- **SETUP:**
 - Connect to Wifi network & Firebase database until connected .
Similarly, set the pin mode of the WeMosD1R2 digital pins to “output” to the multiplexer selector pins.
- **LOOP:**
 - Enable Selector Pins on channel **i** of Multiplexer & read voltage values from turbiditySensor through its subroutines.
 - Enable Selector Pins on channel **j** of Multiplexer & read voltage values from pHSensor through its subroutines.
 - Enable Selector Pins on channel **k** & read voltage values from TDS Sensor through its subroutines.
 - Read values from other sensors as well.
 - Transmit the captured values to the Firebase Database
 - Jump to LOOP again

Note: **i, j, k** should not be equal to each other.(0 <= **i,j,k** <= 15)

5.10 Algorithm Explanation:

The algorithm given is for the microcontroller(WeMosD1R2) to control the sensors in order for them to take the input data from the turbidity, PH and TDS sensor which will transfer them wirelessly to the Google Firebase.

But first, we had set up some requirements in the **Google Firebase** so that it can authenticate and allow us to store data. We extract the Firebase Host link and Authorization key. We use them to include in the program.

Initially, libraries for ESPWifi, Firebase, TDS and EEPROM were included. Global variables for wifi details, firebase credentials , Multiplexer I/O pins and object of the GravityTDS class were declared.

In the **SETUP** part, the code will initiate a wifi connection and continue to do so until it is finally connected to the internet. Similarly it will try to connect to the Firebase until it achieves success. An object of the gravityTDS class is

also initialized.

In the **LOOP**, channel **i** is turned on using the selector pins to read values from turbidity. Similarly, the same goes for pH and TDS sensors except that a different channel is used (**j & k**).

Since **we do not have the budget to purchase other 6 sensors**, random values are used instead.

After collecting all 9 necessary values, the method of the Firebase class, `Firebase.setFloat()` is used to transmit the data one by one for each 9 values in real time. It returns 1 if successfully transmitted to the Firebase. After all 9 values have been transmitted, the entire section under LOOP will run again.

turbiditySensor(): In this subroutine, the sensor will read the voltage values. It is then converted into NTU(Nephelometric Turbidity Unit). The relationship between the output voltage and NTU is given by: $NTU = (-1120.4 * outputvoltage^2) + (5742.3 * outputvoltage) - 4352.9$.

phSensor(): In this subroutine, first a calibration value is calculated after physically calibrating the pH sensor. The sensor will then read voltage values. The relationship between the output voltage and pH value is given by: $PH_value = -5.70 * voltage + calibration_value$.

tdsSensor(): In this subroutine, an already inbuilt class “GravityTDS” for capturing/reading TDs values was used. The code just updates and reads the values.

Multiplexing: A combination of 16 4-bit binary digits are used to select the channel for allowing the sensor values to pass through the Mux to the microcontroller. But in our project, only **3 combinations** were used. The truth table for the selector pins on the multiplexer are as follows

Table 5.10: Truth Table for Multiplexer Selector Pins

Analog Pin for Operation	Binary Combination			
	S0	S1	S2	S3
C0	0	0	0	0
C1	1	0	0	0

C2	0	1	0	0
C3	1	1	0	0
C4	0	0	1	0
C5	1	0	1	0
C6	0	1	1	0
C7	1	1	1	0
C8	0	0	0	1
C9	1	0	0	1
C10	0	1	0	1
C11	1	1	0	1
C12	0	0	1	1
C13	1	0	1	1
C14	0	1	1	1
C15	1	1	1	1

5.11 Firebase Result:

After the complete execution of the Arduino sketch, the below data are received on the Database.

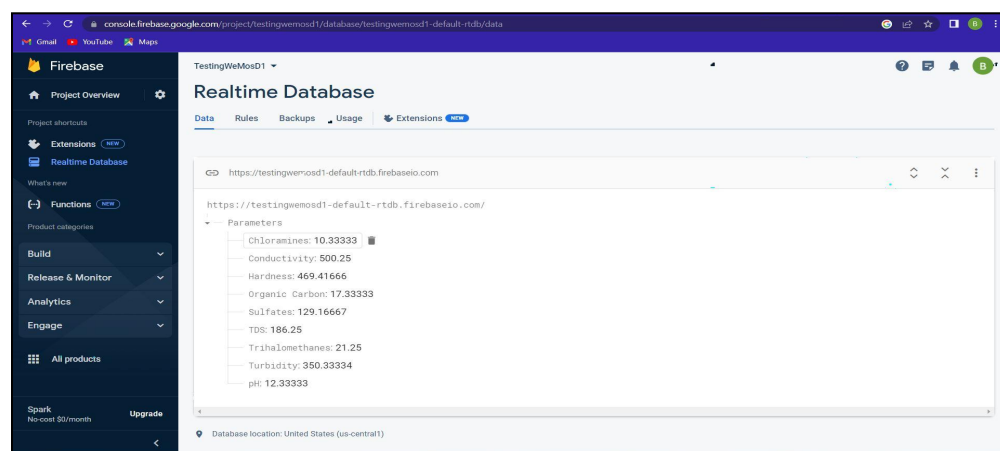


Fig 5.11: Data From Sensors Sent To Firebase.

6. MACHINE LEARNING DEVELOPMENT

6.1 Kaggle Water Quality Datasets

The dataset to be used for training the Android App (Machine Learning/Deep Learning Portion) has been collected from <https://www.kaggle.com/datasets/adityakadiwal/water-potability> . It contains 9 features/attributes, namely:

1. **pH:** Evaluates the acid-base balance of water. It ranges from 0-14 as per the pH scale.
2. **Hardness:** Mainly caused by calcium and magnesium salts.
3. **Total Dissolved Solids(TDS):** Water contains many dissolved substances including some organic minerals or salts like potassium, calcium, sodium, bicarbonates, chlorides, magnesium, sulfates, etc. All these substances constitute the TDS. It is Measured in Mg / L
4. **Chloramines:** They are major disinfectants in public water systems. Chloramines are measured in Mg / L or ppm.
5. **Sulfates:** Naturally occurring substances found in minerals, soil and rocks. It is also measured in Mg / L
6. **Conductivity:** Pure water is not a good conductor of electricity. Electrical conductivity (EC) actually measures the ionic process of a solution that enables it to transmit current. It is measured in $\mu\text{S}/\text{cm}$
7. **Trihalomethanes(THM):** THMs are chemicals which may be found in water treated with chlorine. The concentration of THMs in drinking water varies according to the level of organic material in the water. It is measured in ppm.
8. **Organic Carbon:** Total Organic Carbon (TOC) in source waters comes from decaying natural organic matter (NOM) as well as synthetic sources. It is measured in Mg / L
9. **Turbidity:** The turbidity of water depends on the quantity of solid matter present in the suspended state. It is measured in NTU (Nephelometric Turbidity unit).

The dataset also has the target label, “**Potability**”. This indicates if water is safe for human consumption or not. “1” means Potable and “0” means not potable.

ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
NaN	204.890455	20791.318981	7.300212	368.516441	564.308654	10.379783	86.990970	2.963135	0
3.716080	129.422921	18630.057858	6.635246	NaN	592.885359	15.180013	56.329076	4.500656	0
8.099124	224.236259	19909.541732	9.275884	NaN	418.606213	16.868637	66.420093	3.055934	0
8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	4.628771	0
9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0
5.584087	188.313324	28748.687739	7.544869	326.678363	280.467916	8.399735	54.917862	2.559708	0
10.223862	248.071735	28749.716544	7.513408	393.663396	283.651634	13.789695	84.603556	2.672989	0
8.635849	203.361523	13672.091764	4.563009	303.309771	474.607645	12.363817	62.798309	4.401425	0
NaN	118.988579	14285.583854	7.804174	268.646941	389.375566	12.706049	53.928846	3.595017	0
11.180284	227.231469	25484.508491	9.077200	404.041635	563.885481	17.927806	71.976601	4.370562	0
7.360640	165.520797	32452.614409	7.550701	326.624353	425.383419	15.586810	78.740016	3.662292	0
7.974522	218.693300	18767.656682	8.110385	NaN	364.098230	14.525746	76.485911	4.011718	0
7.119824	156.704993	18730.813653	3.606036	282.344050	347.715027	15.929536	79.500778	3.445756	0
NaN	150.174923	27331.361962	6.838223	299.415781	379.761835	19.370807	76.509996	4.413974	0
7.496232	205.344982	28388.004887	5.072558	NaN	444.645352	13.228311	70.300213	4.777382	0
6.347272	186.732881	41065.234765	9.629596	364.487687	516.743282	11.539781	75.071617	4.373348	0

Fig 6.1.1: First 16 rows of Dataset Instances.

ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
6.702547	207.321086	17246.920347	7.708117	304.510230	329.266002	16.217303	28.878601	3.442983	1
11.491011	94.812545	37188.826022	9.263166	258.930600	439.893618	16.172755	41.558501	4.369264	1
6.069616	186.659040	26138.780191	7.747547	345.700257	415.886955	12.067620	60.419921	3.669712	1
4.668102	193.681735	47580.991603	7.166639	359.948574	526.424171	13.894419	66.687695	4.435821	1
7.808856	193.553212	17329.802160	8.061362	NaN	392.449580	13.903225	NaN	2.798243	1
9.419510	175.762646	33155.578218	7.350233	NaN	432.044783	11.039070	69.845400	3.298875	1
5.126763	230.603758	11983.869376	6.303357	NaN	402.883113	11.168946	77.488213	4.708658	1
7.874671	195.102299	17404.177061	7.509306	NaN	327.459760	16.140368	78.698446	2.309149	1

Fig 6.1.2: Last 8 rows of Dataset Instances.

6.2 ML Model Creation

Random Forest Algorithm was used in creating the model for classifying the data. Some of the rows of the dataset contain null values which belong to the columns, namely “pH”, “Sulfates” and “Trihalomethanes”. To fill such gaps, an average of a required column was taken and calculated.

Data Visualization was performed and it is observed that most of the data

have similar distribution. But since some columns have different units of measurement, **normalization** is performed using `MinMaxScaler()`.

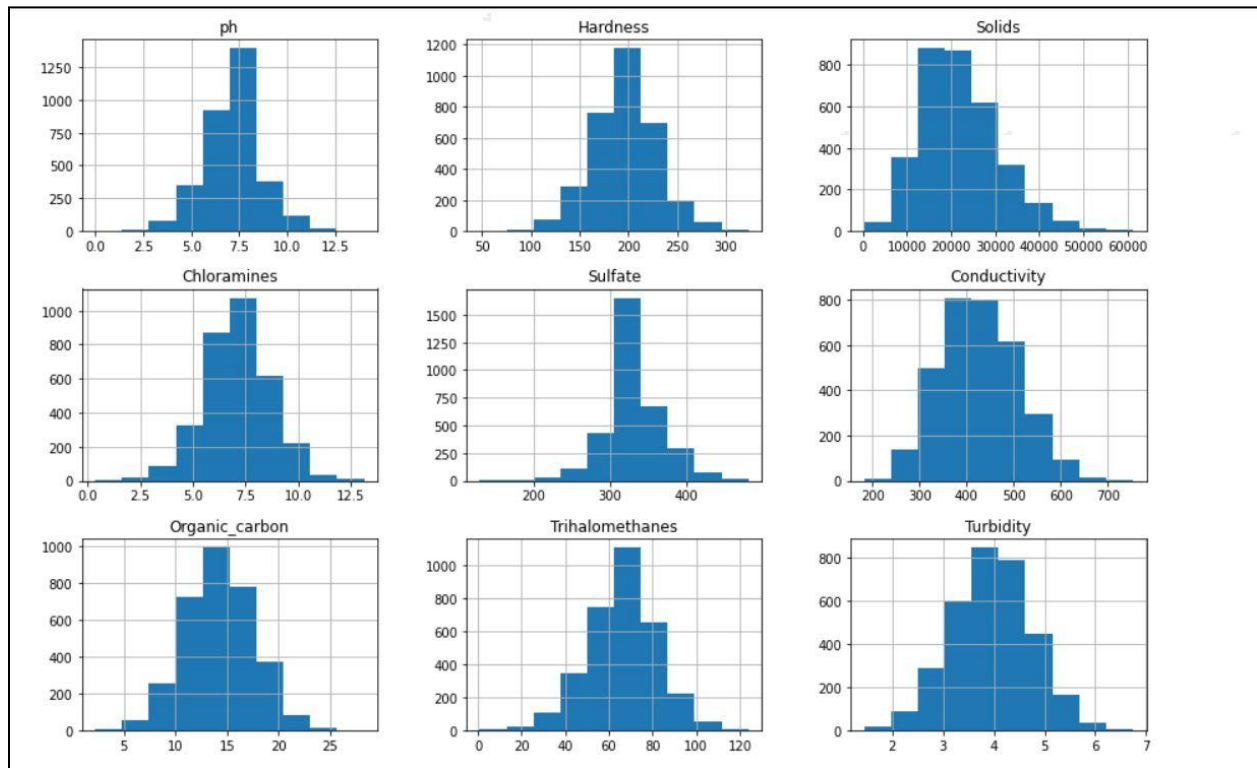


Fig 6.2.1: A Histogram of Different Dataset Features Showing Almost Similar Distribution.

```
from sklearn import preprocessing

scaler = preprocessing.MinMaxScaler()
water_quality_df_norm = pd.DataFrame(data= scaler.fit_transform(water_quality_df), columns=water_quality_df.columns)
water_quality_df_norm.head(10)
```

Fig 6.2.2: Normalization of all the Dataset Features.

Feature Correlation was done using heatmap function(belongs to the seaborn library). It was found out that no features are related to each other. Hence **reduction** in the number of features could not be done.

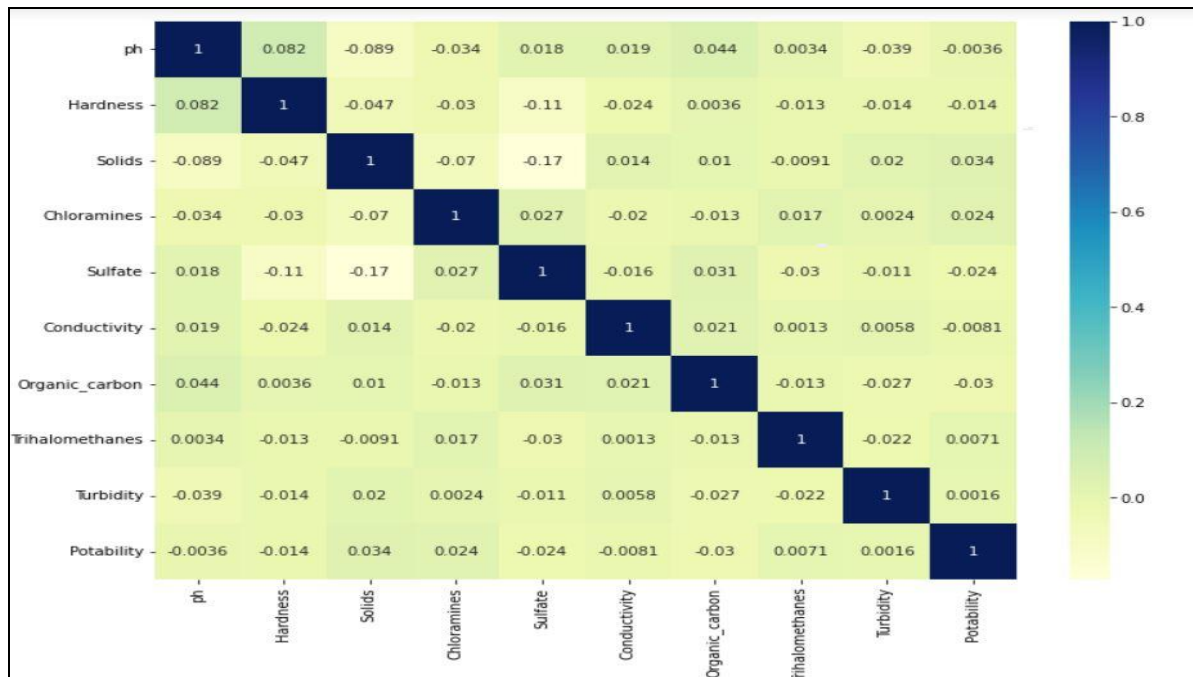


Fig 6.2.3: Table Showing the Correlation Between Corresponding Features. No Features are Positively & Negatively Related to Each Other.

The “testing” and “training” data have been divided in the ratio of 20:80. 20% to the “test data” and 80% for “training data”

```
water_X = water_quality_df_norm.drop('Potability', axis=1)
water_Y = water_quality_df_norm['Potability']

# For training and testing dataset
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(water_X, water_Y, test_size=0.2, random_state=101)
```

Fig 6.2.4: Dataset Division. Assigning 20% to Test Data.

RandomForestClassifier (from ScienceKit Learn Library) was used in making the model. An “estimator/decision trees” of “200” and “criterion” of “gini” was chosen to train the model.

```
from sklearn.ensemble import RandomForestClassifier

random_forest_model = RandomForestClassifier(n_estimators = 200, criterion = 'gini')
random_forest_model.fit(X_train, Y_train)

RandomForestClassifier(n_estimators=200)
```

Fig 6.2.5: Creating a Random Forest Model and Training it.

The **training and testing** score for this “Random Forest Classifier” model is found out to be “100 %” and “69.36 %” respectively.

```
print(f'Training Score = {random_forest_model.score(X_train, Y_train) * 100}')
print(f'Testing Score = {random_forest_model.score(X_test, Y_test) * 100}')

Training Score = 100.0
Testing Score = 69.35975609756098
```

Fig 6.2.6: Accuracy of Model on Training and Test Data.

The “Random Forest” model is now ready to be used. But first, it is to be converted into **.tflite** format so that it can be integrated with the android application.

6.3 ML Model Conversion

Out of all the trained ML/DL Models on these specific water potability data sets acquired from Kaggle, which is the same data set required, the Random Forest Classification Model is the one that gives the highest accuracy, with a score of 69%. But this Model could not be used directly with an Android Application. This is because the ML model is large in size, and thus might make the Application take a lot of storage space, especially when loaded into the RAM, and this situation is not ideal when it comes to mobile devices with a small memory size. Another problem of loading the ML Model directly into Android Application is the Computational Resources. ML Models required a large amount of computational resources to run. This can lead to

many difficulties. Mobile devices do not have that amount of computational power to run these models. And moreover, this amount of computation will drain the battery very fast. Also, the using of ML Models directly with Android Applications might lead to compatibility issues. Thus, because of these reasons, the ML Model first needs to be converted to something that is compatible and can be used with Android Applications. The ML Models first need to be converted into files that can be understood by Android.

There are a couple of ways this can be done. But, in this particular project, Tensorflow Lite was used. There are a number of reasons why Tensorflow Lite is used to convert the ML Models. Firstly, it optimizes particularly for mobile devices and can run models efficiently. Secondly, it provides tools for model compression and quantization which greatly reduce the size of the model. It is cross-platform, easy to integrate, flexible and lastly it is open source. Tensorflow Lite converts the model to a **.tflite** extension which can be used directly with Android Applications.

The input to the model (Model_Name.tflite) is first define according to the input of the trained model using keras:

```
inputs = tf.keras.Input(shape=(9,))
outputs = tf.keras.layers.Dense(1, activation='sigmoid')(inputs)
keras_model = tf.keras.Model(inputs=inputs, outputs=outputs)
```

Fig 6.3.1: Specifying Input Layer of the Model.

The Model is then converted to a **tflite** model using the TFLiteConverter, a class provided by Tensorflow for converting ML Models to Tensorflow Lite:

```
converter = tf.lite.TFLiteConverter.from_keras_model(keras_model)
tflite_model = converter.convert()
```

Fig 6.3.2: Convert Model to tflite.

This model is then saved to a file with a **.tflite** extension which can then be used with Android Applications:

```
with open('random_forest_model.tflite', 'wb') as f:
    f.write(tflite_model)
```

Fig 6.3.3: Save Model to tflite file.

7. ANDROID APPLICATION DEVELOPMENT

The Android Application was developed in Android Studio, an official integrated development environment (IDE) for developing Android Applications, using the Java programming language. The **Android Application is first set up with firebase** to allow fetching and sending data to its Real Time Firebase. The minimum SDK is set to 21 and the maximum SDK is set to 30. The minimum SDK is set to 21 because to integrate an ML Model to Android, the minimum SDK should not be less than 19. The App name is **SafeDrop** with the **Water Drop Icon** as shown below.

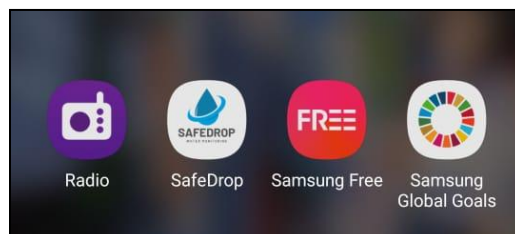


Fig 7.1: App Icon in Phone.

7.1 Connect Application With Firebase:

While creating a project in firebase, the app is register with that particular project by filling in the packet name of the app - **com.example.appname**, the **SHA-1** value of the App in the firebase **SHA-1** field and Downloading the **google-service.json**. The **google-service.json** is then pasted in the **My Application/app** folder in the Android Application Project. Add the firebase SDK to **Android/Gradle Grid/build.gradle** folder of the Android Project. The next step is to go to **tools** in the Android Studio Navigation bar and Select Firebase, a couple of options are available, select **Realtime Database** and connect the App with Firebase Realtime Database.

7.2 App Design:

The Application consists of 5 activities, namely, the Login Activity, the Registration Activity, Menu Activity, General Main Activity and the Admin Activity.

The **Login Activity** is for the user to login their credentials. Phone number and a password is used.

The **Registration Activity** is for any user to register with the Application. No user without completing the registration process can access the application content. The Registration Activity takes the Full Name, the Phone Number, Email Address and the password for the application from the user.

The **Menu Activity** is for the user to choose from the Areas available to access the water status or water information of a particular River.

The **General Activity** is for the General user. It displays only a few characteristics of the water.

The **Admin Activity** Displays all the characteristics of the water present in the database and also displays the water status.

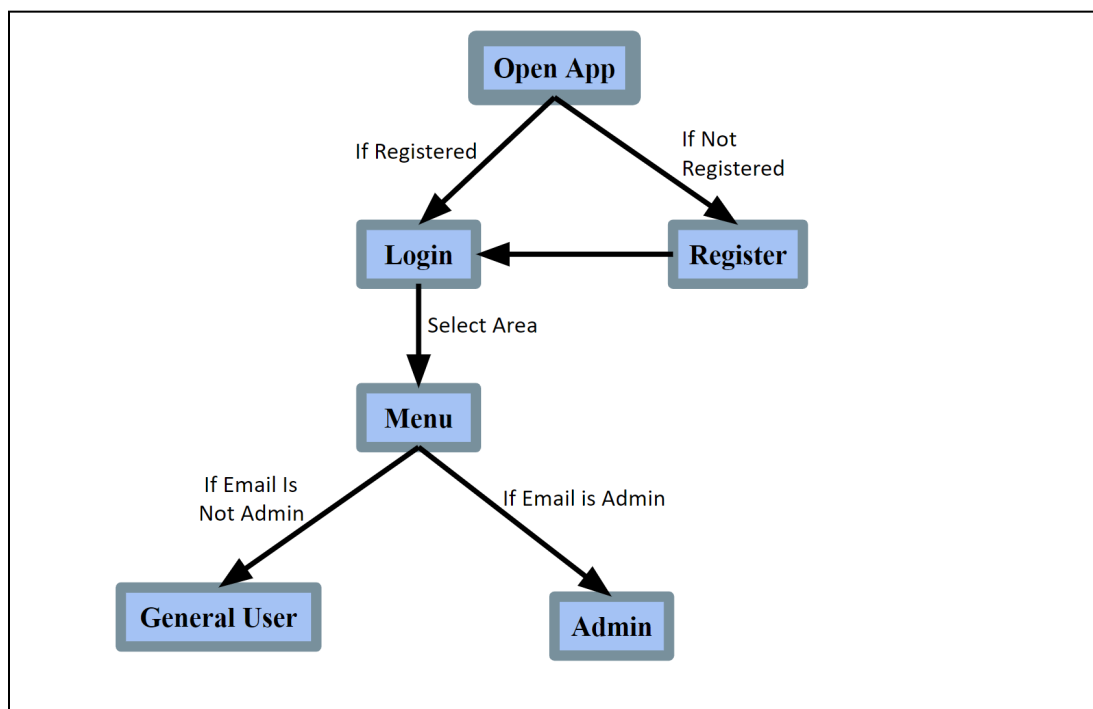


Fig:7.2.1: Logic App Design.

7.3 User Backend Credentials Structure:

For the Backend, Google Realtime Firebase is used to store all the water data and the users credentials.

The users act as the root. Phone Number acts as the Parent of all the credentials (Password, Email and Full Name). These Data are sent from the application during registration.

The data are sent to the database using the following code:

```
DatabaseReference databaseReference =
FirebaseDatabase.getInstance().getReferenceFromUrl("database_url");

databaseReference.child("users").addListenerForSingleValueEvent(new
 ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot snapshot) {
databaseReference.child("users").child(phoneNumber).child(otherCredentials).set
tValue(value);
    })
```

Fig 7.3.1: Push User Credentials to Ffirebase.

The data is stored in the database as specified in the code above.

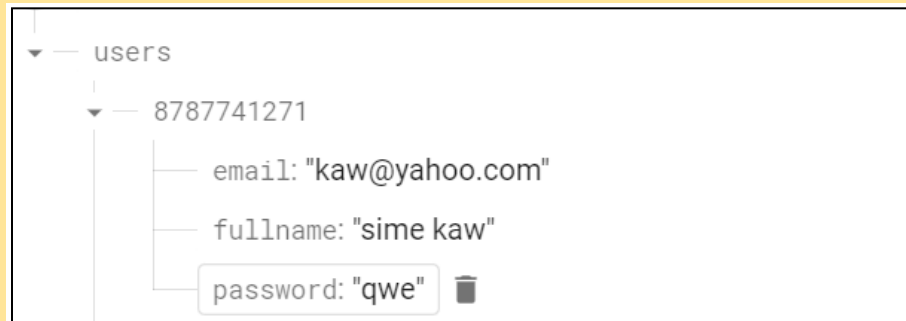


Fig 7.3.2: User Credentials Structure in Database.

7. 4 Working Of Application:

The app launcher Activity is the Login Activity. The users can then enter their credentials or if not yet registered, they can click on the register button to complete the registration process and then Login. Once logged in, the User will then be taken to the Menu Activity where they can choose the area they choose to see the water status of. The email of the user is stored when they logged in. Now, once the user clicks on the area of their choice, the application will check their email ID. If the email ID belongs to the admin, the Admin Activity is launched where all the water information available will be displayed and the water status is also displayed or else the user is taken to the General Main Activity where only a little information is displayed. The user can go back to the Menu Activity to select another river. Once they choose to logout, they can do so from the General Mail Activity and Admin Activity for General User and for Admins respectively.

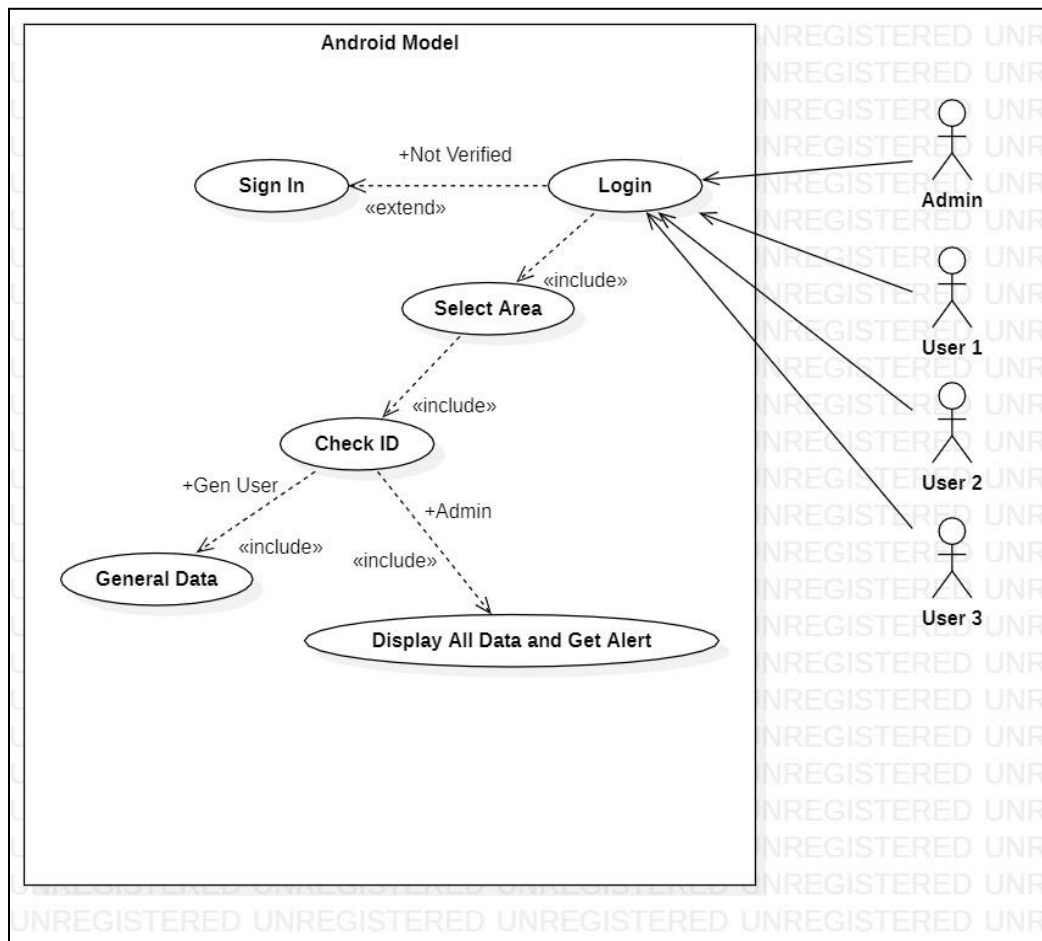


Fig 7.4.1: Use Case Diagram of the App Working and Design.

The water status in the Admin Activity is predicted using the ML *Random Forest Classification* Model integrated into the application. The inputs (in float values) to this model are the data in the database that was being pushed by the sensors from those water bodies. The output of the model is a float value of either 1.0 or 0.0 depending on the inputs. 1.0 means that the water is potable and is safe for consumption. 0.0 means that water is non-potable and water is not safe for human consumption. Based on these two outputs, the water status in the application is set to SAFE or DANGER.

7.5 Listen For Change In Data Value In Real Time:

The Input to the Model or the fetching of Data is done every time there is a change in the data in the firebase. Thus, give the proper information to the users. The code to listen to this changes is:

```
DatabaseReference databaseReference =
FirebaseDatabase.getInstance().getReferenceFromUrl("firebase_database_url");

databaseReference.child(root).addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot snapshot) {

        // Fetching Data from the Firebase
        float waterInfo =
snapshot.child("Parameters").child("waterInfo").getValue(Float.class);
    });
```

Fig 7.5.1: Listen for Change in Data in Realtime.

Everytime there is change in data in the firebase realtime database the data is fetched and the Model is invoked to make the prediction.

7.6 Android Application Algorithm

The Algorithm used in designing and developing the **Android Application** is given below:

Initialization:

- Link the Views with IDs to variables in Java files.

- Store Firebase Realtime Database to a Variable.

Saving Credentials and ID Checking:

- Save the User Credentials when logged in into a Bundle.
- Passed the Bundle to Menu Activity.
- The Menu Activity then retrieved these Credentials.
- These Credentials are Used to check the Email ID.
- Based on the Email ID the respective Activity is launched.

Fetching Data:

- Once in the `GeneralMainActivity` or in the `Admin` Activity the Data fetching from the Database starts.
- Fetch each Data of a river using the above ***databaseReference*** variable.
- Store the Data into another variable.

Display Fetched Data:

- The fetched Data are displayed in their respective Views.

Feeding Data to Model:

- Store the fetch Data using Byte Buffer.
- Feed Input to the Model.
- Run the Model.

Model Result and Prediction

- Store Model output to a float variable.
- Display Water Status.

Close Model

- Close the Model to free resources.

8. ML WITH ANDROID INTEGRATION

The Model is integrated with the application by importing the random_forest.tflite into android studio. Once import is done, it automatically installs all the other dependencies required to run the Model into the application

The tensorflow lite file is imported into android studio by going to **File -> New -> Others -> Tensorflow Lite Model**. Select the file from the folder it is in and click next the file is imported.

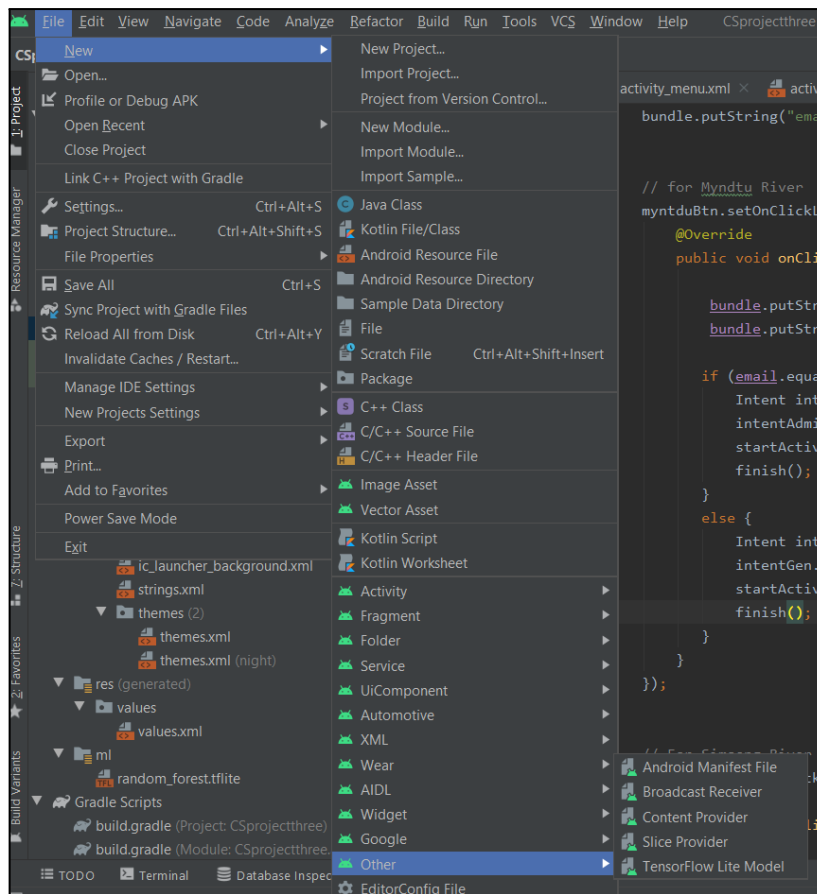


Fig 8.1: Import Tensorflow Lite File.

Once, Import is done, the file with the picture below will appear in the **app/ml** directory of the android project,

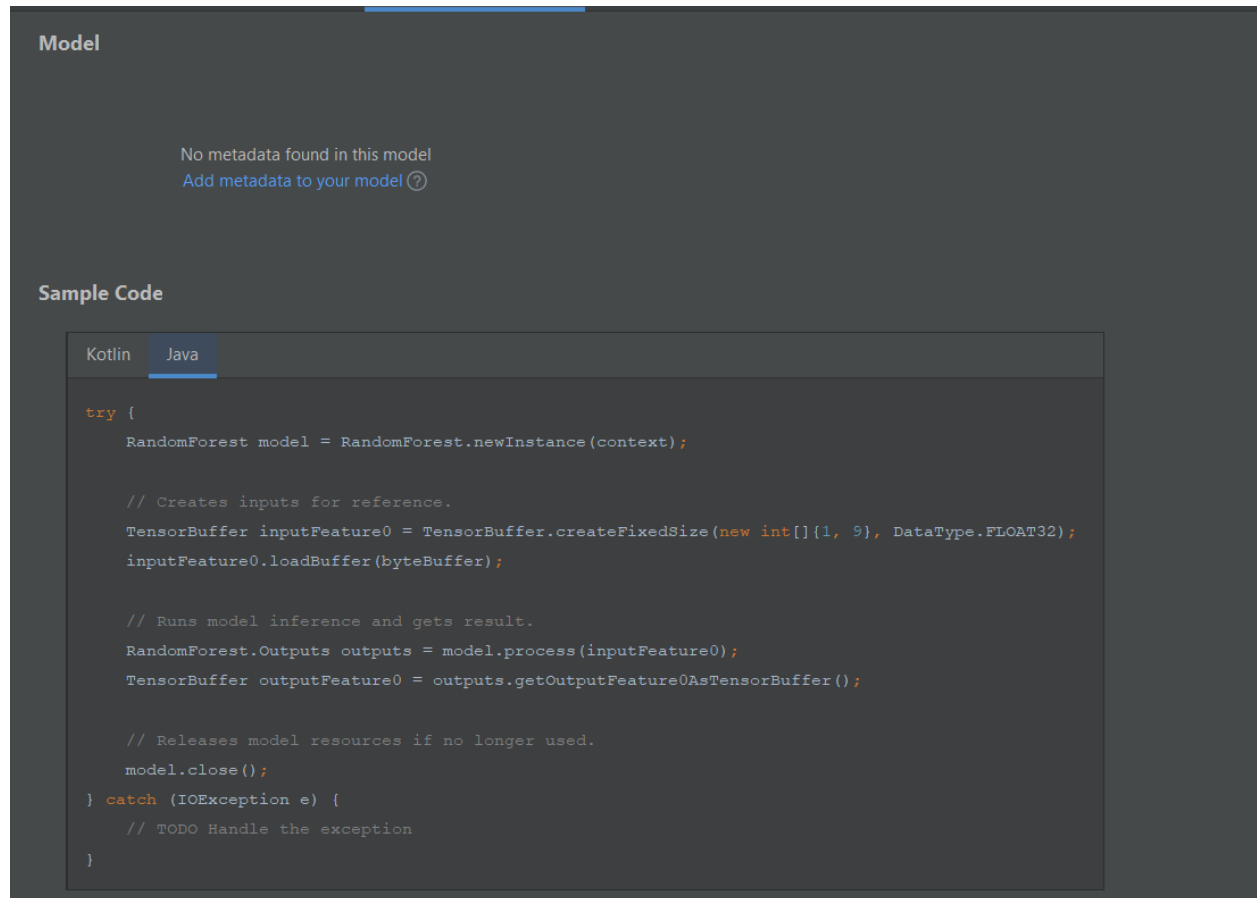


Fig 8.2: ML Model in Android.

The Java code in this file is then pasted into the Java file and Import the necessary packets, replace `getApplicationContext()` and define the ByteBuffer for the input to the model and thus, ML Model is integrated with the Android Application.

9. CONCLUSION

Using the microcontroller, WeMos D1 R2, CD74HC4076 Analog/Digital Mux and along with the turbidity, PH and TDS sensor we are able to collect the pH, Turbidity, Total Dissolved Solid, etc, value of water and push the collected data into Google Firebase Realtime Database. The collected data is then fetched by the Android Application whenever there is change in any of the parameter's values and fed to the Random Forest ML Model. Based on the Model output (1.0 or 0.0) the water status is set to Safe (1.0) or Danger (0.0).

This project, "IoT Design And Integrated App Development For Water Quality Monitoring", if implemented can help monitor the water quality in real time and thus helps to ensure a safe drinking water for the people using these waters in their everyday lives. It makes it easy for the personnels looking after the department of water safety to be informed about the standard of the water, and if the water standards deviate from the recommended standard, they can then easily inform the people about the particular issue and thus prevents many people from falling ill by consuming water of unchecked standard and in turn improved the living standard of the people.

REFERENCES

1. Water Quality Parameters

- a. Confirming Turbidity levels for drinking water
 - i. <https://www.h2olabcheck.com/blog/view/turbidity>
- b. Confirming Sulfate levels for drinking water
 - i. https://www.epa.gov/sites/default/files/2014-09/documents/support_cc1_sulfate_healtheffects.pdf
- c. Confirming Electrical Conductivity levels for drinking water
 - i. <https://nmtracking.doh.nm.gov/environment/water/PHConductivity>.
- d. Confirming Hardness level for drinking water
 - i. https://apps.who.int/iris/bitstream/handle/10665/70168/WHO_HSE_WSH_10.01_10_Rev1_eng.pdf
- e. Confirming Trihalomethanes levels for drinking water
 - i. <https://dhss.delaware.gov/dph/files/trihalomfaq.pdf>
- f. Confirming Organic Carbon levels for drinking water
 - i. <https://www2.gov.bc.ca/assets/gov/environment/air-land-water/water/waterquality/water-quality-guidelines/approved-wqgs/organic-carbon-tech.pdf>
- g. Confirming TDS(Total Dissolved Solids) levels for drinking water
 - i. <https://www.bisleri.com/blog-detail/understanding-tds-and-its-role-in-drinking-water>
- h. Confirming Chloramines levels for drinking water
 - i. https://www.cdc.gov/healthywater/drinking/public/water_disinfection.html
- i. Confirming pH levels for drinking water
 - i. <https://cdn.who.int/media/docs/default-source/wash-documents/wash-chemicals/ph.pdf>

- j.** Confirming Oxygen levels Demand for drinking water
 - i. <https://www.sciencedirect.com/topics/earth-and-planetary-sciences/chemical-oxygen-demand>
- k.** Confirming Nitrate levels for drinking water
 - i. <https://www.health.state.mn.us/communities/environment/water/contaminants/nitrate.html>
- l.** Confirming Fluoride levels for drinking water
 - i. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3593199/#:~:text=According%20to%20World%20Health%20Organization,water%20is%200.5%E2%80%931%20ppm.>
- m.** Confirming Iron levels for drinking water
 - i. <http://www.idph.state.il.us/envhealth/factsheets/ironFS.html>
- n.** Confirming Manganese levels for drinking water
 - i. <https://www.iowadnr.gov/Portals/idnr/uploads/water/wso/docs/ManganeseFactSheetandFAQ.pdf>
- o.** Confirming Copper levels for drinking water
 - i. https://www.health.wa.gov.au/Articles/A_E/Copper-in-drinking-water#:~:text=These%20guidelines%20set%20two%20levels,prevent%20any%20health%20related%20problems
- p.** Confirming Zinc levels for drinking water
 - i. <https://scdhec.gov/sites/default/files/docs/HomeAndEnvironment/Docs/Zinc.pdf>
- q.** Confirming Bacteria level for drinking water
 - i. https://portal.ct.gov/-/media/Departments-and-Agencies/DPH/dph/environmental_health/pdf/04BacteriainPrivateDrinkingWaterWells0409pdf.pdf

2. Hardware Design & Development

- a. Reading Turbidity values
 - i. <https://how2electronics.com/diy-turbidity-meter-using-turbidity-sensor-arduino>
- b. Understanding WeMosD1R2 specifications
 - i. <https://protosupplies.com/product/esp8266-d1-r2-wifi-processor-with-uno-footprint/>
- c. Understanding CD74HC4076
 - i. <https://solarduino.com/increase-analog-pins-for-nodemcu-using-16-channel-analog-multiplexer-module/>
- d. Reading ph Values
 - i. <https://www.electronicclinic.com/ph-meter-arduino-ph-meter-calibration-diymore-ph-sensor-arduino-code/>
- e. Reading TDS values
 - i. <https://how2electronics.com/tds-sensor-arduino-interfacing-water-quality-monitoring/>
- f. Setting up of Firebase
 - i. <https://innovationyourself.com/send-real-time-data-to-firebase/>
- g. Sending Data to Firebase
 - i. <https://how2electronics.com/send-sensor-data-google-firebase-esp8266/>
- h. Interfacing with Arduino
 - i. <https://www.freecodecamp.org/news/create-your-own-electronics-with-arduino-full-course/>

3. Kaggle Dataset

- a. <https://www.kaggle.com/datasets/adityakadiwal/water-potability>

4. Machine Learning

a. Building Random Forest Model

- i. <https://www.datacamp.com/tutorial/random-forests-classifier-python>

b. Understanding Machine Learning

- i. <https://www.simplilearn.com/tutorials/machine-learning-tutorial/machine-learning-steps>

5. Android Development

a. Convert ML Model to tflite:

- i. <https://www.tensorflow.org/lite/models/convert>

b. Deploy tflite in Android Application:

- i. <https://medium.com/the-stem/ml-in-android-3-deploy-tflite-on-android-using-java-6a18431644b6>

c. Responsive Display:

- i. <https://www.geeksforgeeks.org/scrollview-in-android/>

d. Passing Data Between Activities:

- i. <https://www.geeksforgeeks.org/how-to-send-data-from-one-activity-to-second-activity-in-android/>

e. Set The Application Icon:

- i. <https://developer.android.com/codelabs/basic-android-kotlin-training-change-app-icon>

APPENDIX

TERMS RELATED TO WATER PARAMETERS

The terms and their meaning related to all their parameters are given below.

Physical Parameters:

1. **Turbidity:** Turbidity is the cloudiness of water. It is a measure of the ability of light to pass through water.
2. **Temperature:** Measure of the kinetic energy of water and is expressed in degrees Fahrenheit (F) or Celsius (C).
3. **Taste:** Normally, Water is believed to be *tasteless*. If water tastes sweet, sour, bitter or any other flavor then that can be because of two reasons: Minerals and chemicals present alters water taste.
4. **Color:** Color is measured by comparing the water sample with standard color solutions or colored glass disks.
5. **Solids:** Solids occur in water either in solution or in suspension. These two types of solids can be identified by using a glass fiber filter that the water sample passes through. By definition, the suspended solids are retained on the top of the filter and the dissolved solids pass through the filter with the water.
6. **Odor:** Sensation that is due to the presence of substances that have an appreciable vapour pressure and that stimulate the human sensory organs in the nasal and sinus cavities.

Chemical Parameters:

1. **pH:** It defines how acidic or alkaline the water is.
2. **Chlorine:** Chloride occurs naturally in groundwater, streams, and lakes, but the presence of relatively high chloride concentration in freshwater (about 250 mg/L or more) may indicate wastewater pollution.
3. **Chlorine residual:** In drinking water, a residual of about 0.2 mg/L is optimal. The residual concentration which is maintained in the water distribution system ensures good sanitary quality of water.

4. **Sulfate:** Sulfate ions (SO_4^{2-}) occur in natural water and in wastewater. If high concentrations are consumed in drinking water, there may be objectionable tastes or unwanted laxative effects, but there is no significant danger to public health.
5. **Nitrate:** Nitrate can be present in some lakes, rivers and underground water. High concentration of nitrate in water can be harmful.
6. **Fluoride:** A moderate amount of fluoride ions (F^-) in drinking water contributes to good dental health. High concentration of fluoride in water can be harmful for human consumption.
7. **Iron & Manganese:** Iron (Fe) and Manganese (Mn) do not cause health problems, but they impart a noticeable bitter taste to drinking water even at very low concentration.
8. **Hardness:** The amount of dissolved calcium and magnesium in the water.
9. **Dissolved oxygen:** The amount of oxygen that is present in the water.
10. **Copper & Zinc:** Copper (Cu) and zinc (Zn) are nontoxic if found in small concentrations. Actually, they are both essential and beneficial for human health and growth of plants and animals. They can cause undesirable effects in the water if present in high concentration,
11. **Biochemical Oxygen Demand (BOD):** Higher BOD indicates more oxygen is required, which is less for oxygen-demanding species to feed on, and signifies lower water quality. Inversely, low BOD means less oxygen is being removed from water, so water is generally purer.
12. **Chemical Oxygen Demand (COD):** The amount of oxygen consumed when the water sample is chemically oxidized.
13. **Toxic Inorganic/Organic Substances:** A wide variety of inorganic toxic substances may be found in water in very small or trace amounts. Even in trace amounts, they can be a danger to public health. There are more than 100 compounds in water that have been listed in the literature as toxic organic compounds. They will not be found naturally in water; they are usually man-made pollutants.
14. **Alkalinity:** A measure of the ability of the water body to neutralize acids and bases and thus maintain a fairly stable pH level.
15. **Radioactive substances (alpha emitters and beta emitters):** Radioactive substances are atoms that decay naturally. They can give off alpha particles, beta particles and gamma radiation.

Biological Parameters:

1. **Bacteria:** A lot of dangerous waterborne diseases are caused by bacteria. Typhoid and paratyphoid fever, leptospirosis, tularemia, shigellosis, and cholera are all caused by bacteria.
2. **Protozoa:** Parasitic protozoa that are transmitted through water and those can cause human infections.
3. **Algae:** Certain types of algae, if present in water can cause harm to the health of humans and animals.
4. **Viruses:** Some viral pathogens can cause waterborne diseases if water is not treated properly.
5. **E.Coli:** Cause severe disease and may be fatal in small children and the elderly.
6. **Fecal Coliform:** Coliform bacteria are organisms that are present in the environment and in the feces of all warm-blooded animals and humans. Coliform bacteria will not likely cause illness. However, their presence in drinking water indicates that disease-causing organisms (pathogens) could be in the water system.
7. **Salmonella:** Salmonella can affect the intestinal tract if present in drinking water.