WATER MANAGEMENT SYSTEM

ECD416: PROJECT PHASE II REPORT

submitted by

HARI MENON

TCR20EC029

SHINO SHAJU

TCR20EC052

ALAN NELSON

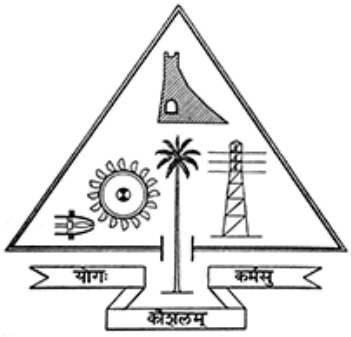
TCR20EC010

ANIRUDH SREERAM

TCR20EC015

to

the APJ Abdul Kalam Technological University
in partial fulfillment of the requirements for the award of the Degree
of
Bachelor of Technology
In
Electronics and Communication Engineering



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

GOVERNMENT ENGINEERING COLLEGE THRISSUR

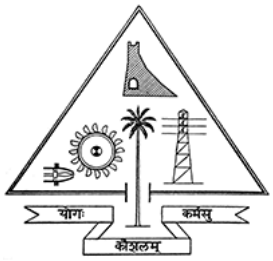
KERALA

JUNE 2024

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

GOVERNMENT ENGINEERING COLLEGE THRISSUR

THRISSUR - 680 009,



Certificate

*This is to certify that this is a bonafide record of the Project Work titled "**Water Management System**" done by*

Shino Shaju	**Reg. No. TCR20EC052**
Hari Menon	**Reg. No. TCR20EC029**
Alan Nelson	**Reg. No. TCR20EC010**
Anirudh Sreeram	**Reg. No. TCR20EC015**

*of fourth year B. Tech. Electronics and Communication Engineering in partial fulfilment of the requirement for the award of **Bachelor of Technology Degree in Electronics and Communication Engineering** under the **A. P. J. Abdul Kalam Technological University** for the year 2024.*

Guide :
Dr. Roy Francis
Associate Professor
Dept. of ECE
Government Engineering College
Thrissur

Project Coordinator:
Prof. Vijeesh V
Assistant Professor
Dept. of ECE
Government Engineering College
Thrissur

Head Of Department:
Dr. Sinith M.S
Associate Professor
Dept. of ECE
Government Engineering College
Thrissur

Declaration

We undersigned hereby declare that the project report (Water Management System), submitted for partial fulfillment of the requirements for the award of the degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under supervision of Dr. Roy Francis, Associate Professor, Government Engineering College, Thrissur. This submission represents our ideas in our own words and where ideas or words of others have been included; we have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

GEC THRISSUR
Date : 29-04-2024

Hari Menon
Shino Shaju
Alan Nelson
Anirudh Sreeram

Acknowledgement

We wish to record our indebtedness and thankfulness to all those who helped us prepare this project titled Water Management System and present it in a satisfactory way. First and foremost we thank Dr. Sinith M.S., Head Of the Department, Dept of Electronics and Communication Engineering for giving us the opportunity to present this project. We extend our gratitude to Dr. Roy Francis, Associate Professor, Dept of Electronics and Communication Engineering for his valuable guidance, encouragement, and cooperation during the course of this project and its presentation. It was his support that resulted in the successful presentation of this project. Finally, we would like to extend our sincere gratitude towards teachers, friends of Dept of Electronics and Communication Engineering and our family members who have always been helpful.

GEC Thrissur
Date : 29-04-2024

Hari Menon
Shino Shaju
Alan Nelson
Anirudh Sreeram

Abstract

Our project is an automated water management system that reduces water wastage, improves water quality and measures important parameters like, Total Dissolved Solids, turbidity and pH. It includes an integrated system with automated pressure sensors, pumps, UV purification and water quality sensors. It also automatically fills the tank when the water level goes below a threshold level. A user-friendly interface is included for real-time monitoring and control and a central database will be maintained for future analysis and a machine learning model to predict the water potability. This project is intended to be finally implemented in the ECE department of GEC Thrissur. Further application includes its usage in households, office buildings etc.

Contents

List of Figures

List of Tables

Abbreviations

IDE	Integrated Development Environment
PWM	Pulse Width Modulation
TDS	Total Dissolved Solids
QDA	Quadratic Discriminant Analysis
USB	Universal Serial Bus
VS	Visual Studio
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
ML	Machine Learning
GND	Ground

1 Introduction

This chapter lists the objectives of the project. A brief background study and the scheme of the project are also included.

1.1 Objectives

- **Optimize Water Usage:** Implement a prototype or scaled-down version of an Automated Water Management System to minimize water wastage by addressing issues like tap leakages, pipe leakages, and overflow.
- **Real-time Water Quality Monitoring:** Design a system with sensors for instant monitoring of crucial water quality parameters, ensuring adherence to predefined standards.
- **IoT Integration:** Incorporate Internet of Things (IoT) technology for efficient data transmission, remote monitoring, and real-time alerts, aligning with emerging technological trends.
- **User-Friendly Interface and Database:** Develop an accessible interface for real-time control and monitoring. Implement a centralized database for historical data storage, facilitating informed decision-making and system improvement.

1.2 Background

In response to the growing global concern regarding responsible water management and the need for sustainable solutions, our project addresses these challenges with a focus on the Electronics and Communication Engineering (ECE) Department at Government Engineering College (GEC) Thrissur. Water, a vital resource, plays a critical role in supporting life and various activities, necessitating effective and efficient management strategies. The project's primary objectives center around reducing water wastage, enhancing water quality, and introducing advanced monitoring capabilities within the ECE Department. Recognizing the importance of automated systems in achieving these goals, the project integrates cutting-edge technologies such as automated sensors, UV purification, and water quality sensors. These components collectively contribute to a comprehensive Automated Water Management System designed to optimize water usage, maintain high water quality standards, and ensure a reliable supply.

Furthermore, the incorporation of automated tank refilling mechanisms addresses issues related to water level fluctuations, guaranteeing a continuous and uninterrupted water supply. The implementation of UV purification technology as an additional layer of treatment underscores the commitment to providing clean and safe water within the department.

To empower users with real-time insights and control, a user-friendly interface has been developed, allowing staff and administrators to actively monitor and manage the water system. Additionally, a centralized database has been established to store historical data, enabling future analysis and informed decision-making for the continuous improvement of the system.

As environmental sustainability becomes a priority in educational institutions, the adoption of this Automated Water Management System aligns with the progressive vision of GEC Thrissur's ECE Department. By pioneering this initiative, the department not only sets an example for responsible water management but also contributes to the broader movement towards sustainable practices in academic institutions.

1.3 Problem Statements

- **Water Wastage:** Addressing wasteful practices like tap and pipe leakages, the project implements an Automated Water Management System to promote efficient water usage.
- **Limited Water Resources:** Responding to the strain on water resources from increasing demands, the project offers a solution for effective water resource management through real-time monitoring.
- **Water Quality Monitoring:** Focusing on key parameters, the project ensures water quality by monitoring Dissolved Oxygen, turbidity, and ammonia levels within the ECE department.
- **Manual Monitoring Challenges:** Overcoming the drawbacks of manual methods, the project automates monitoring processes, providing accurate real-time data for informed decision-making.
- **Lack of Comprehensive System:** Addressing the absence of an integrated solution, the project proposes a holistic Automated Water Management System, streamlining monitoring, control, and data analysis.
- **Environmental Impact:** Tackling concerns of water pollution and environmental impact, the project contributes to sustainability by reducing wastage and enhancing water quality.
- **Technological Integration:** Embracing IoT, the project leverages advanced technology for efficient data transmission, remote monitoring, and real-time alerts, reflecting a forward-looking approach.

1.4 Scope

The scope of this project is extensive, encompassing the design and implementation of an Automated Water Management System with advanced monitoring features. The system's capabilities include real-time measurement and control of crucial water quality parameters such as Dissolved Solids, turbidity, and pH levels. Additionally, it integrates automated sensors, pumps, and UV purification for efficient water usage, reduced wastage, and continuous supply. The project's scope extends to the incorporation of IoT technology, emphasizing its potential for practical application in solving water management challenges. With a user-friendly interface for monitoring and a centralized database for data analysis, the project not only addresses the immediate needs of the ECE department at GEC Thrissur but also sets the stage for broader implications in sustainable water resource management. This project can also be implemented for the entire college as well in the future.

2 Relevance

The project holds significant relevance in addressing critical issues related to water management and quality monitoring. In a world facing increasing water scarcity and pollution, the development and implementation of an Automated Water Management System with advanced monitoring capabilities contribute substantially to sustainable water usage and conservation. By incorporating sensors for parameters such as Dissolved Solids, turbidity, and pH levels, the system ensures real-time monitoring and control, essential for maintaining water quality. The integration of automated sensors, pumps, and UV purification enhances efficiency, reduces wastage, and guarantees a continuous water supply.

Furthermore, the project aligns with the global emphasis on IoT-based solutions, showcasing the practical application of cutting-edge technology in solving real-world challenges. The use of a user-friendly interface for monitoring and a centralized database for data analysis emphasizes the importance of informed decision-making for water resource management.

In the context of the ECE department at GEC Thrissur, this project serves as an exemplary model for leveraging technology to address environmental concerns. It not only provides a solution for water management within the department but also offers a potential blueprint for other institutions seeking sustainable practices. The relevance of this project extends beyond the classroom, impacting the broader conversation on responsible water management and technological innovation for environmental sustainability.

3 Literature Survey

[1] explores the broad applications of water resources monitoring, spanning river sections, water sources, groundwater, water functionality, social water users, and sewage treatment plants. This monitoring relies on automatic technologies focusing on data collection, analysis, transmission, and application. The core emphasis is on water resource data application and management, notably within water resources management centers. These centers leverage advanced technologies to collect, analyze, and process data, generating graphical statements, reports, and documents. These outputs serve as valuable resources for management departments, addressing informational needs and emphasizing the crucial role of effective data management in informed decision-making and sustainable water resource practices.

[2] explores the design and implementation of a continuous water quality monitoring system with Dissolved Oxygen (DO) and pH sensors, a PCDuino microcontroller, a sample collection unit, and a PC-based graphical display. The central PCDuino microcontroller controls data transmission and the automatic sampling unit, operating based on a comparison of measured parameters against predefined thresholds. The significance of monitoring DO and pH is highlighted, with probes triggering an alert signal if values fall below the threshold. It underscores the integration of advanced sensor technologies, microcontroller-based control, and real-time graphical display for effective water quality monitoring, emphasizing practical applications for environmental management.

[3] explores the evolving landscape of smart water quality monitoring, influenced by advancements in communication technology. Numerous recent studies are reviewed, providing a comprehensive overview of state-of-the-art smart monitoring systems. It introduces a novel, power-efficient in-pipe monitoring solution using Internet of Things (IoT)

technology, designed for simplicity and enhanced energy efficiency. This model not only tests water samples but also analyzes uploaded data over the Internet. A notable feature is its ability to alert remote users in case of deviations from predefined water quality standards. [3] emphasizes ongoing efforts to integrate IoT into water quality monitoring, highlighting technical innovations and practical implications for water safety and security.

[4] focuses on the critical necessity of water consumption for both humans and other living organisms, emphasizing the limited availability of water resources due to pollution. To address this issue and ensure the sustainability of water functions, an integrated system based on the Internet of Things (IoT) is proposed for real-time water quality monitoring using sensors. The use of Raspberry Pi as an embedded system facilitates the manufacturing of detecting sensor devices, and remote communication technology enables seamless data transmission between devices. This IoT water quality monitoring system operates as an automated solution for surface water, providing essential real-time data for effective environmental management.

[5] focuses on the escalating water crisis driven by growing populations, increased agricultural and industrial demands, and the subsequent exacerbation of the situation. Existing efforts, such as automatic public tap control, aim to curb water wastage.The Automatic Water Management System (AWMS) is introduced as a solution to monitor and control water wastage arising from issues like tap leakage, pipe leakage, and water tank overflow. The proposed AWMS is designed with efficient sensors to swiftly address water wastage concerns, making it a valuable addition to buildings by providing real-time water level information to property owners. [5] underscores the critical role of water monitoring systems in mitigating the global water crisis, emphasizing the need for efficient solutions like AWMS to combat water wastage.

4 General Architecture

The Chapter presents the General architecture of the project. It also includes the component selection and lists the hardware and software chosen.

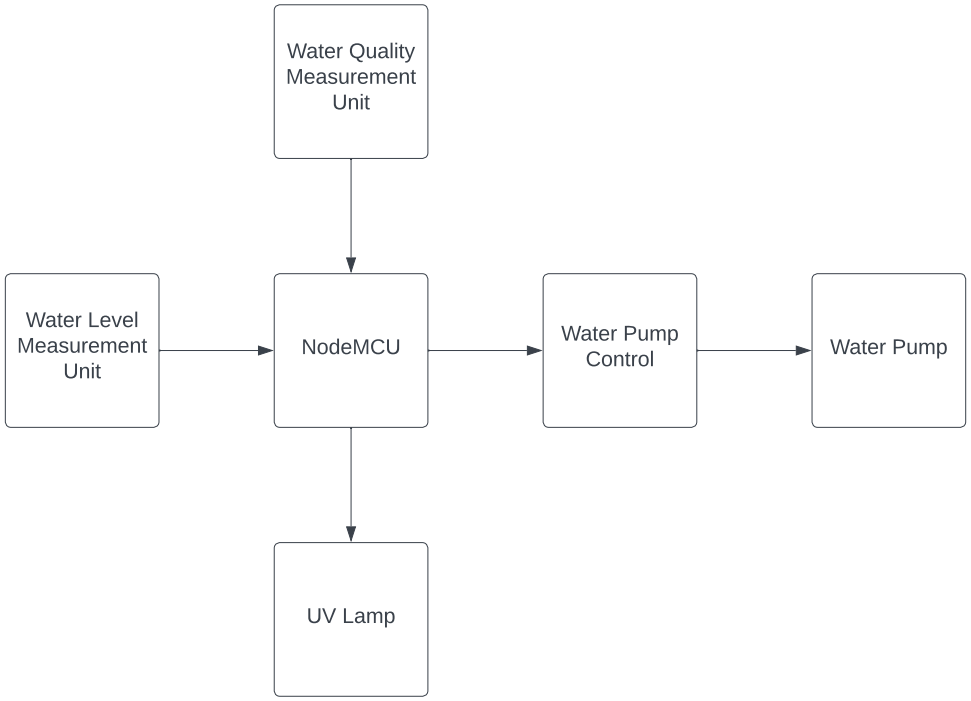


Figure 4.1: General Block Diagram of the Water Management System

4.1 Hardware

The Table 4.1 shows the list of hardware used for our system.

Table 4.1: List of Hardware Components

Sl. No	Item Description	Quantity
1	ESP 32 Micro-controller	1
2	Analog pH Sensor	1
3	Analog TDS Sensor	1
4	Turbidity sensor SKU:SEN0189	1
5	UV lamp	1
6	3W speaker	1
7	INMP441 I2S microphone module	1

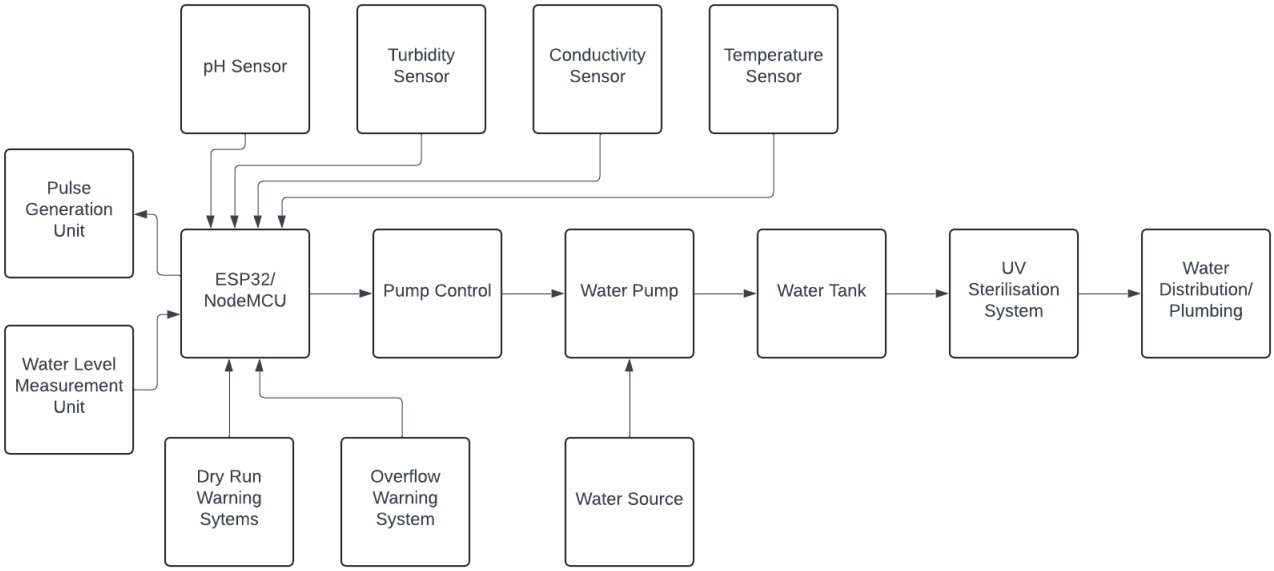


Figure 4.2: Block Diagram of the Hardware Section

4.1.1 ESP 32 Microcontroller

ESP32 is a single 2.4 GHz Wi-Fi-and-Bluetooth combo chip designed with the TSMC low-power 40 nm technology. It is designed to achieve the best power and RF performance, showing robustness, versatility and reliability in a wide variety of applications and power scenarios. An ESP32 will be used as a controller to read the water quality parameters and to publish this data to an custom app.

Specifications

- Single or Dual-Core 32-bit LX6 Microprocessor with clock frequency up to 240 MHz.
- 520 KB of SRAM, 448 KB of ROM and 16 KB of RTC SRAM.
- Supports 802.11 b/g/n Wi-Fi connectivity with speeds up to 150 Mbps.
- Support for both Classic Bluetooth v4.2 and BLE specifications.
- 34 Programmable GPIOs.
- Up to 18 channels of 12-bit SAR ADC and 2 channels of 8-bit DAC
- Serial Connectivity include 4 x SPI, 2 x I2C, 2 x I2S, 3 x UART.
- Ethernet MAC for physical LAN Communication (requires external PHY).
- 1 Host controller for SD/SDIO/MMC and 1 Slave controller for SDIO/SPI.
- Motor PWM and up to 16-channels of LED PWM.
- Secure Boot and Flash Encryption.
- Cryptographic Hardware Acceleration for AES, Hash (SHA-2), RSA, ECC and RNG.

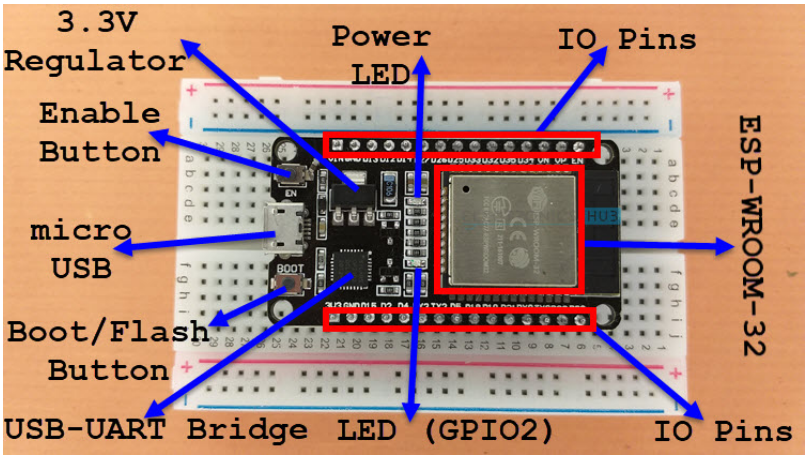


Figure 4.3: ESP 32 Pin-out

4.1.2 Sound Source

A sound source is used to generate the pulses which are to be reflected from the water surface. This is done using a pulse generator. The frequency is in the audible region to ensure that planar waves are generated. Planar waves are used as they propagate better inside the tube to be inserted into the water tank. The pulse signal is generated from the microcontroller and passed through an audio amplifier to drive the speakers. The speaker used is a 3W 4Ω speaker.

4.1.3 pH Sensors

The pH sensor is essential for assessing water acidity or alkalinity, typically within a pH range of 0 to 14. Comprising a data collector and an electrode probe, it measures the hydrogen ion concentration in water. The data collector processes signals from the probe using analog-to-digital converters, ensuring accuracy. The electrode probe, submerged in the water, generates a voltage proportional to pH, while a reference electrode stabilizes readings. Together, they provide precise pH measurements for various applications, such as drinking water monitoring and industrial processes. This enables proactive management to maintain water quality within acceptable pH ranges..

Figure 4.4: pH Sensor Kit

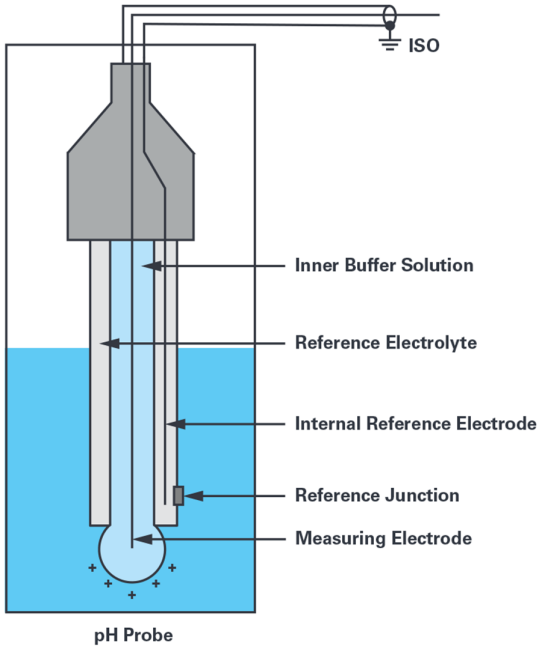


Figure 4.5: pH sensor Structure

4.1.4 TDS sensor

A TDS sensor, or Total Dissolved Solids sensor, measures the concentration of dissolved solids in a liquid. It typically works by passing a small electrical current through the liquid and measuring its conductivity. This conductivity is directly related to the concentration of dissolved ions in the liquid, including minerals, salts, and metals. TDS sensors are commonly used in water quality monitoring applications to assess the purity of water for drinking, industrial use, or environmental analysis.

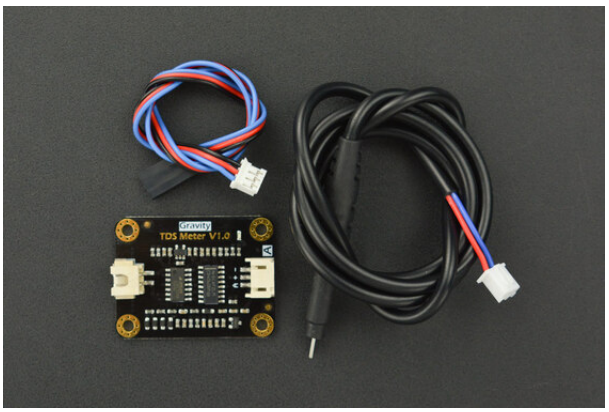


Figure 4.6: TDS sensor

4.1.5 Turbidity sensor

A turbidity sensor is a device designed to measure the cloudiness or haziness of a liquid caused by suspended particles. This sensor plays a crucial role in water quality monitoring and environmental analysis. Typically employed in applications such as wastewater treatment, drinking water quality assessment, and environmental research, turbidity sensors utilize light scattering or absorption techniques to quantify the concentration of particles in a liquid. Commonly measured in nephelometric turbidity units (NTU), the sensor emits light into the liquid, and the amount of scattered or absorbed light is detected and correlated to the turbidity level. The sensor's precision in discerning suspended solids in water makes it an essential tool in ensuring compliance with water quality standards, allowing for timely detection of changes in water clarity that may indicate pollution or other environmental concerns.

Figure 4.7: Turbidity sensor

4.1.6 UV Lamp

The UV lamp is used to purify the water once the turbidity has lowered to a particular value by allowing the water to settle for a particular duration of time. According to Class A purification standards, at least $40mJ/cm^3$ has to be illuminated into the water for properly killing bacteria and viruses.

Figure 4.8: Philips TUV T5 Lamp for disinfection

4.1.7 Level sensing unit

The level sensing unit incorporates a sound source that emits sound waves directed towards the surface of the water within the tank. These sound waves propagate through the water and are reflected back towards the source. A microphone positioned adjacent to the sound source captures the reflected waves. By measuring the time difference between the emission of the sound wave and the reception of its reflection, the unit calculates the distance traveled by the sound wave. Utilizing the known speed of sound, this distance is then converted into the level of water within the tank. This approach enables precise and real-time monitoring of water levels, providing valuable data for the automated water management system he equation for calculating the level of water in the tank using the time difference between the emission and reception of sound waves is:

The equation for calculating the level of water in the tank using the time difference between the emission and reception of sound waves is:

$$\text{Level} = \frac{c \times t}{2}$$

Where:

Level : the level of water in the tank (in meters),

c : the speed of sound in the medium (in meters per second),

t : the time difference between the emission and reception of the sound wave (in seconds).

This equation assumes that the sound wave travels at a constant speed c through the medium (in this case, water) and that the time difference t is accurately measured. The division by 2 accounts for the round-trip travel of the sound wave (to and from the surface of the water).

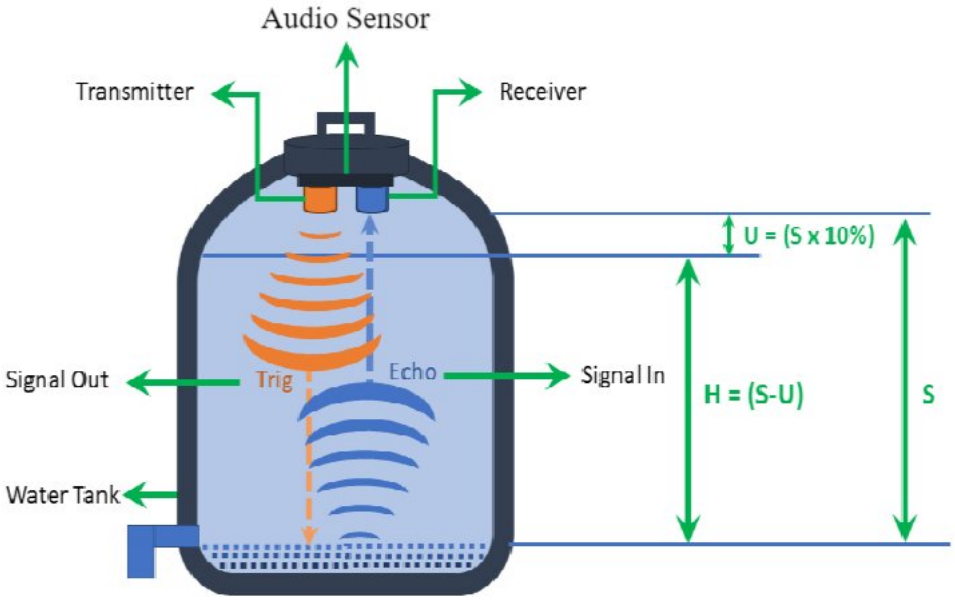


Figure 4.9: Level Sensing Unit

4.2 Software

4.2.1 App

The user interface design was made using a software called Figma. It includes a login page, a parameter reading interface, and an in-depth parameter analysis page. It also contains a score bar that will highlight if there is a significant variation in the parameters. The application will be made using a Software Development Kit (SDK) called Flutter and the database used is called Firebase. The parameter readings will be stored in Firebase which will then be displayed in the app.

A machine learning algorithm has been built and trained using a predefined dataset which predicts whether the water is potable or not depending on the parameter values. Each parameter is assigned a particular threshold and random forest is the algorithm used. Feature

extraction and scaling was also done on the dataset.

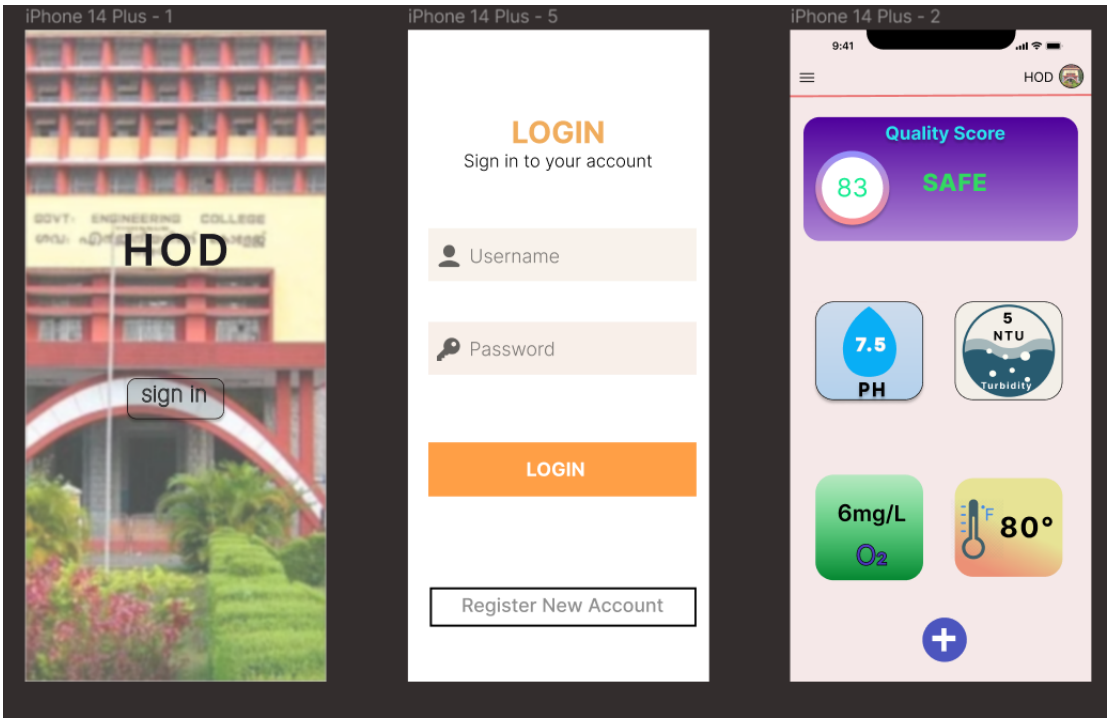


Figure 4.10: App UI

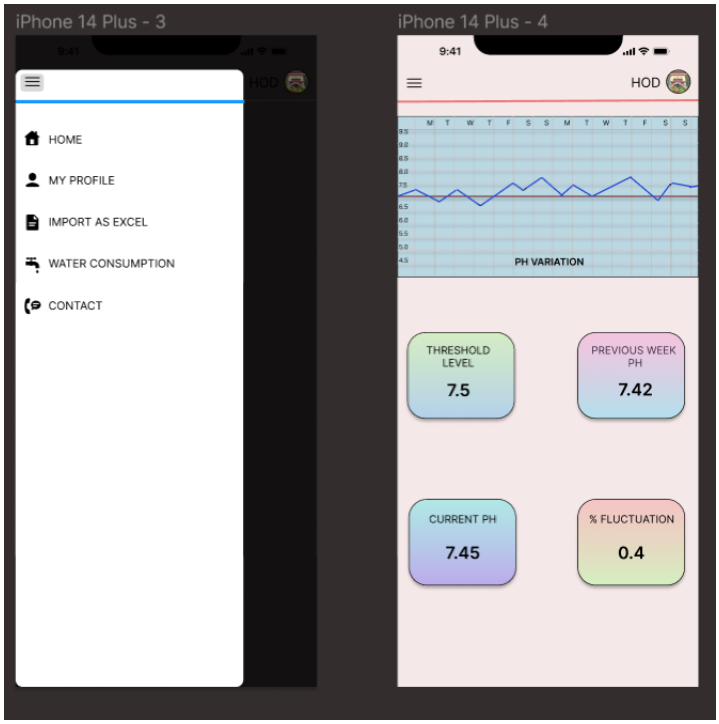


Figure 4.11: App UI2

Arduino IDE is used to program the ESP32. As Firebase is the database used, it is connected with the Arduino IDE using the ”Firebase ESP8266” library, as this library is

compatible with ESP32 as well. A real-time database or Firestore is set up and the database credentials including the Project ID, Web API Key and database URL are included in the ESP32 Arduino code. The Firebase configuration file is then downloaded to be added to the Flutter app as a YAML file. We then used HTTP or WebSocket communication to connect the Flutter app to the ESP32. It is also possible to connect them via Bluetooth using inbuilt Flutter packages like ”flutterblue”.To introduce a new layer of security, Firebase security authentication can also be added so that only the department HOD will be able to access the database.

The sensors provide the readings to ESP32. ESP32, which has been connected to the Firebase cloud via wifi, uploads the sensor readings into the database. The app will thus display the parameter readings from the database.

4.2.2 Webpage

This part of the project focuses on water quality analysis using machine learning techniques, specifically predicting water potability. Key features such as pH, solids, conductivity and turbidity are explored through data exploration to understand their distributions and characteristics. The software provides functionalities for data preparation, cleaning, exploratory data analysis, and the development of machine learning models for potability prediction. With a user-friendly interface, users can input water quality data and visualize potability predictions effortlessly. The predictive model, trained on diverse water quality parameters, contributes to ensuring access to clean and potable water—a fundamental aspect of human well-being.

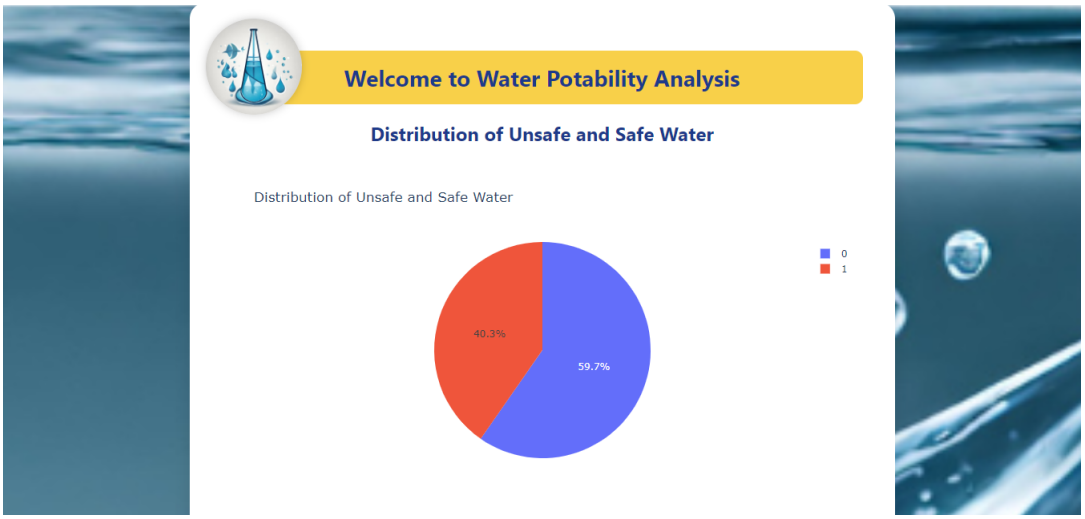


Figure 4.12: Webpage 1

Figure 4.13: Webpage 2

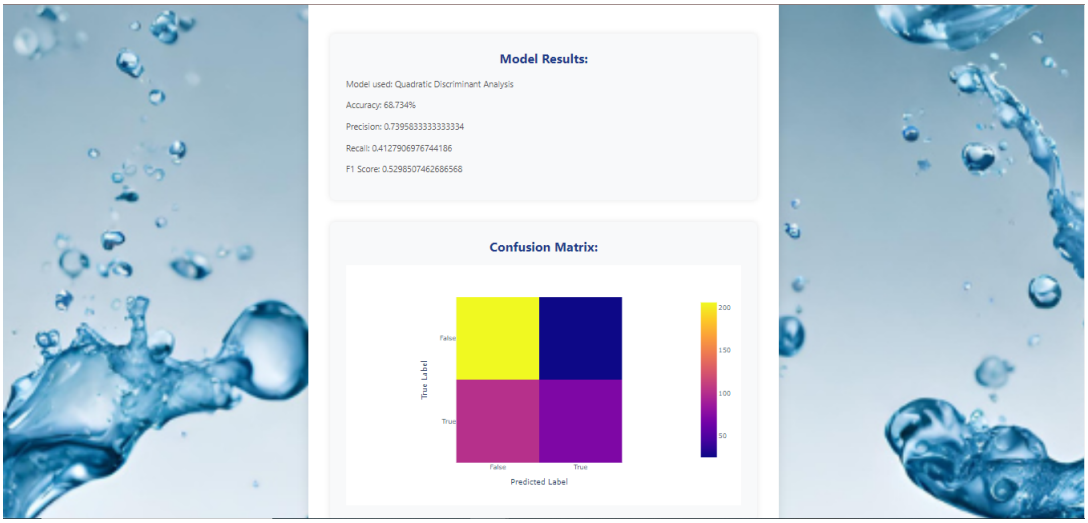


Figure 4.14: Webpage 3

4.2.3 Block diagram of software architecture

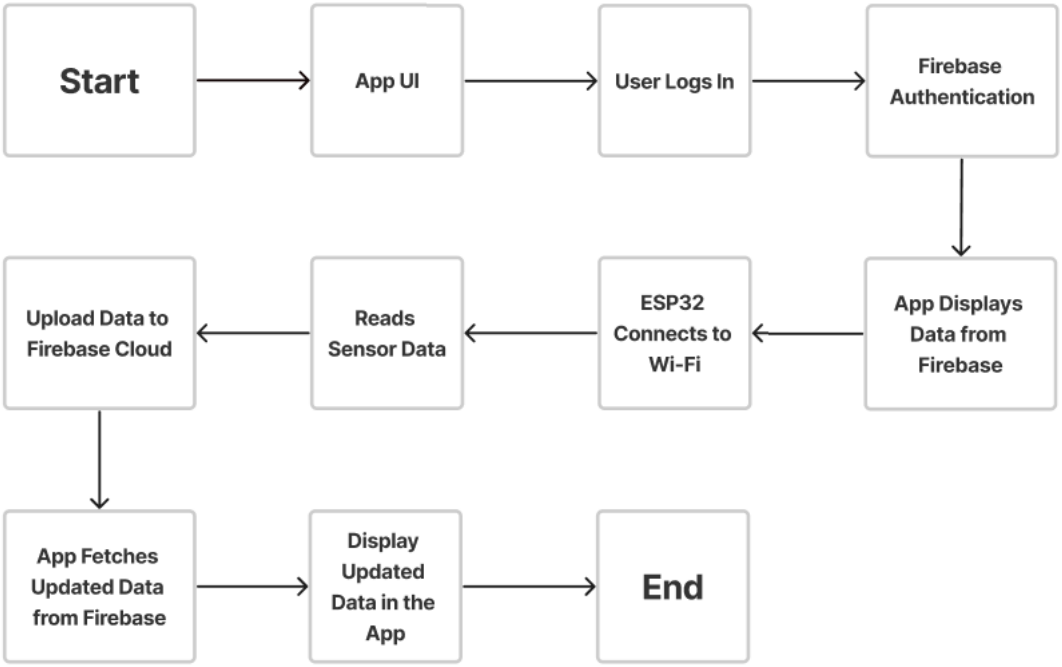


Figure 4.15: Block diagram for the System

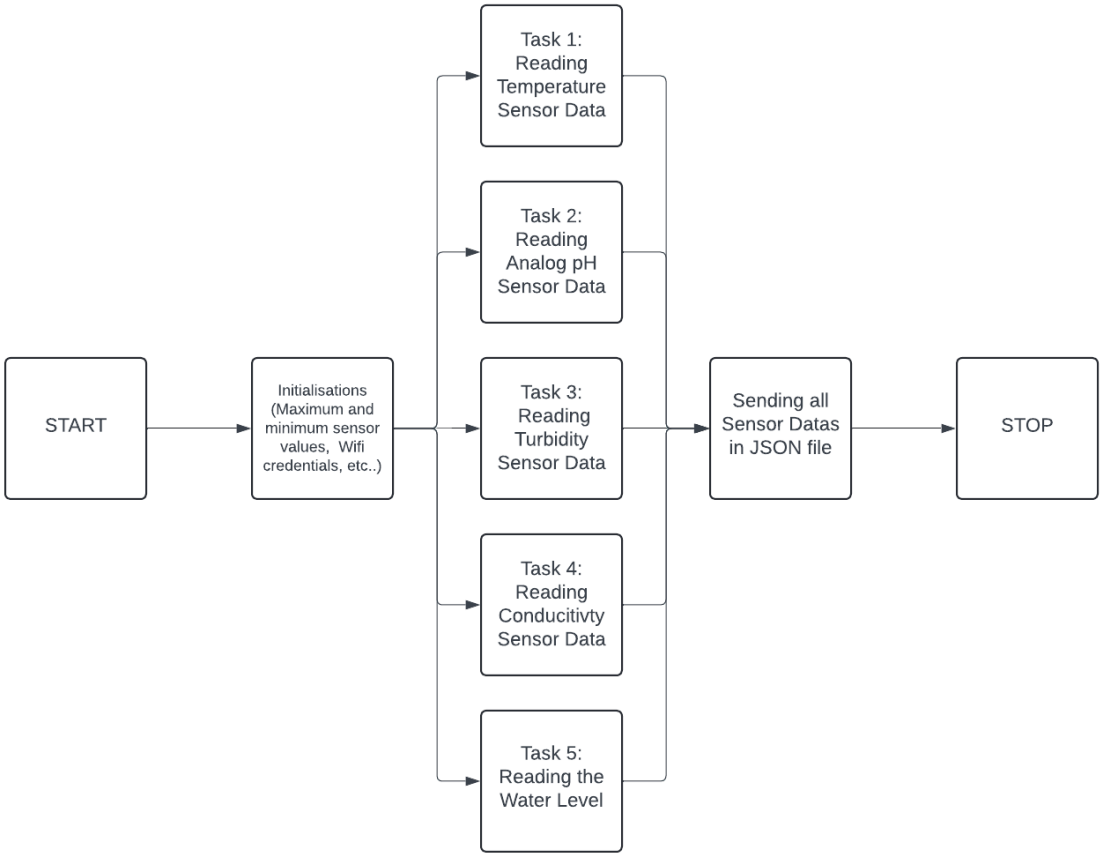


Figure 4.16: Program Flow chart for ESP32

4.2.4 Code Snippets

The below code is used to control sensors with ESP32 and send data to firebase.

```
#include <Arduino.h>
#include <WiFi.h>
#include <Firebase_ESP_Client.h>
#include <DHT.h>
#include "DFRobot_ESP_PH.h"
#include "EEPROM.h"

// Define sensor pins and constants
#define PH_PIN 35
#define TdsSensorPin 34
#define DHTPIN 2
#define trigPin 25
#define echoPin 26
#define relayPin 12
#define turbiditySensorPin 32

// Define sensor types and instances
DFRobot_ESP_PH ph;
DHT dht(DHTPIN, DHTTYPE);

// Define Firebase parameters
#define WIFI_SSID "Redmi 9 Prime"
#define WIFI_PASSWORD "******"
#define API_KEY "AIzaSyACL3Y-1P_48fA7qMTwrNMZ9BB_yeuwMTE"
#define USER_EMAIL "hod@gmail.com"
#define USER_PASSWORD "654321"
#define DATABASE_URL
↪  "https://test-cli-c7c44-default-rtdb.asia-southeast1.firebasedatabase.app/"
FirebaseData fbdo;
FirebaseAuth auth;
```

```
30 FirebaseConfig config;
31
32 // Define variables
33 float temperature = 25, humidity, conductivity, pH;
34 float turbidity = 0.0;
35 float turbiditySensorValue = 0.0;
36 float maxTurbidity = 0.0, minTurbidity = 1000.0;
37 float maxHumidity = 0.0, minHumidity = 100.0;
38 float maxTemperature = 0.0, minTemperature = 100.0;
39 float maxConductivity = 0.0, minConductivity = 100.0;
40 float maxpH = 0.0, minpH = 14.0;
41 float ppm = 0.0;
42
43 // Define Firebase paths
44 String tempPath = "/temperature";
45 String maxTempPath = "/max temperature";
46 String minTempPath = "/min temperature";
47 String humPath = "/humidity";
48 String maxHumPath = "/max humidity";
49 String minHumPath = "/min humidity";
50 String timePath = "/timestamp";
51 String conductivityPath = "/conductivity";
52 String maxConductivityPath = "/max conductivity";
53 String minConductivityPath = "/min conductivity";
54 String turbidityPath = "/turbidity";
55 String maxTurbidityPath = "/max turbidity";
56 String minTurbidityPath = "/min turbidity";
57 String pHPath = "/pH";
58 String maxpHPath = "/max pH";
59 String minpHPath = "/min pH";
60 String ppmPath = "/ppm";
61
62 // Initialize WiFi connection
```

```
63 void initWiFi() {
64   WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
65   Serial.print("Connecting to WiFi ..");
66   while (WiFi.status() != WL_CONNECTED) {
67     Serial.print('.');
68     delay(1000);
69   }
70   Serial.println(WiFi.localIP());
71   Serial.println();
72   Serial.begin(115200);
73   EEPROM.begin(32);
74   ph.begin();
75 }
76 
77 // Get current time
78 unsigned long getTime() {
79   time_t now;
80   struct tm timeinfo;
81   delay(100);
82   if (!getLocalTime(&timeinfo)) {
83     return(0);
84   }
85   time(&now);
86   return now;
87 }
88 
89 // Task for DHT sensor readings
90 void dhtSensor(void * parameters) {
91   for (;;) {
92     temperature = dht.readTemperature();
93     vTaskDelay(1000/portTICK_PERIOD_MS);
94   }
95 }
```

```
96
97  // Task for pH sensor readings
98  void pHSensor(void * parameters) {
99    for (;;) {
100     float voltage;
101     static unsigned long timepoint = millis();
102     if (millis() - timepoint > 1000U) {
103       timepoint = millis();
104       voltage = analogRead(PH_PIN) / ESPADC * ESPVOLTAGE;
105       temperature = readTemperature();
106       pH = ph.readPH(voltage, temperature);
107     }
108     vTaskDelay(1000/portTICK_PERIOD_MS);
109   }
110 }
111
112 // Task for turbidity sensor readings
113 void turbiditySensor(void * parameters) {
114   for (;;) {
115     turbiditySensorValue = analogRead(turbiditySensorPin);
116     turbidity = map(turbiditySensorValue, 0, 4095, 100, 0);
117     vTaskDelay(1000/portTICK_PERIOD_MS);
118   }
119 }
120
121 // Task for conductivity sensor readings
122 void conductivitySensor(void * parameters) {
123   float VREFF = 3.3;
124   int sensorValue;
125   float tdsValue = 0;
126   float temperature1 = 25;
127   for (;;) {
128     sensorValue = analogRead(TdsSensorPin);
```

```
129     float sensorVoltage = sensorValue * VREFF / 4096;
130     float compensationCoefficient = 1.0 + 0.02 * (temperature1 - 25.0);
131     float compensationVoltage = sensorVoltage / compensationCoefficient;
132     tdsValue = (133.42 * compensationVoltage * compensationVoltage *
        ↪  compensationVoltage - 255.86 * compensationVoltage *
        ↪  compensationVoltage + 857.39 * compensationVoltage) * 0.5;
133     ppm = tdsValue;
134     conductivity = ppm / 0.7;
135     vTaskDelay(1000/portTICK_PERIOD_MS);
136   }
137 }
138
139 void setup() {
140   pinMode(trigPin, OUTPUT);
141   pinMode(echoPin, INPUT);
142   pinMode(relayPin, OUTPUT);
143   Serial.begin(115200);
144   initWiFi();
145   configTime(0, 0, ntpServer);
146   config.api_key = API_KEY; // Assign the api key (required)
147   auth.user.email = USER_EMAIL;// Assign the user sign in credentials
148   auth.user.password = USER_PASSWORD;// Assign the user sign in credentials
149   config.database_url = DATABASE_URL;// Assign the RTDB URL (required)
150   if (Firebase.signUp(&config, &auth, "", "")) {
151     Serial.println("ok");
152   } else {
153     Serial.printf("%s\n", config.signer.signupError.message.c_str());
154   }
155   Firebase.reconnectWiFi(true);
156   fbdo.setResponseSize(4096);
157   Firebase.begin(&config, &auth);
158   String databasePath = "/UsersData/readings";
159   xTaskCreate(dhtSensor, "DHTSENSOR", 1000, NULL, 1, NULL);
```

```
160   xTaskCreate(pHSensor, "PHSENSOR", 1000, NULL, 1, NULL);
161   xTaskCreate(turbiditySensor, "TURBIDITYSENSOR", 1000, NULL, 1, NULL);
162   xTaskCreate(conductivitySensor, "CONDUCTIVITYSENSOR", 1000, NULL, 1, NULL);
163 }
164
165 void loop() {
166   digitalWrite(relayPin, LOW);
167   digitalWrite(trigPin, LOW);
168   delayMicroseconds(2);
169   digitalWrite(trigPin, HIGH);
170   delayMicroseconds(10);
171   digitalWrite(trigPin, LOW);
172   long duration = pulseIn(echoPin, HIGH);
173   float distanceCm = duration * SOUND_SPEED / 2;
174   float distanceInch = distanceCm * CM_TO_INCH;
175   Serial.print("Distance (cm): ");
176   Serial.println(distanceCm);
177   Serial.print("Distance (inch): ");
178   Serial.println(distanceInch);
179   delay(1000);
180
181   // Control relay based on distance
182   while (distanceCm >= 18) {
183     digitalWrite(relayPin, HIGH);
184   }
185
186   // Firebase data sending and cleanup
187   if (Firebase.ready() && (millis() - sendDataPrevMillis > timerDelay ||
      ↪  sendDataPrevMillis == 0)) {
188     sendDataPrevMillis = millis();
189     int timestamp = getTime();
190     int timestampPrev = timestamp - deleteDelay;
191     Serial.print("Time: ");
```

```
192     Serial.println(timestamp);
193     String parentPath = databasePath + "/" + String(timestamp);
194     temperature = dht.readTemperature();
195     humidity = dht.readHumidity();
196     FirebaseJson json;
197     json.set(tempPath.c_str(), String(temperature));
198     json.set(humPath.c_str(), String(humidity));
199     if (maxTemperature < temperature) maxTemperature = temperature;
200     if (minTemperature > temperature) minTemperature = temperature;
201     json.set(maxHumPath.c_str(), String(maxHumidity));
202     json.set(minHumPath.c_str(), String(minHumidity));
203     json.set(maxTempPath.c_str(), String(maxTemperature));
204     json.set(minTempPath.c_str(), String(minTemperature));
205     json.set(conductivityPath.c_str(), String(conductivity));
206     if (maxConductivity < conductivity) maxConductivity = conductivity;
207     if (minConductivity > conductivity) minConductivity = conductivity;
208     json.set(maxConductivityPath.c_str(), String(maxConductivity));
209     json.set(minConductivityPath.c_str(), String(minConductivity));
210     json.set(turbidityPath.c_str(), String(turbidity));
211     if (maxTurbidity < turbidity) maxTurbidity = turbidity;
212     if (minTurbidity > turbidity) minTurbidity = turbidity;
213     json.set(maxTurbidityPath.c_str(), String(maxTurbidity));
214     json.set(minTurbidityPath.c_str(), String(minTurbidity));
215     json.set(pHPath.c_str(), String(pH));
216     if (maxpH < pH) maxpH = pH;
217     if (minpH > pH) minpH = pH;
218     json.set(maxpHPath.c_str(), String(maxpH));
219     json.set(minpHPath.c_str(), String(minpH));
220     json.set(ppmPath.c_str(), String(ppm));
221     json.set(timePath, String(timestamp));
222     Serial.printf("Set json... %s\n", Firebase.RTDB.setJSON(&fbdo,
      ↪  parentPath.c_str(), &json) ? "ok" : fbdo.errorReason().c_str());
223     String oldDataPath = databasePath + "/" + String(timestampPrev);
```

```
224     Serial.printf("Deleting old data at path: %s\n", oldDataPath.c_str());
225     if (Firebase.RTDB.deleteNode(&fbdo, oldDataPath.c_str())) {
226       Serial.println("Old data deleted successfully");
227     } else {
228       Serial.printf("Failed to delete old data: %s\n",
        ↪  fbdo.errorReason().c_str());
229     }
230     vTaskDelay(1000/portTICK_PERIOD_MS);
231   }
232 }
233
```

The below code is for initialisation of the app and Firebase

```dart
import 'package:firebaseproject/ui/splash_screen.dart';
import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart';
import 'firebase_options.dart';

void main() async{
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});
  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      title:'Water Management',
      home: LoginScreen(),
    );
  }
}

```

The below code is for Firebase authentication and login

```dart
import 'dart:async';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebaseproject/ui/auth/signup_screen.dart';
import 'package:firebaseproject/ui/project_ui/home_screen2.dart';
import 'package:firebaseproject/utils/utils.dart';
```

```dart
import 'package:firebaseproject/widgets/round_button.dart';
import 'package:flutter/material.dart';

class LoginScreen extends StatefulWidget {
  const LoginScreen({super.key});
  @override
  State<LoginScreen> createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  final _formKey=GlobalKey<FormState>();
  final emailController=TextEditingController();
  final passwordController=TextEditingController();
  final _auth=FirebaseAuth.instance;
  bool loading=false;

  @override
  void dispose() {
    // TODO: implement dispose
    super.dispose();
    emailController.dispose();            //dispose when screen is not there
    passwordController.dispose();
  }

  void login(){
    setState(() {
      loading=true;
    });
    _auth.signInWithEmailAndPassword(email: emailController.text.toString(),
    ↪ password: passwordController.text.toString()).then((value) {
          Utils().toastMessage(value.user!.email.toString());  //to display
          ↪ if successful
```

```
36            Navigator.push(context,MaterialPageRoute(builder:
              ↪  (context)=>HomeScreenTwo()));
37            setState(() {
38              loading=false;
39            });
40      }).onError((error, stackTrace){
41        debugPrint(error.toString());
42        Utils().toastMessage(error.toString());
43        setState(() {
44          loading=false;
45        });
46      });
47    }
48    @override
49    Widget build(BuildContext context) {
50      return Scaffold(
51        appBar: AppBar(
52          automaticallyImplyLeading: false,           //to remove back button
53          backgroundColor: Colors.deepPurple,
54          title: const Padding(padding:EdgeInsets.only(bottom:8),child:
            ↪  Center(child: Text("Login",style: TextStyle(color:
            ↪  Colors.white),))),
55        ),
56        body:Padding(
57          padding: const EdgeInsets.symmetric(horizontal: 20),
58          child: Column(
59            mainAxisAlignment: MainAxisAlignment.center,
60            crossAxisAlignment: CrossAxisAlignment.center,
61            children: [
62              Form(
63                key:_formKey,
64                child:Column(
65                  children: [
```

```
                    TextFormField(
                      keyboardType: TextInputType.emailAddress,
                      controller: emailController,
                      decoration: const InputDecoration(
                        hintText: 'Email',
                        prefixIcon: Icon(Icons.alternate_email),
                      ),
                      validator: (value){
                            if(value!.isEmpty) return 'Enter email';
                            return null;
                      },   //if user has not entered anything and is trying to
                      ↪  login
                    ),

                    const SizedBox(height:10),
                    TextFormField(
                      controller: passwordController,
                      obscureText: true,              //to hide the password
                      decoration: const InputDecoration(
                        hintText: 'Password',
                        prefixIcon: Icon(Icons.lock_open),
                      ),

                      validator: (value){
                        if(value!.isEmpty) return 'Enter password';
                        return null;
                      },),],)
                ),

                 const SizedBox(height: 50,),
                 RoundButton(title: 'Login',loading:loading,onTap: (){
                             if(_formKey.currentState!.validate()){
                                         login();
```

```
                            }
                        },),
                const SizedBox(height: 30,),

                Row(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: [
                    const Text("dont have an account?"),
                    TextButton(onPressed: (){
                            Navigator.push(context,MaterialPageRoute(builder:
                            ↪  (context)=>SignUpScreen()));
                    },
                        child: const Text('Sign up')
                    ),],),],),));
  }
}
```

The below code is the app implementation with UI and data logging

```
import 'package:firebase_database/firebase_database.dart';
import 'package:firebaseproject/Contact/contactUs.dart';
import 'package:firebaseproject/ui/auth/login_screen.dart';
import 'package:firebaseproject/ui/project_ui/home_screen2.dart';
import 'package:firebaseproject/widgets/round_button.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;
import 'dart:convert';

class HomeScreenTwo extends StatefulWidget {
  const HomeScreenTwo({super.key});
  @override
  State<HomeScreenTwo> createState() => _HomeScreenTwoState();
}
```

```dart

class _HomeScreenTwoState extends State<HomeScreenTwo> {
  String prediction3="dd";
  final ref=FirebaseDatabase.instance.ref("UsersData/readings");

  void _showDialog(String problem){
    showDialog(context: context, builder: (context){
       return CupertinoAlertDialog(
         title: Text(problem),
         content:const Text("Please go to Contacts Section to solve the problem"),
         actions: [
           MaterialButton(onPressed: (){Navigator.pop(context);},child:Text('Go Back')),
           MaterialButton(onPressed: (){Navigator.push(context,MaterialPageRoute(builder: (context)=>Contact()));},child:Text('Go to Contacts')),
         ],
       );});
  }

  Future<String> predictWaterPotability(Map<dynamic, dynamic> data) async {
    final response = await http.post(
      Uri.parse('https://8b01-42-108-125-60.ngrok-free.app/predict'),
      headers: <String, String>{
        'Content-Type': 'application/json; charset=UTF-8',
      },
      body: jsonEncode(data),
    );
    if (response.statusCode == 200) {
      return jsonDecode(response.body)['prediction'];
    } else {
      throw Exception('Failed to predict water potability.');
```

```
45      }
46    }
47
48    @override
49    Widget build(BuildContext context) {
50      return Scaffold(
51      appBar: AppBar(
52            title:const Text("Water parameters",style: TextStyle(color:
          ↪  Colors.white),),
53            backgroundColor: Colors.deepPurple,
54          ),
55          drawer: Drawer(
56            child:ListView(
57              padding: const EdgeInsets.only(top:0),
58              children: [
59                const UserAccountsDrawerHeader(
60                    decoration: BoxDecoration(
61                      color:Color(0xff764abc),
62                    ),
63                    accountName: Text('Head of Department'),
64                    accountEmail: Text('hod@gmail.com')
65                ),
66                ListTile(
67                  leading:const Icon(Icons.home),
68                  title:const Text('Home'),
69                  onTap: (){
70                    Navigator.push(context,MaterialPageRoute(builder:
                    ↪  (context)=>HomeScreenTwo()));
71                  },),
72                ListTile(
73                  leading: const Icon(Icons.logout),
74                  title:const Text('Logout'),
75                  onTap: (){
```

```
                Navigator.push(context,MaterialPageRoute(builder:
                ↪  (context)=>LoginScreen()));
              },),
            ListTile(
              leading:const Icon(Icons.call),
              title:const Text('Contact'),
              onTap: (){
                Navigator.push(context,MaterialPageRoute(builder:
                ↪  (context)=>Contact()));
              },),],),
      ),

  body: SafeArea(
  child:Column(
  children: [
  Expanded(
  child: SingleChildScrollView(
  child: StreamBuilder(
      stream:ref.onValue,
      builder: (context,snapshot){
      if(!snapshot.hasData){return const Text("loading");}
      else{
          Map<dynamic,dynamic> map=snapshot.data!.snapshot.value as dynamic;
          List<dynamic> list=[];
          list.clear();
          list=map.values.toList();
          list.sort((a, b) {
          if (a.containsKey('timestamp') && b.containsKey('timestamp')) {
              return a['timestamp'].compareTo(b['timestamp']);
          } else {return 1;}
          });

          int length1=snapshot.data!.snapshot.children.length;
```

```
107             double pH=double.parse(list[length1-1]['pH']);
108             double conductivity=double.parse(list[length1-1]['conductivity']);
109             double turbidity=double.parse(list[length1-1]['turbidity']);
110             double tds=double.parse(list[length1-1]['ppm']);
111
112             void trial() async{
113             String prediction2=await predictWaterPotability(
114                 {
115                 "ph":double.parse(list[length1-1]["pH"]),
116                 "Hardness": 150.0,
117                 "Solids":double.parse(list[length1-1]["ppm"]),
118                 "Chloramines":7.3,
119                 "Sulfate":368.5,
120                 "Conductivity":double.parse(list[length1-1]["conductivity"]),
121                 "Organic_carbon":10.379,
122                 "Trihalomethanes":86.99,
123                 "Turbidity":double.parse(list[length1-1]["turbidity"]),
124                 });
125                 prediction3=prediction2;
126             }
127             trial();
128
129             if(prediction3=="Potable") return Center(child: Container(child:
                ↪  Center(child: Text("Water is portable",)),color:
                ↪  Colors.greenAccent,height: 200,width:100));
130
131             if(pH>9.0  pH<5.0) {
132             return CupertinoAlertDialog(
133                 title: Text("PH out of range: $pH"),
134                 content: const Text("Threshold: 5-9"),
135                 actions: [
136                 MaterialButton(onPressed: () {
137                     Navigator.pop(context);
```

```
138                 }, child: const Text("Go Back")),
139                 MaterialButton(onPressed: () {
140                     Navigator.push(context, MaterialPageRoute(
141                         builder: (context) => Contact()));
142                 }, child: const Text("Contact"),)
143                 ],);
144             }
145             if(turbidity>6.0  turbidity<0.0) {
146             return CupertinoAlertDialog(
147                 title: Text("Turbidity out of range: $turbidity"),
148                 content: const Text("Threshold: 0-5"),
149                 actions: [
150                 MaterialButton(onPressed: () {
151                      Navigator.pop(context);
152                 }, child: const Text("Go Back")),
153                 MaterialButton(onPressed: () {
154                     Navigator.push(context, MaterialPageRoute(
155                         builder: (context) => Contact()));
156                 }, child: const Text("Contact"),)
157                 ],);
158             }
159             if(conductivity>1000.0  conductivity<200.0) {
160             return CupertinoAlertDialog(
161                 title: Text("Conductivity out of range: $conductivity"),
162                 content: const Text("Threshold: 1-5"),
163                 actions: [
164                 MaterialButton(onPressed: () {
165                     Navigator.pop(context);
166                 }, child: const Text("Go Back")),
167                 MaterialButton(onPressed: () {
168                     Navigator.push(context, MaterialPageRoute(
169                         builder: (context) => Contact()));
170                 }, child: const Text("Contact"),)
```

```
171                 ],);
172             }
173             if(tds>350.0  tds<50.0) {
174             return CupertinoAlertDialog(
175                 title:  Text("TDS out of range: $tds"),
176                 content: const Text("Threshold: 0-150"),
177                 actions: [
178                 MaterialButton(onPressed: () {
179                     Navigator.pop(context);
180                 }, child: const Text("Go Back")),
181                 MaterialButton(onPressed: () {
182                      Navigator.push(context, MaterialPageRoute(
183                         builder: (context) => Contact()));
184                 }, child: const Text("Contact"),)
185                 ],);
186             }
187             return Column(
188                 mainAxisAlignment: MainAxisAlignment.center,
189                 children: [
190                 const SizedBox(height: 100),
191                 Row(
192                     children: [
193                     Padding(padding:const EdgeInsets.only(left:
                    ↪  40,right:30,bottom: 40,top:20),child:
                    ↪  Contain(message:list[length1-1]['humidity'] ,
                    ↪  color:HexColor('#90EE90'),title:
                    ↪  "Humidity",max:list[length1-1]['max humidity'])),
194                     Padding(padding:const EdgeInsets.only(left:
                    ↪  40,right:30,bottom: 40,top:20),child: Contain(message:
                    ↪  list[length1-1]['temperature'],
                    ↪  color:HexColor('F4F82D'),title:"Temperature",
                          max:list[length1-1]['max temperature'])),
195                 ],),
196
```

```
                Row(
                    children: [
                    Padding(padding:const EdgeInsets.only(left:
                    ↪ 40,right:30,bottom: 40),child:
                    ↪ Contain(message:list[length1-1]['pH'] ,
                    ↪ color:HexColor('ffdab9'),title:
                    ↪ "PH",max:list[length1-1]['max pH'])),
                    Padding(padding:const EdgeInsets.only(left:
                    ↪ 40,right:30,bottom: 40),child:
                    ↪ Contain(message:list[length1-1]['turbidity'] ,
                    ↪ color:HexColor('9FDAED'),title:
                    ↪ "Turbidity",max:list[length1-1]['max turbidity'])),
                    ],),
                Row(
                    children: [
                    Padding(padding:const EdgeInsets.only(left:
                    ↪ 40,right:30,bottom:40),child:
                    ↪ Contain(message:list[length1-1]['conductivity'] ,
                    ↪ color:HexColor('974DF6'),title:
                    ↪ "Conductivity",max:list[length1-1]['max
                    ↪ conductivity'])),
                    Padding(padding:const EdgeInsets.only(left:
                    ↪ 40,right:30,bottom:40),child:
                    ↪ Contain(message:list[length1-1]['ppm'] ,
                    ↪ color:HexColor('FB73CD'),title: "TDS")),
                    ],),]
            );}}
    ),),),],),));
  }
```

Below is code for implementing Machine learing algorithm and webpage.

```
from flask import Flask, render_template, request
import pandas as pd
import plotly.express as px
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score,
↪  recall_score, f1_score, ConfusionMatrixDisplay
from flask import  jsonify

from sklearn.preprocessing import StandardScaler
from collections import OrderedDict

app = Flask(__name__)

# Load the data
data = pd.read_csv("app\static\water_potability.csv")
data = data.dropna()

# Prepare the data for dropdown options
dropdown_options = {
    'ph': 'PH',
    'Conductivity': 'Conductivity',
    'Turbidity': 'Turbidity'
}

# Train the model
X = data.drop("Potability", axis=1)
y = data["Potability"]
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
↪  random_state=42)
qda_model = QuadraticDiscriminantAnalysis()
qda_model.fit(X_train, y_train)


plt.figure(figsize=(15, 10))
sns.histplot(data['Potability'])
plt.title("Distribution of Unsafe and Safe Water")


@app.route('/', methods=['GET', 'POST'])
def index():
    potability_counts = data['Potability'].value_counts()
    fig = px.pie(names=potability_counts.index,
    ↪  values=potability_counts.values, title="Distribution of Unsafe and Safe
    ↪  Water")
    static_plot = fig.to_html(full_html=False)
    # Generate histogram for the selected feature
    selected_feature = request.form.get('feature',
    ↪  list(dropdown_options.keys())[0])
    fig = px.histogram(data, x=selected_feature, color='Potability',
    ↪  title=f"Distribution of {dropdown_options[selected_feature]}")
    histogram_plot = fig.to_html(full_html=False, default_height=500,
    ↪  default_width=700)

    # Make predictions on the testing set
    y_pred = qda_model.predict(X_test)

    # Calculate metrics
    accuracy = round(accuracy_score(y_test, y_pred)*100,3)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
```

```
    f1 = f1_score(y_test, y_pred)
    conf_matrix = confusion_matrix(y_test, y_pred)


    colorscale =   [0.25, '#ffcc00'],  # Dark yellow
    [0.5, '#800080'],   # Purple
    [0.75, '#00cccc'],  # Cyan
    [1, '#008000']     # Green
    # Convert confusion matrix to HTML table
    conf_matrix_fig = px.imshow(conf_matrix, labels=dict(x="Predicted Label",
    ↪  y="True Label"),x=['False', 'True'], y=['False', 'True'],
    color_continuous_scale=colorscale)
    conf_matrix_plot = conf_matrix_fig.to_html(full_html=False)


    return render_template('index.html',
    ↪  static_plot=static_plot,dropdown_options=dropdown_options,
    ↪  histogram_plot=histogram_plot, selected_feature=selected_feature,
    ↪  accuracy=accuracy, precision=precision, recall=recall, f1=f1,
    ↪  conf_matrix=conf_matrix_plot)


# Function to preprocess incoming data
def preprocess_data(data):
    # Create an ordered dictionary with constant values for features other than
    ↪  conductivity, pH, and turbidity

    column_order = ['ph', 'Hardness', 'Solids', 'Chloramines', 'Sulfate',
    ↪  'Conductivity', 'Organic_carbon', 'Trihalomethanes', 'Turbidity']
    # Combine constant values with incoming data


    # Convert combined data to DataFrame
```

```
    df = pd.DataFrame(data,columns=column_order, index=[0])
    df.fillna(method='ffill', inplace=True)
    # Scale numerical features
    scaler = StandardScaler()
    numerical_features = ['Conductivity', 'ph', 'Turbidity']  # Only scale
    ↪ these features
    df[numerical_features] = scaler.fit_transform(df[numerical_features])

    return df

# Flask route to accept new data from Flutter app
@app.route('/predict', methods=['POST'])
def predict():
    # Get the JSON data sent from the Flutter app
    request_data = request.json
    print(request_data)
    # Preprocess the incoming data
    input_data = preprocess_data(request_data)

    # Make predictions
    prediction = qda_model.predict(input_data)
    print(prediction)
    # Convert prediction to a human-readable format (if needed)
    prediction_label = "Potable" if prediction[0] == 1 else "Not Potable"

    # Return the prediction as JSON
    return jsonify({"prediction": prediction_label})

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True)
```

Below is the HTML code for Webpage.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Water Quality Analysis</title>
    <style>
        body {
            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
            margin: 0;
            padding: 0;
            background-image: url("static\\images\\background1.png");
            background-size: cover;
            background-color: #f0f4f7; /* Fallback color */
        }
        .container {
            max-width: 800px;
            margin: 0 auto;
            padding: 40px;
            position: relative;
            background-color: #ffffff;
            border-radius: 20px;
            box-shadow: 0 0 20px rgba(0, 0, 0, 0.1);
            animation: fadeIn 0.5s ease-in-out;
        }
        .logo {
            position: absolute;
            top: 20px;
            left: 20px;
            width: 120px;
            height: auto;
            border-radius: 50%;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
```

```
            animation: slideInLeft 0.5s ease-in-out;
        }
        h1, h2 {
            color: #1e3a8a;
            text-align: center;
            margin-top: 20px;
            animation: slideInUp 0.5s ease-in-out;
        }
        h1 {
            font-size: 28px;
            background-color: #f8d049;
            padding: 15px 0;
            border-radius: 10px;
        }
        h2 {
            font-size: 24px;
            margin-bottom: 20px;
        }
        .plot {
            margin-bottom: 40px;
            animation: slideInUp 0.5s ease-in-out;
        }
        form {
            margin-bottom: 40px;
            text-align: center;
            animation: slideInUp 0.5s ease-in-out;
        }
        label {
            font-weight: bold;
            color: #1e3a8a;
            margin-right: 10px;
        }
        select {
```

```
            padding: 10px;
            border: 1px solid #1e3a8a;
            border-radius: 5px;
            background-color: #fff;
            color: #1e3a8a;
            margin-right: 10px;
        }
        button {
            padding: 10px 25px;
            background-color: #1e3a8a;
            color: #fff;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            transition: background-color 0.3s ease-in-out;
        }
        button:hover {
            background-color: #0c2353;
        }
        .model-results, .confusion-matrix {
            background-color: #f8f9fa;
            padding: 30px;
            border-radius: 10px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            margin-bottom: 40px;
            animation: slideInUp 0.5s ease-in-out;
        }
        .model-results h2, .confusion-matrix h2 {
            margin-top: 0;
        }
        p {
            color: #555;
            line-height: 1.5;
```

```
        }

        /* Keyframe animations */
        @keyframes fadeIn {
            from { opacity: 0; }
            to { opacity: 1; }
        }
        @keyframes slideInLeft {
            from { transform: translateX(-100px); opacity: 0; }
            to { transform: translateX(0); opacity: 1; }
        }
        @keyframes slideInUp {
            from { transform: translateY(50px); opacity: 0; }
            to { transform: translateY(0); opacity: 1; }
        }
    </style>
</head>
<body>
    <div class="container">

        <img src="static\images\logo1.png" alt="Logo" class="logo">

        <h1>Welcome to Water Potability Analysis</h1>
        <h2>Distribution of Unsafe and Safe Water</h2>
        <div class="plot">
            {{static_plot|safe}}
        </div>


        <form action="/" method="post">
            <label for="feature">Select a feature:</label>
            <select name="feature" id="feature">
                { {% for key, value in dropdown_options.items() %}
```

```
133         <option value="{{ key }}" {% if key == selected_feature %}  selected
            ↪  {% endif %}>{{ value }}</option>
134         {% endfor %}
135             </select>
136             <button type="submit">Submit</button>
137         </form>
138
139
140         <div class="plot">
141             {{ histogram_plot|safe }}
142         </div>
143
144
145         <div class="model-results">
146             <h2>Model Results:</h2>
147             <p>Model used: Quadratic Discriminant Analysis</p>
148             <p>Accuracy: {{ accuracy }}%</p>
149             <p>Precision: {{ precision }}</p>
150             <p>Recall: {{ recall }}</p>
151             <p>F1 Score: {{ f1 }}</p>
152         </div>
153
154
155         <div class="confusion-matrix">
156             <h2>Confusion Matrix:</h2>
157             <div>
158                 {{ conf_matrix|safe }}
159             </div>
160         </div>
161     </div>
162 </body>
163 </html>
```

4.3 Theory

4.3.1 Assumptions

- The project assumes a consistent and reliable power supply for the uninterrupted operation of the Automated Water Management System.
- Reliable internet connectivity is assumed for seamless data transmission, enabling real-time monitoring, alerts, and communication between the system and the centralized database.
- The project relies on regular maintenance and calibration of sensors and equipment to ensure accurate and reliable data collection for water quality monitoring.
- Once the water has settled for long enough, the water will get clear enough to pass thorugh enough UV rays for proper illumination throughout the are of purification.

5 Methodology

This chapter deals with the working of the project. The connection diagram, and program explanation are included here.

5.1 Working Principle

The Water Level Measurement Unit works by sending a pulse of low-frequency sound waves from a sound source and measuring the time delay between the produced wave and the reflected wave. Low-frequency sound is used to ensure that the wave propagates as a planar wave in the tube. The time duration measurement works by accepting the pulses through a microphone. The signal from the microphone is sent to the ESP32 and in the ESP, the signal is first filtered and then undergoes peak detections. The two peaks are detected and the maximum peak (which corresponds to the generated pulse) and the second peak (which corresponds to the first reflection). The time difference between the peaks is calculated from sample indexes and the water level is calculated by taking into account by taking into account of the sampling frequency.

5.2 Connection Diagram

The Pulse Detection Circuit consists of a MAX4466 Microphone module that has a preamplifier and built-in low pass filter with adjustable gain which is suitable for audio applications. The output of the module is connected to GPIO 35, configured as 12-bit ADC of the ESP32 and is used to send the two peaks to the ESP which then detects the time delay between the pulse generation and the first reflection. This delay is used to calculate the water level. This functionality may also be implemented using an INMP441 I2S microphone module for greater noise immunity for the audio signal. In this case, GPIO 35 is connected to SCK pin, and GPIO 32 to the SD pin.

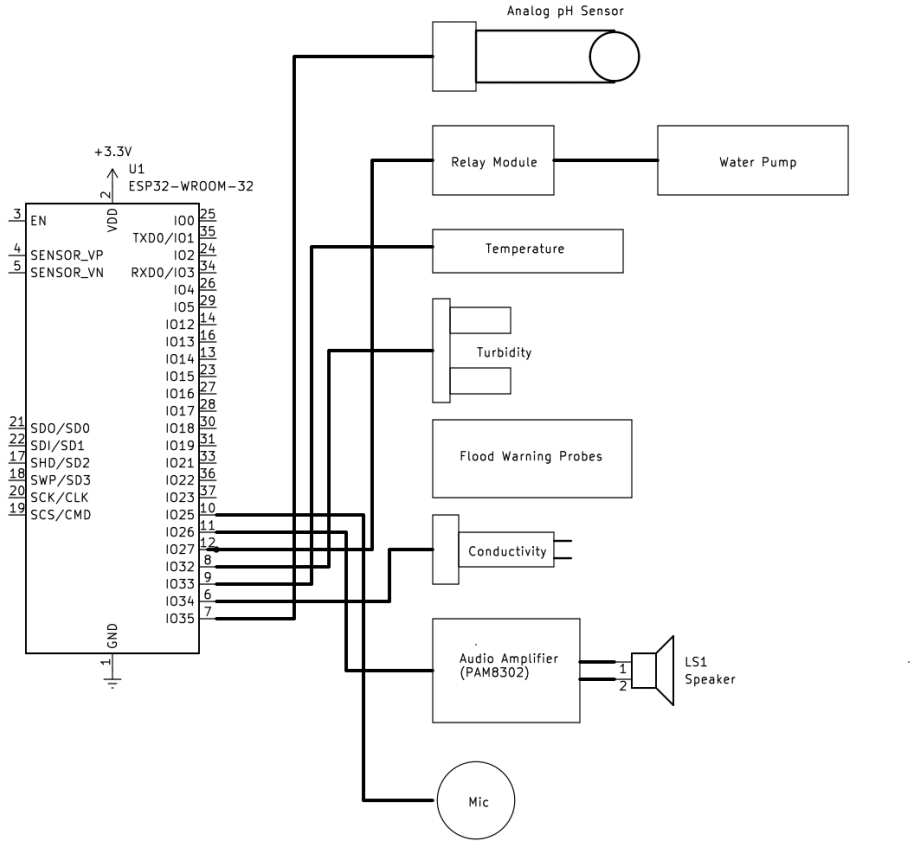


Figure 5.1: Complete Circuit Diagram

5.3 Prototype

A breadboard prototype for the pulse detection circuit was made and verified. However, this hardware implementation was later changed to a software implementation due to ease of setting up and reducing the cost of implementation. The sensor segment was connected and verified along with the automatic pump control was also tested and verified.

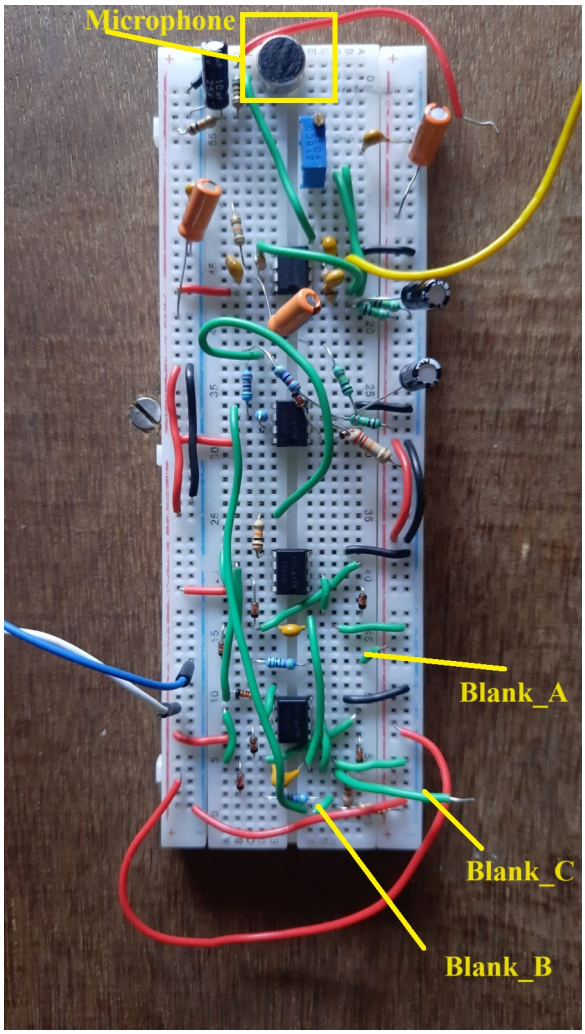


Figure 5.2: Breadboard prototype for the pulse detection circuit

6 Hardware Description

- **Water Level Measurement Unit:** This unit incorporates a sound transmitter and receiver system, which utilizes ultrasonic or similar sound-based technology to measure the water level within the tank. The transmitter emits sound waves, which bounce off the surface of the water and return to the receiver. By calculating the time taken for the sound waves to travel to the water surface and back, the microcontroller can determine the water level within the tank.

- **Automated Water Pump:** The automated water pump is responsible for pumping water into or out of the tank based on the water level detected by the water level measurement unit. The microcontroller controls the operation of the water pump, automatically turning it on or off as needed to maintain the desired water level.

- **Water Quality Analysis Unit:** This unit comprises various sensors to monitor the quality of the water inside the tank. It commonly includes sensors such as pH (acidity/alkalinity), turbidity, and possibly total dissolved solids (TDS) sensors. These sensors continuously or periodically measure the respective parameters of the water and provide data to the microcontroller for analysis.

- **Control Unit:** The control unit consists of the microcontroller, which serves as the central processing unit of the system. It receives inputs from the water level measurement unit and water quality analysis unit, processes the data, and controls the operation of the water pump accordingly. Additionally, it may include other components such as memory storage, communication modules (e.g., Wi-Fi, Bluetooth), and interfaces for user interaction.

- **Enclosure:** Weather-proof enclosures are used for the sensors at the outlet section of

the water tank to ensure that the sensors are always in contact with the water. Another enclosure for the Water Level Sensing unit is to be placed at the top of the water tank, which also houses the Flood Warning Probes. This enclosure has a hole for preventing the build-up of pressure inside the sensor. Another enclosure weather-proof enclosure is to be used for housing the ESP32, which is also placed on top of the tank.

7 Results and Discussion

Through the integration of machine learning models, a user-friendly application, and the deployment of appropriate sensors for water parameter detection and purification, significant progress has been achieved in enhancing water management efficiency. The results obtained demonstrate the successful implementation of advanced technologies to monitor and optimize water quality, ensuring safer and more sustainable water resources. The developed Machine Learning model, coupled with real-time sensor data, has facilitated accurate prediction and prompt action for maintaining optimal water conditions. Additionally, the user-friendly application provides stakeholders with intuitive access to crucial information, empowering informed decision-making for effective water resource management. The water level measurement circuit has been built and tested and a real-time database was also created and connected with the ESP32. Overall, the successful completion of this phase marks a significant milestone towards addressing water sustainability challenges, paving the way for further advancements in the field.

8 Conclusion

In conclusion, the development and implementation of the water parameter monitoring and purifying system for the ECE department represent a significant milestone in advancing technological solutions for sustainable water management within our college community. This project not only showcases the interdisciplinary synergy between environmental science and electronics engineering but also exemplifies a proactive approach to addressing local challenges. The integration of vital sensors and a robust purification system, underscores the system's effectiveness in real-time water quality assessment and remediation. The progress of this project provides a tangible demonstration of its practical utility and potential for broader campus-wide applications. As we move forward, the knowledge and insights gained from this project contribute not only to the betterment of water resource management but also stand as a testament to the innovative capacity of the ECE department in tackling contemporary environmental issues. This endeavor serves as a model for leveraging technology to enhance sustainability efforts within educational institutions, fostering a culture of responsibility and engagement in addressing the environmental challenges of our time.

Bibliography

[1] J.-H. Xu and A.-L. Luo, "Research on water resources automatic monitoring and management system," in *2012 Fourth International Conference on Computational and Information Sciences*, 2012, pp. 1135–1138.

[2] G. Wiranto, G. A. Mambu, Hiskia, I. D. P. Hermida, and S. Widodo, "Design of online data measurement and automatic sampling system for continuous water quality monitoring," in *2015 IEEE International Conference on Mechatronics and Automation (ICMA)*, 2015, pp. 2331–2335.

[3] S. Geetha and S. Gouthami, "Internet of things enabled real time water quality monitoring system," *Smart Water*, vol. 2, no. 1, p. 1, Jul. 2017.

[4] R. P. N. Budiarti, A. Tjahjono, M. Hariadi, and M. H. Purnomo, "Development of iot for automated water quality monitoring system," in *2019 International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE)*, 2019, pp. 211–216.

[5] S. Anand, A. Nath, A. P. Jayan, B. Brunda.I.B., and C. Vidyashree.C, "Automatic water management system." EAI, 1 2021.