# Water Quality Monitoring System Using IOT and TinyML

Members:

Samet Can Şahin – sametcan.sahin@agu.edu.tr

Derya İpek Er – deryaipek.er@agu.edu.tr

Serdar Albayrak – serdar.albayrak@agu.edu.tr

Advisor: Dr Abdulkadir Köse

**GitHub Link:**

https://github.com/SametCanSahin/COMP413-G4-WaterQualitySystem/tree/main

# I. Introduction

As the global water resource quality and quantity are diminishing, the importance of eco-friendly systems like rainwater harvesting and water recycling is rising. But the evaluation of the rainwater for recycling, or for any other purpose, usually relies on conventional methods that are very costly and give no timely feedback." The "Smart Water Quality Monitoring System Using TinyML & IoT" developed for this thesis uses the capabilities of TinyML and Internet of Things to determine the suitability of the harvested water for recycling in real-time. The authors claim that by processing pH, TDS, and temperature values directly on the ESP32 microcontroller, it offers an "independent and economical approach that can give instant results in the form of Reusable/Not Suitable.

# II. Objective

**System Model:**

The main objective of this project is to develop an intelligent, autonomous and low-cost water monitoring system that analyzes in real time the reuse potential of resources such as rainwater or greywater, using TinyML and IoT technologies.

## a) Specific Objectives:

Real-Time Data Acquisition:

The system integrates DS18B20 pH (acidity/alkalinity), TDS (Total Dissolved Solids) and temperature sensors to collect real-time data for the analysis of the chemical and physical properties of water.

## b) TinyML Application:

The goal is to train an artificial neural network (ANN) model using the Edge Impulse platform to process data from collected sensors and run this model on an ESP32 microcontroller to classify the device as "consumable" or "non-consumable" without needing an internet connection.

## c) Hybrid Decision-Making Mechanism and Security:

In addition to AI predictions, the system's reliability will be enhanced by the development of a rule-based safety protocol that will automatically mark water as "unfit for consumption" when the pH value is below 6.0 or above 9.0.

## d) IoT Integration and Remote Monitoring:

To enable the transfer of processed data and classification results to the Blynk IoT Cloud platform via the Wi-Fi module.

### e) User-Friendly Feedback Interface:

The mobile application visualizes the water status for users using clear color codes (Green for potable, Red for non-potable) and real-time sensor readings.

### f) Contribution to Sustainability Goals:

The system developed aims to provide a technological solution to the United Nations Sustainable Development Goals "Clean Water and Sanitation (SDG 6)" and "Sustainable Cities and Communities (SDG 11)".

## III. Background:

The parts used for this project are listed below:

**Hardware:**

Perforated conductor plate

Copper wires

Esp32 dev module

Arduino PH Sensor

Liquid Temperature Sensor (DS18B20)

TDS sensor (Total Dissolved Solids)

Pin headers

**Software:**

Arduino IDE

Updated Python Platform

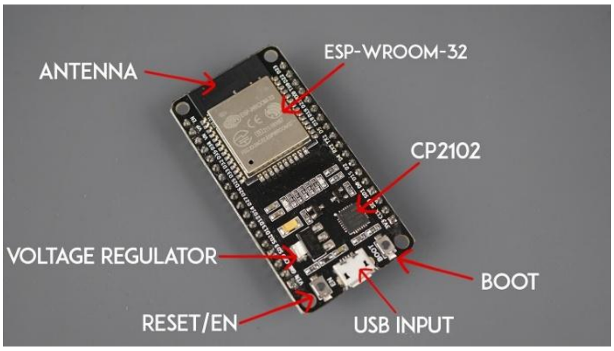
Edge Impulse Platform

Blynk IOT Platform

## IV. Methodology:

The system hardware is designed to accurately measure water quality parameters (pH, TDS, Temperature) and process this data on the on-device.

## Hardware:




**Central Processing Unit:** Microcontroller (Central Processing Unit): The microcontroller employed in this project is the ESP32 microcontroller. The ESP32 Microcontroller was selected for its strong processing ability and its ability to conduct Wi-Fi and Bluetooth functions, allowing it to execute TinyML models. This acts as the "Edge Processing Unit," which handles the process of decision-making by processing data from sensors.

**Data Acquisition Layer:**

**Temperature Sensor (DS18B20):** This is a water-proof digital sensor used for temperature measurement and temperature compensation for other sensors, including TDS. In my code design, this sensor is connected to GPIO pin number 32 using the One-Wire protocol. To be able to obtain meaningful data from the sensor pull-up resistors needed in circuitry design.

**TDS Sensor (Analog):** Used to measure the total amount of dissolved solids in water and to estimate electrical conductivity. It is connected to GPIO 34 analog pin.

**pH Sensor (Analog):** Used to measure the acidity or alkalinity level of water. It is connected to the GPIO 35 analog pin. This sensor operates at 5V so that for the secure data acquisition voltage divider is needed at the signal end of this sensor.

### Circuitry:

Sensors operate at different voltages, so that for the PH sensor voltage divider is required. For temperature sensor 5k pull-up resistor required.

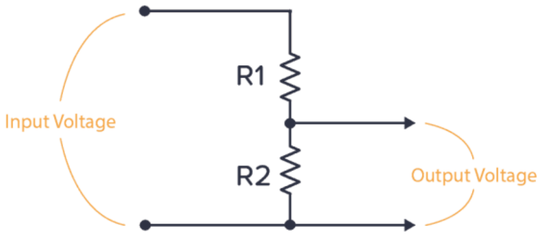



*Figure 1- Voltage Divider*

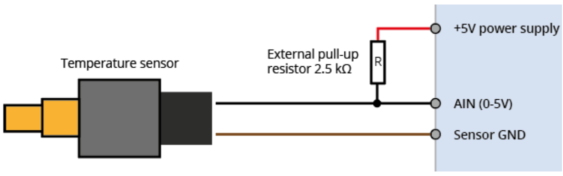



*Figure 2- Pull-up Resistor*

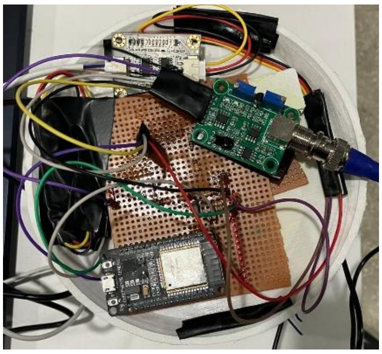


*Figure 3 – Complete Circuitry*

## Software Design:

The software architecture is built on the "Sense-Think-Act" cycle and consists of three main phases: Data Preprocessing, TinyML Inference, and IoT Communication.

### Data Acquisition & Pre-processing:

The system collects raw data every 2 seconds. Before inference, the TDS value is compensated using the temperature reading from the DS18B20 sensor (formula → $1.0 + 0.02 * (temp - 25.0)$) to ensure accuracy.

### Implementation of TinyML:

The decision-making algorithm is designed using an Artificial Neural Network (ANN) model developed through Edge Impulse. This model is then used in the ESP32 board via the "WaterQuality_inferencing.h" library. A hybrid safety algorithm will override the AI prediction in case of values out of range for pH (< 6.0 or > 9.0), and it will automatically classify water quality as "Not Suitable."

**IoT Integration:**

Data processing and classification outputs are connected to the Blynk IoT Cloud through Wi-Fi. Data is displayed in real-time parameters (V0, V1, V2), as well as the status of water through colors (Green (23C48E), indicating Reusable water, and Red (D3435C), indicating Not Suitable water).

**Embedded Code:**

```
/* ============================================================
   ESP32 SU KALİTE SİSTEMİ - RENKLİ VERSİYON (COLOR CHANGING)
   - Consumable Water -> YEŞİL
   - Unconsumable Water -> KIRMIZI
   ============================================================ */

// 1. BLYNK AYARLARI
#define BLYNK_TEMPLATE_ID "TMPL6NcE-av5V"
#define BLYNK_TEMPLATE_NAME "Water Quality System"
#define BLYNK_AUTH_TOKEN "9jRlw8yJNDOjcR2CPXPf1pdLUCJm8TQV"

// 2. WIFI AYARLARI
char ssid[] = "AGUN";
char pass[] = "A03q8uN1";

// 3. KÜTÜPHANELER
#define BLYNK_PRINT Serial
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include "DFRobot_ESP_PH.h"

// !!! KÜTÜPHANE İSMİNİ KONTROL ET !!!
#include <WaterQuality_inferencing.h>

// 4. PIN TANIMLAMALARI
#define PIN_PH 35
#define PIN_TDS 34
#define PIN_TEMP 32

// Nesneler
OneWire oneWire(PIN_TEMP);
DallasTemperature sensors(&oneWire);
DFRobot_ESP_PH ph;
BlynkTimer timer;

// Değişkenler
float tdsDegeri = 0;
```

```
float phDegeri = 0;
float sicaklik = 0;

static bool debug_nn = false;

/* ============================================================
   ANA FONKSİYON
   ============================================================ */
void yapayZekaVeSensorIslemleri() {

  // --- A) SENSÖRLERİ OKU ---
  sensors.requestTemperatures();
  float okunanSicaklik = sensors.getTempCByIndex(0);
  if (okunanSicaklik > -100) sicaklik = okunanSicaklik;
  else sicaklik = 25.0;

  float voltagePH = analogRead(PIN_PH) / 4095.0 * 3300;
  phDegeri = ph.readPH(voltagePH, sicaklik);

  float voltageTDS = analogRead(PIN_TDS) * (3.3 / 4096.0);
  float compensationCoefficient = 1.0 + 0.02 * (sicaklik - 25.0);
  float compensationVolatge = voltageTDS / compensationCoefficient;
  tdsDegeri = (133.42 * compensationVolatge * compensationVolatge *
compensationVolatge
               - 255.86 * compensationVolatge * compensationVolatge
               + 857.39 * compensationVolatge) * 0.5;

  // Blynk'e Sensör Verilerini Gönder
  Blynk.virtualWrite(V0, sicaklik);
  Blynk.virtualWrite(V1, phDegeri);
  Blynk.virtualWrite(V2, tdsDegeri);

  // --- B) YAPAY ZEKA HAZIRLIĞI ---
  size_t beklenenSayi = EI_CLASSIFIER_DSP_INPUT_FRAME_SIZE;
  float features[beklenenSayi];

  for (int i = 0; i < beklenenSayi; i += 3) {
      features[i] = tdsDegeri;
      if (i+1 < beklenenSayi) features[i+1] = phDegeri;
      if (i+2 < beklenenSayi) features[i+2] = sicaklik;
  }

  // --- C) TAHMİNİ ÇALIŞTIR ---
  signal_t signal;
  numpy::signal_from_buffer(features, beklenenSayi, &signal);

  ei_impulse_result_t result = { 0 };
  run_classifier(&signal, &result, debug_nn);
```

```
  // --- D) SONUCU BELİRLE VE RENK AYARLA ---
  String hamSonuc = "";
  float enYuksekIhtimal = 0.0;

  // En yüksek sonucu bul
  for (size_t i = 0; i < EI_CLASSIFIER_LABEL_COUNT; i++) {
      if (result.classification[i].value > enYuksekIhtimal) {
          enYuksekIhtimal = result.classification[i].value;
          hamSonuc = String(result.classification[i].label);
      }
  }

  // Ekrana Yazılacak Değer ve Rengi
  String ekranaYazilacakSonuc = "Analyzing...";
  String renkKodu = "#FFFFFF"; // Varsayılan Beyaz

  // 1. Durum: TEMİZ SU (YEŞİL)
  if (hamSonuc == "Kullanilabilir" || hamSonuc == "Temiz" || hamSonuc ==
"Icilebilir" || hamSonuc == "consumable") {
      ekranaYazilacakSonuc = "Consumable Water";
      renkKodu = "#23C48E"; // BLYNK YEŞİLİ
  }
  // 2. Durum: KİRLİ SU (KIRMIZI)
  else {
      ekranaYazilacakSonuc = "Unconsumable Water";
      renkKodu = "#D3435C"; // BLYNK KIRMIZISI
  }

  // 3. Durum: pH GÜVENLİK KONTROLÜ (KIRMIZI)
  if (phDegeri < 6.0 || phDegeri > 9.0) {
      ekranaYazilacakSonuc = "Unconsumable Water";
      renkKodu = "#D3435C"; // Tehlike Kırmızısı
  }

  // --- E) BLYNK'E GÖNDER (YAZI + RENK) ---

  // Önce rengi ayarla
  Blynk.setProperty(V3, "color", renkKodu);

  // Sonra yazıyı gönder
  Blynk.virtualWrite(V3, ekranaYazilacakSonuc);

  Serial.print("KARAR: "); Serial.println(ekranaYazilacakSonuc);
}

/* ============================================================
   SETUP VE LOOP
   ============================================================ */
void setup() {
```

```
  Serial.begin(115200);
  sensors.begin();
  ph.begin();

  Serial.print("WiFi: "); Serial.println(ssid);
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);

  timer.setInterval(2000L, yapayZekaVeSensorIslemleri);
}

void loop() {
  Blynk.run();
  timer.run();
}
```

## Box Design:

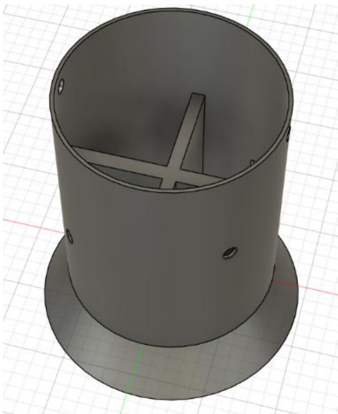


*Figure 4- Box Design on Fusion360*

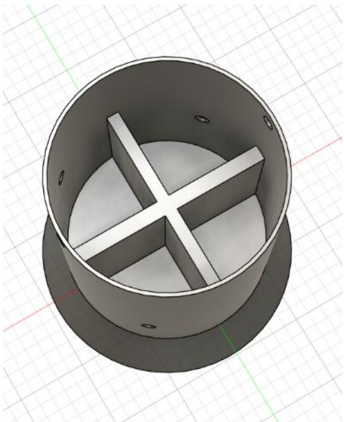


*Figure 5- Box Design*

For the box design, Fusion360 CAD program is used and box is 3d printed.

*Figure 6- 3D Print Result*

## V. Results and Discussion

During the initial stages of the project's development, in order to analyze water quality in more detail, measurements were taken under 4–5 different pollution scenarios to create datasets, which were then uploaded to the Edge Impulse platform. However, testing revealed that the model's multi-layered structure failed to achieve the desired accuracy and frequently misclassified categories. Due to these technical challenges and the inability to obtain reliable results, the model had to be simplified to ensure more stable and dependable system performance. It was retrained to focus on two main classes: "consumable" and "non-consumable" water. The data collected in the current training phase totals 1 minute and 58 seconds. During training, 100% of the dataset was used for training purposes, with no separate test set allocated. This limitation makes it difficult to measure the model's performance and carry out an unbiased evaluation on unseen data. In the AI model design phase, temperature data was considered not to be a direct validation parameter for water usability and was therefore excluded from analysis via the "Flatten" block. However, in the embedded software part of the project, it was observed that three types of data (TDS, pH, and temperature) were included in the loop when creating the features array. Aligning the model's 2-input structure (TDS and pH) with the code's 3-input setup is important for the stability of the project. During data processing on the ESP32, temperature data from the DS18B20 sensor is incorporated into formulas to compensate the TDS value based on temperature and produce more accurate results. To enhance the system's reliability, a hybrid decision-making mechanism and rule-based safety protocol have been developed in addition to AI predictions. This mechanism ensures that if the pH value falls below 6.0 or rises above 9.0 (extreme values) the water status is automatically classified as "Unconsumable Water," regardless of the AI model's prediction, thereby prioritizing safety. All sensor data and analysis outputs are transmitted via the Wi-Fi module to the Blynk IoT Cloud platform. The user-friendly mobile interface provides real-time monitoring by visualizing water status with clear messages and color codes: green for consumable water and red for non-consumable water.

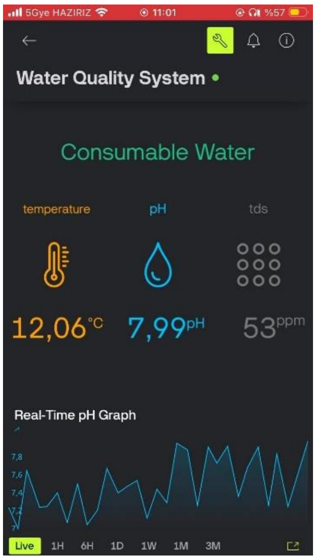



*Figure 7- Consumable Water*

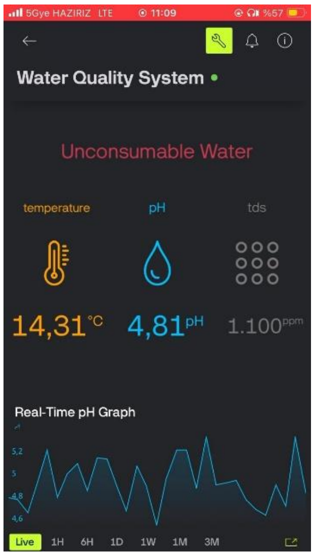



*Figure 8- Unconsumable*

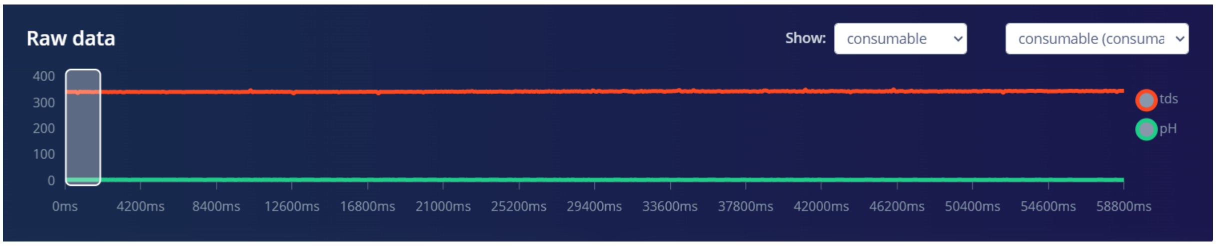



*Figure 9- Consumable Water Raw Data*

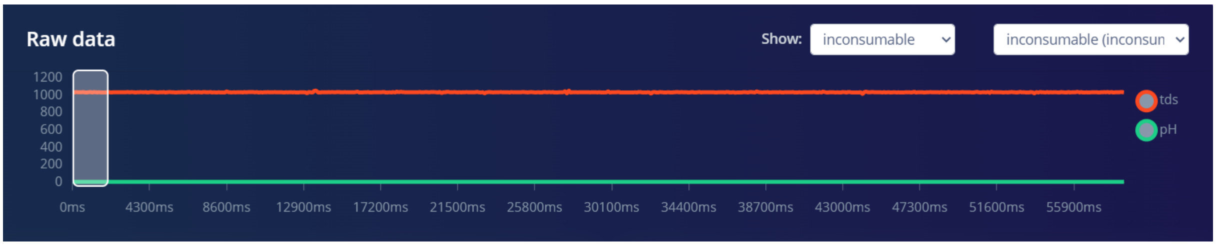



*Figure 10- Unconsumable Water Raw Data*

# VI. Conclusion

In this project, a real-time water quality monitoring system was developed by integrating TinyML, sensor-based data acquisition and IoT communication into a low-cost embedded platform. pH, TDS and temperature measurements were processed on-device using a neural network model deployed on the ESP32 microcontroller, enabling offline classification of water as reusable or not. A rule-based safety layer was added to override the classifier in out-of-range pH conditions to increase decision reliability.

The processed data and classification output were transmitted to the Blynk IoT platform, providing remote visualization and user feedback through a simple interface. The results demonstrate that TinyML-based inference on microcontrollers can provide an efficient, fast and low-cost alternative to traditional laboratory-based analysis methods for rainwater or greywater assessment. Future work may focus on expanding sensor inputs (e.g., turbidity, dissolved oxygen), performing field validation, and optimizing model performance to increase prediction robustness and overall system applicability.

# VII. References

[1] United Nations, "Sustainable Development Goal 6: Clean Water and Sanitation and SDG 11: Sustainable Cities and Communities," 2015.

[2] Espressif Systems, "ESP32 Series Datasheet: High-performance Microcontrollers with Wi-Fi and Bluetooth Capabilities for Edge Computing," 2023.

[3] Edge Impulse, "Edge Impulse Documentation: TinyML Development for Embedded Artificial Neural Networks (ANN)," 2024.

[4] Blynk IoT, "Blynk.Cloud Documentation: Real-Time Data Visualization and IoT Management for Mobile Applications," 2024.

[5] Dallas Semiconductor, "DS18B20 Programmable Resolution 1-Wire Digital Thermometer Datasheet," 2019.

[6] DFRobot, "Gravity: Analog pH Sensor / Meter Kit for Arduino/ESP32 Technical Manual," 2024.

[7] ThingsBoard Authors, "ThingsBoard: Open-source IoT Platform for Device Management, Data Collection, Processing and Visualization," 2024.

[8] TDS Compensation Methods, "Mathematical Models for Total Dissolved Solids (TDS) Temperature Compensation," 2024.

[9] TinyML Methodology, "ANN Modeling and Flattening Block Applications for Multi-Sensor Edge Inference," 2024.