DAM WATER MONITORING USING THINGSBOARD

Name	**: Dea Fauziah Lestari**
	: Wafana Mumtaza Dzauqiyah
Semester	**: 3rd (Third Semester)**
Quarter	**: 2nd (Second Quarter)**
Class	**: 3 ISA 1**

Continuing Education Program Center for Computing and Information Technology

Faculty of Engineering University of Indonesia

2025

PROJECT ON

"Dam Water Monitoring Using ThingsBoard"

Developed by:
Name:

1. **Dea Fauziah Lestari**
2. **Wafana Mumtaza Dzauqiyah**

Faculty: Mr. Listyo Edi Prabowo, S.T., M.T.

Dam Water Monitoring Using ThingsBoard

Batch Code: 3ISA1

Start Date: December 21st, 2024
End Date: January 2nd, 2025

Name of Faculty: Mr. Listyo Edi Prabowo, S.T., M.T.

Names of Developer:

1. Dea Fauziah Lestari
2. Wafana Mumtaza Dzauqiyah

Date of Submission: January 2, 2025

CERTIFICATE

This is to certify that the work completed by Dea Fauziah Lestari and Wafana Mumtaza Dzauqiyahainal in this project titled "Dam Water Monitoring Using ThingsBoard” is original. This report was written as part of our NIIT project.

Coordinator:
Mr. Listyo Edi Prabowo, S.T., M.T.

ACKNOWLEDGMENT

First of all, let's praise our gratitude to Allah SWT for his blessings so that we can complete this ISAS entitled “Embedded System: Dam Water Monitoring”. Don’t forget to thank our faculty, Mr. Listyo Edi Prabowo, S.T., M.T. for the guidance that has been given to us to arrange this paper and our friends who have supported us until this very day.

The purpose in writing this paper is to fulfill our curiosity regarding how sensors work and are implemented. The writers chose this title because of the motivation to help those who work in this field to monitor the water flow in a Dam, especially in rural areas considering that Indonesia just passed the rainy season and the Dam played a significant role in controlling the water flow across cities.

We hope this paper will be helpful and useful for everyone who reads it. We are open for critics and suggestions for our improvement on future papers.

Depok, December 30th, 2024

Authors

SYSTEM ANALYSIS

System Summary:

The Internet of Things (IoT) refers to a vast and interconnected network of physical devices equipped with sensors, software, and other technologies that enable them to exchange data without requiring direct human involvement. This concept, first introduced by computer scientist Kevin Ashton in 1999, has since evolved into a cornerstone of modern technology.

In this project, the authors have created a system to monitor water levels in Dams. It comprises ESP8266 and sensors that can upload data to ThingsBoard. This enables an efficient monitoring and alerts the users of any danger.

System Processes:

The sensor-based monitoring system for water dams involves several key processes. Ultrasonic sensors measure water levels using sound waves, flow meter sensors track water flow rates through outlets, and DHT sensors provide environmental data like temperature and humidity. These sensors collect real-time data, which is processed, transmitted to a centralized system, and visualized on ThingsBoard.

COMPONENTS

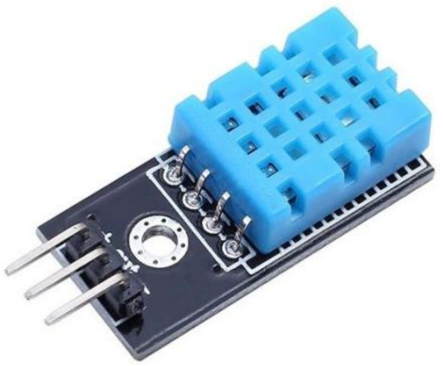
1. **NodeMCU ESP 8266**

2. **Ultrasonic Sensor**

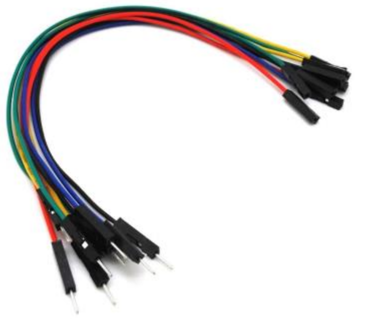
3. **Flow Meter**

COMPONENTS

4. **DHT11 Sensor**

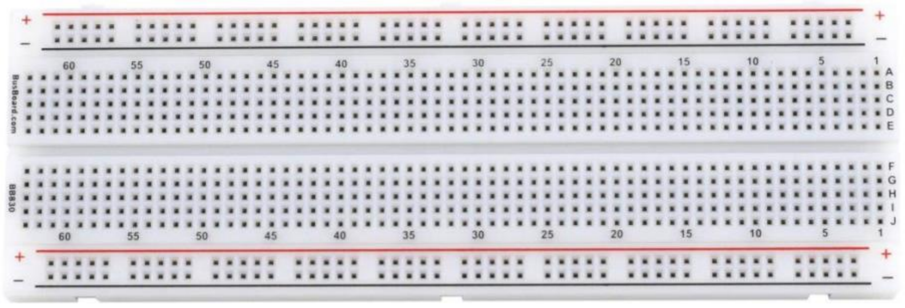


5. **Jumper Wires**

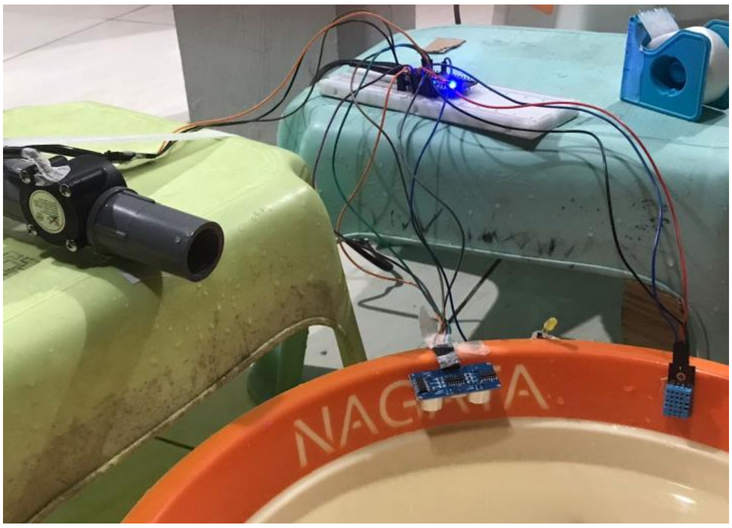


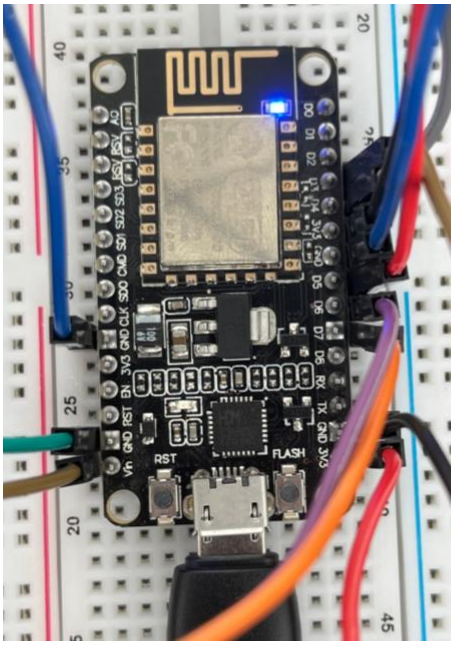
6. **LED**

7. **BreadBoard**



SCHEMATIC





Pin List:

- Ultrasonic Sensor
- Trigger : D7
- Echo : D6

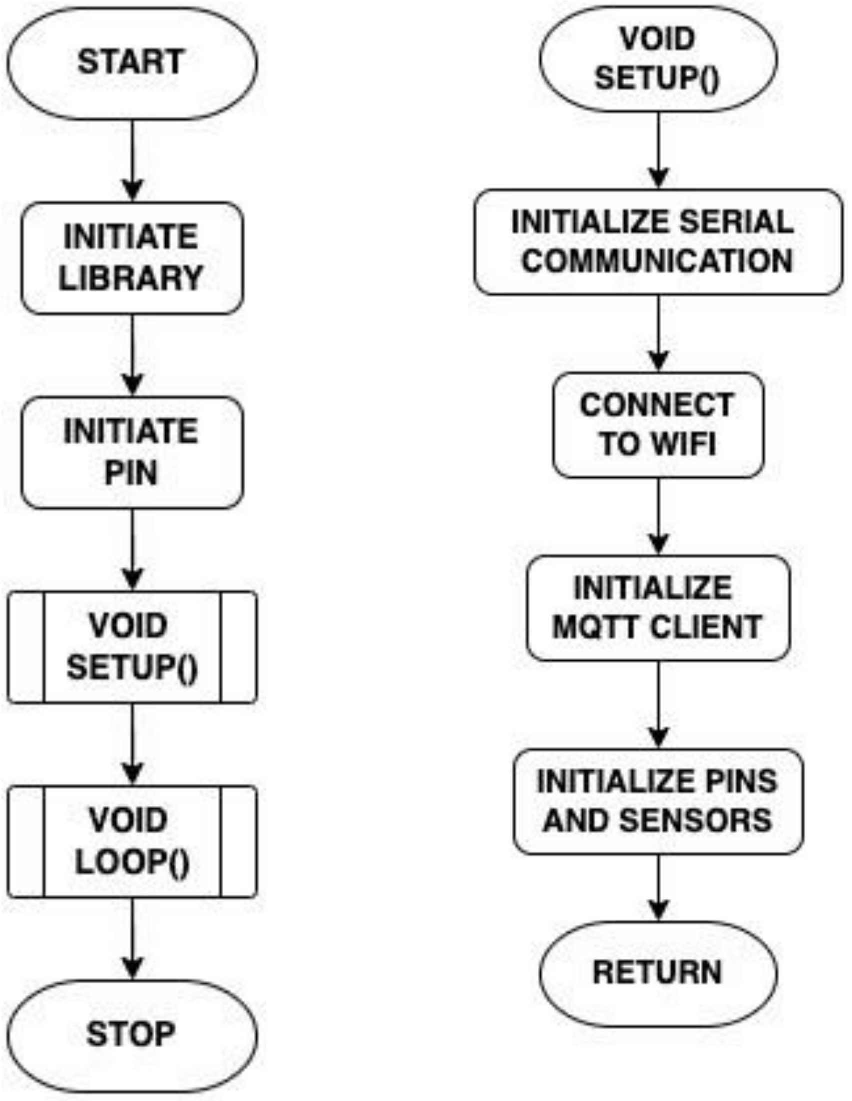
DHT Sensor data pin : D5

Flow Meter Senso data Pin : D3

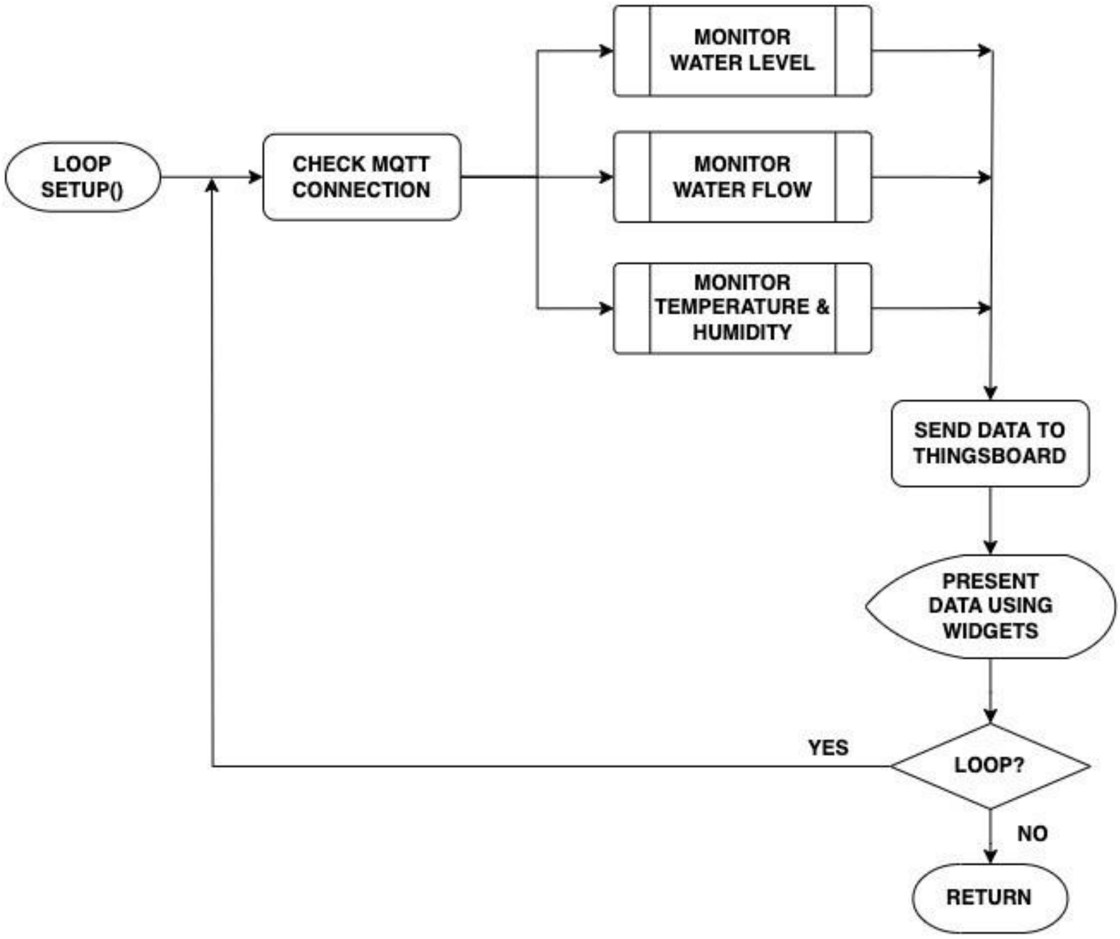
LED Actuator control pin : D4

- Power Connections:
  - All sensors and actuators are powered by the 3.3V pin.
  - Ground (GND) pins of all sensors and actuators are connected to the GND

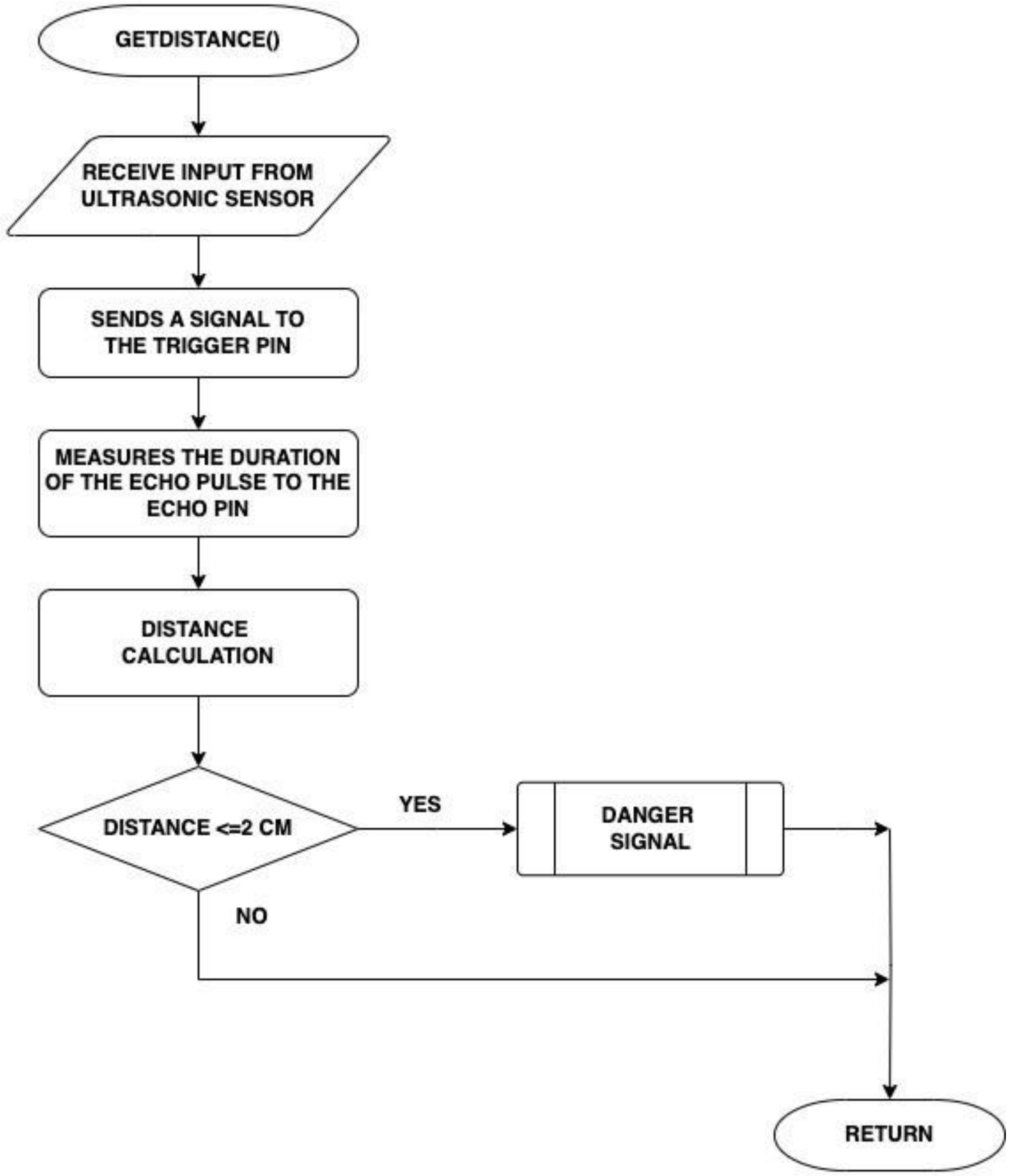
FLOWCHART (MAIN & VOID SETUP)



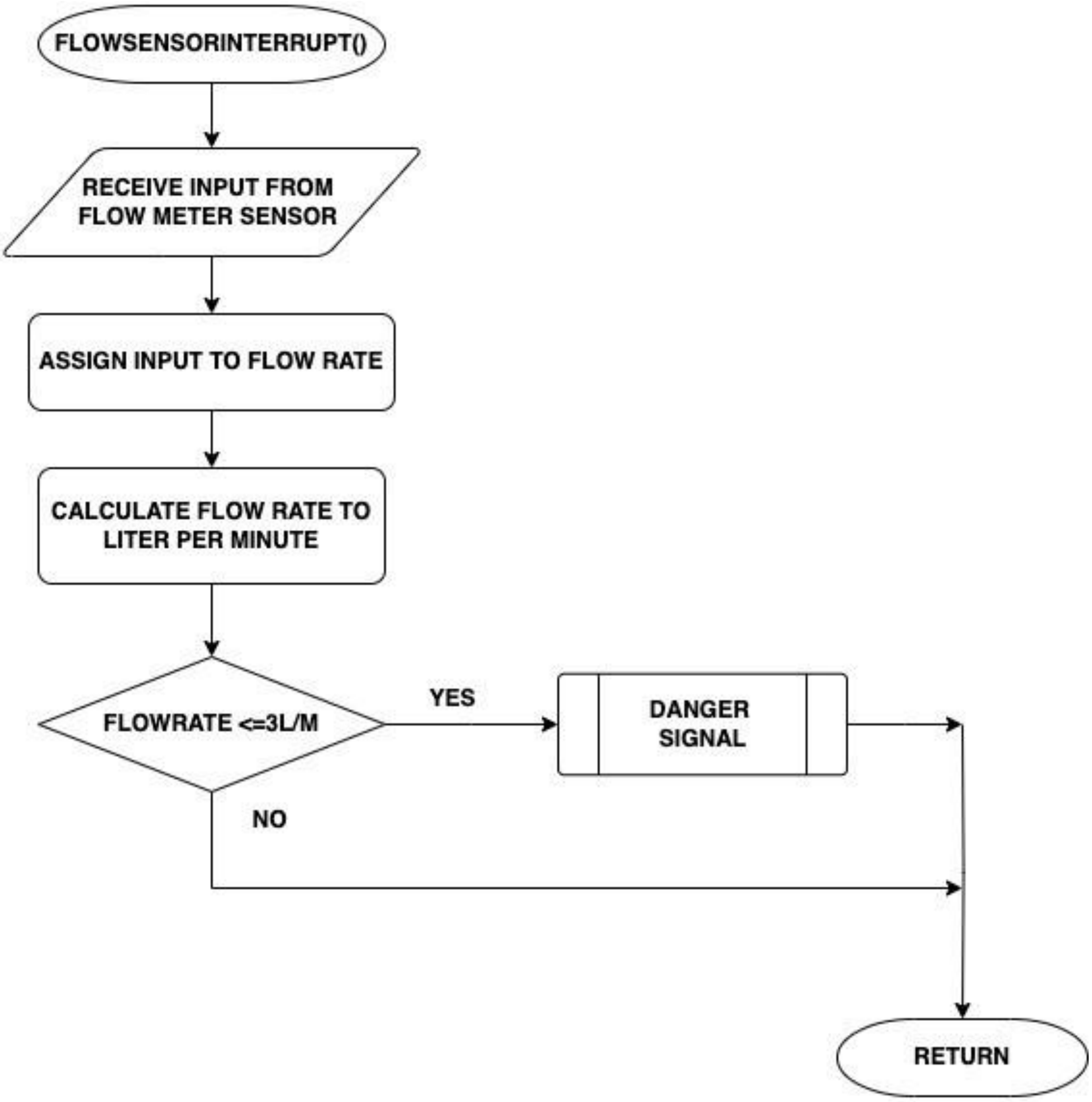
FLOWCHART (LOOP SETUP)



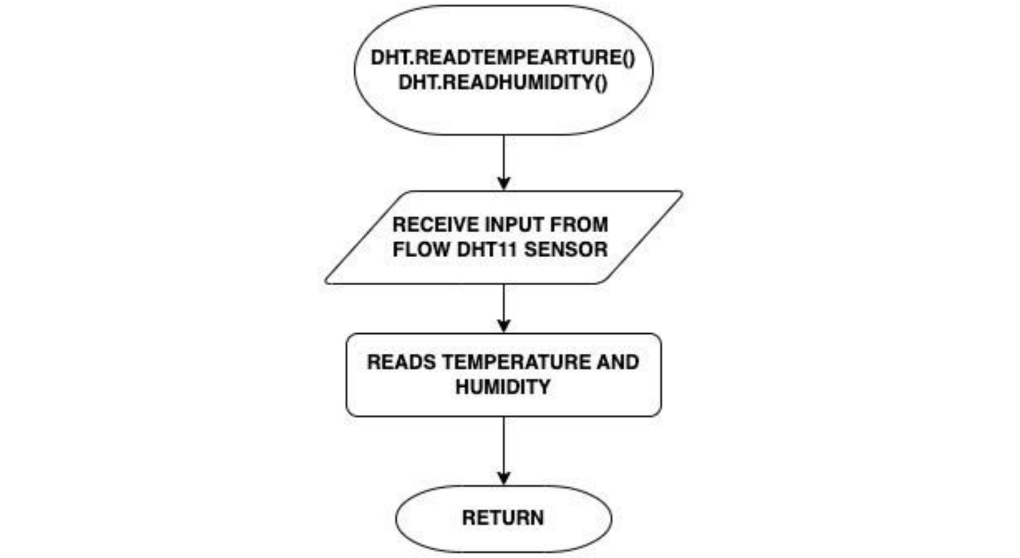
FLOWCHART (ULTRASONIC SENSOR)

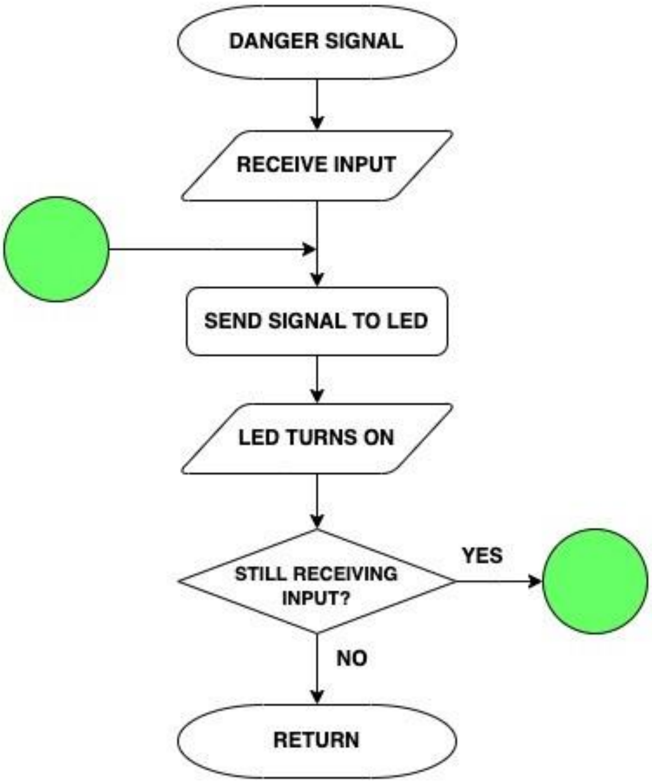

GETDISTANCE()
RECEIVE INPUT FROM ULTRASONIC SENSOR
SENDS A SIGNAL TO THE TRIGGER PIN
MEASURES THE DURATION OF THE ECHO PULSE TO THE ECHO PIN
DISTANCE CALCULATION
DISTANCE <=2 CM
YES
DANGER SIGNAL
NO
RETURN

FLOWCHART (FLOW METER)

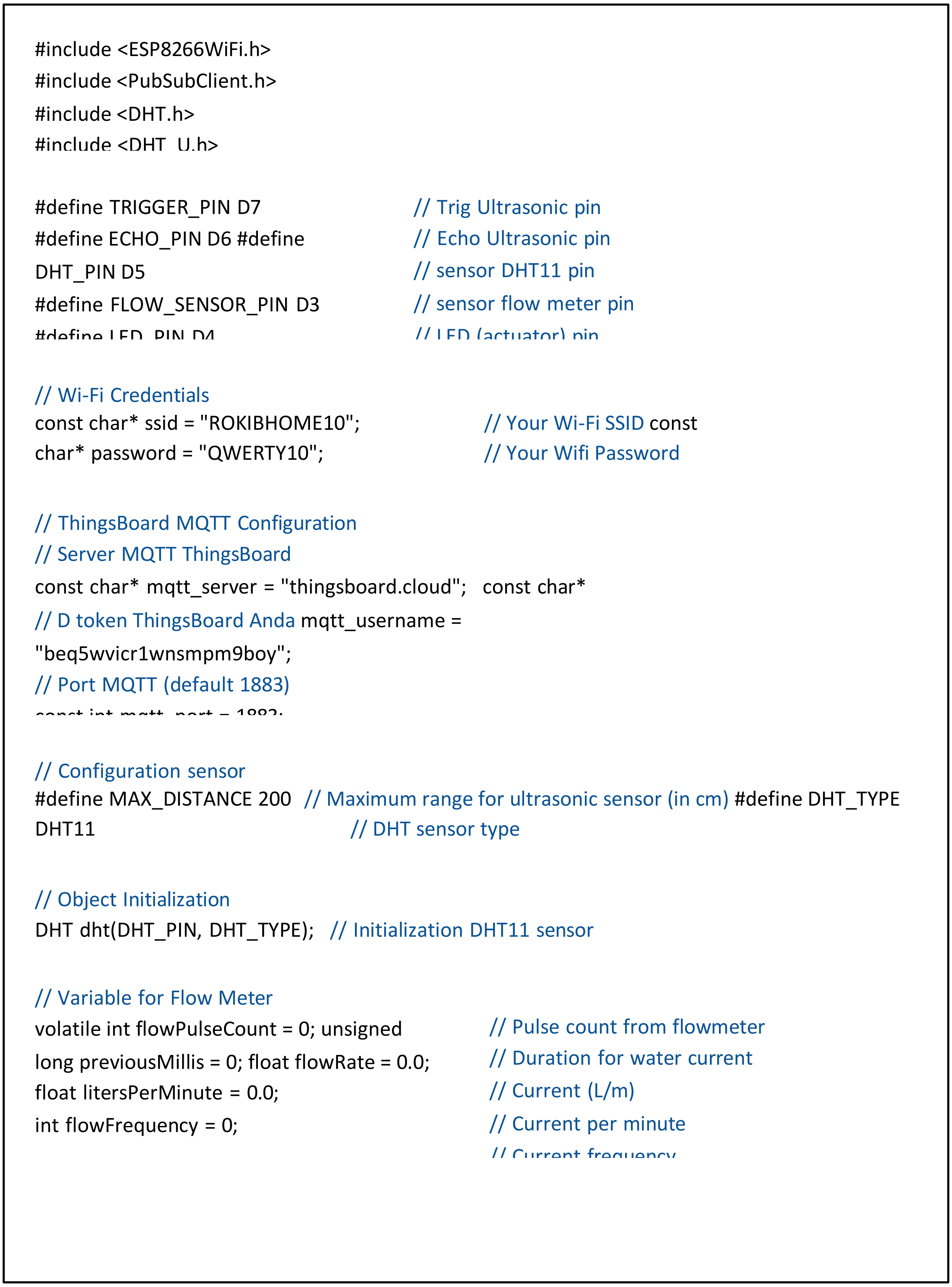

FLOWSENSORINTERRUPT()
RECEIVE INPUT FROM FLOW METER SENSOR
ASSIGN INPUT TO FLOW RATE
CALCULATE FLOW RATE TO LITER PER MINUTE
FLOWRATE <=3L/M
YES
DANGER SIGNAL
NO
RETURN

FLOWCHART (DHT SENSOR AND DANGER SIGNAL)





SOURCE CODE



```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <DHT.h>
#include <DHT_U.h>

#define TRIGGER_PIN D7                    // Trig Ultrasonic pin
#define ECHO_PIN D6 #define               // Echo Ultrasonic pin
DHT_PIN D5                                // sensor DHT11 pin
#define FLOW_SENSOR_PIN D3                // sensor flow meter pin
#define LED_PIN D4                        // LED (actuator) pin

// Wi-Fi Credentials
const char* ssid = "ROKIBHOME10";                // Your Wi-Fi SSID const
char* password = "QWERTY10";                     // Your Wifi Password

// ThingsBoard MQTT Configuration
// Server MQTT ThingsBoard
const char* mqtt_server = "thingsboard.cloud";   const char*
// D token ThingsBoard Anda mqtt_username =
"beq5wvicr1wnsmpm9boy";
// Port MQTT (default 1883)
const int mqtt_port = 1883;

// Configuration sensor
#define MAX_DISTANCE 200  // Maximum range for ultrasonic sensor (in cm) #define DHT_TYPE
DHT11                                   // DHT sensor type

// Object Initialization
DHT dht(DHT_PIN, DHT_TYPE);  // Initialization DHT11 sensor

// Variable for Flow Meter
volatile int flowPulseCount = 0; unsigned        // Pulse count from flowmeter
long previousMillis = 0; float flowRate = 0.0;   // Duration for water current
float litersPerMinute = 0.0;                     // Current (L/m)
int flowFrequency = 0;                           // Current per minute
                                                 // Current frequency
```

SOURCE CODE

```
// Interrupt function to count flowmeter pulse void
IRAM_ATTR flowSensorInterrupt() {
    flowPulseCount++;
}


// WiFi dan MQTT Client WiFiClient
espClient; PubSubClient client(espClient);


void setup() {
    // Serial communication initialization
    Serial.begin(115200);   Serial.println("Sensor
    Initialization...");

    // Wi-Fi initialization WiFi.begin(ssid,
    password);
    while (WiFi.status() != WL_CONNECTED) { delay(500);
        Serial.print(".");
    }
    Serial.println("WiFi  Connected");

    // MQTT initialization client.setServer(mqtt_server,
    mqtt_port); client.setCallback(mqttCallback);

    // DHT sensor initialization dht.begin();

    // Setup for flow meter pin pinMode(FLOW_SENSOR_PIN,
    INPUT_PULLUP);
    attachInterrupt(digitalPinToInterrupt(FLOW_SENSOR_PIN), flowSensorInterrupt, RISING);


    // Setup for Ultrasonic sensor pin pinMode(TRIGGER_PIN,
    OUTPUT); pinMode(ECHO_PIN, INPUT);
```

SOURCE CODE

```
    // Setup for LED pin
    pinMode(LED_PIN, OUTPUT);
    digitalWrite(LED  PIN, LOW);                    // ensure the LED is disabled first

    Serial.println("Setup completed!");
}

void loop() {
    // Reconnect ke MQTT jika terputus if
    (!client.connected()) {
        reconnectMQTT();
    }
    client.loop();   // Maintain connection to ThingsBoard

    // Read data from ultrasonic sensor unsigned int
    distance = getDistance(); if (distance == 0) {
        Serial.println("Out of range");
    } else { Serial.print("Distance: ");
        Serial.print(distance); Serial.println("
        cm");
        sendToThingsBoard("distance", distance);                    // Kirim data jarak ke ThingsBoard
    }



    // Read data from DHT11sensor
    float humidity = dht.readHumidity();
    float temperature = dht.readTemperature();

    if (isnan(humidity) || isnan(temperature)) { Serial.println("Failed to
        read from DHT sensor");
    } else { Serial.print("Temperature: ");
        Serial.print(temperature); Serial.print("°C
        Humidity: "); Serial.print(humidity);
        Serial.println("%");
```

SOURCE CODE

```
     sendToThingsBoard("temperature",   temperature);                    //  Send  temperature
data to ThingsBoard sendToThingsBoard("humidity", humidity);
data to ThingsBoard                                                             //  Send  humidity
    }


    // counting water current from flowmeter
    unsigned long currentMillis = millis();
    if (currentMillis - previousMillis >= 1000) {  // every 1 sec flowRate =
        flowPulseCount;
        litersPerMinute  =  flowRate  /  7.5;                    //  convert  pulse  to             liter  per
minute (for YF-S201 sensor (flow meter))

        // show flow rate
        Serial.print("Flow Rate: "); Serial.print(flowRate);
        Serial.print(" pulses per second, ");
        Serial.print("Liters per Minute: ");
        Serial.println(litersPerMinute);


        sendToThingsBoard("flow_rate",  litersPerMinute);                     // Send      flow  rate
data ThingsBoard

        // Reset pulse count for the next count
        flowPulseCount = 0; previousMillis =
        currentMillis;
    }

    // Check LED condition and control
    if (distance <= 2 && litersPerMinute >= 3.0) {
        // If distance <= 2 cm and water current >= 3 L/m, flash LED
        digitalWrite(LED_PIN, HIGH);
        Serial.println("LED  ON");
    } else {
        // if the condition is not fulfilled, turn off LED digitalWrite(LED_PIN, LOW);
        Serial.println("LED  OFF");
    }

    delay(1000);   // Delay for higher readability in serial monitor
}
```

SOURCE CODE

```
// Function to send data to ThingsBoard
void sendToThingsBoard(String key, float value) {
    String payload = String("{\"") + key + "\":" + String(value) + "}";
    client.publish("v1/devices/me/telemetry", payload.c_str());
}

// Function to reconnect to MQTT void
reconnectMQTT() {
    while (!client.connected()) { Serial.print("Attempting MQTT
        connection...");

        // try MQTT connection
        if (client.connect("NodeMCU", mqtt_username, "")) { Serial.println("connected");
        } else { Serial.print("failed, rc=");
            Serial.print(client.state());   delay(5000);  // Coba lagi
            setelah 5 detik
        }
    }
}

// Callback function to handle messages received from ThingsBoard void
mqttCallback(char* topic, byte* payload, unsigned int length) {
    Serial.print("Message arrived [");
    Serial.print(topic); Serial.print("] ");
    for (int i = 0; i < length; i++) { Serial.print((char)payload[i]);
    }
    Serial.println();
}
```

SOURCE CODE

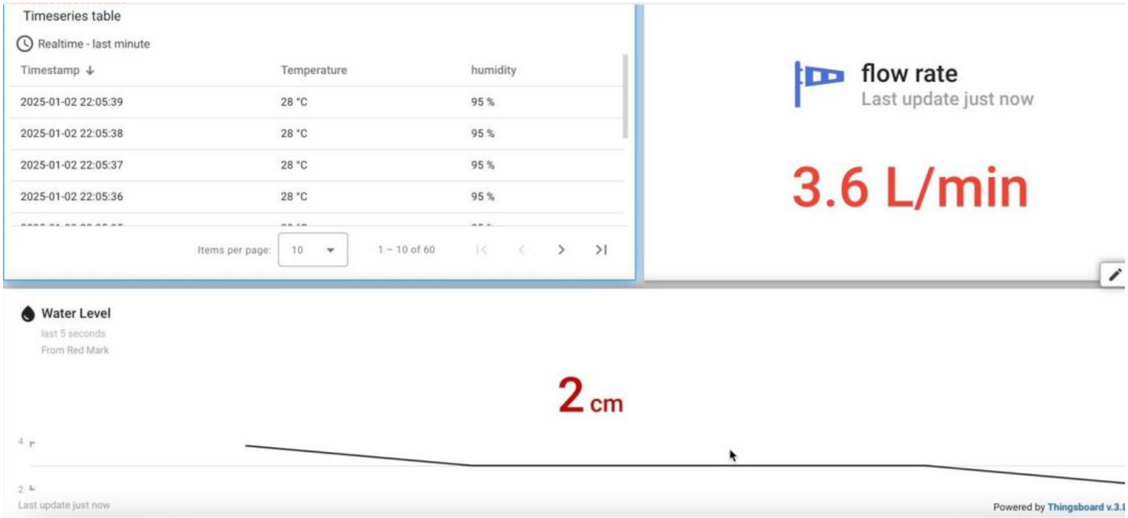
```
// Function to get distance data from Ultrasonic sensor unsigned int getDistance() {
    // Sending pulse to trigger digitalWrite(TRIGGER_PIN, LOW);
    delayMicroseconds(2);  // Delay digitalWrite(TRIGGER_PIN,
    HIGH);
    delayMicroseconds(10);  // Send a trigger pulse for 10 microseconds digitalWrite(TRIGGER_PIN,
    LOW);

    // Read echo pulse
    long duration = pulseIn(ECHO_PIN, HIGH);

    // Counting the distance (in cm)
    unsigned int distance = duration * 0.0344 / 2;   // The speed of sound is 0.0344 cm/us, divide
by 2 because it goes and comes back.
    return distance;
}
```

OUTPUT IN THINGSBOARD AND SERIAL MONITOR

Data Table on ThingsBoard.



Timeseries table

Realtime - last minute

Timestamp ↓	Temperature	humidity
2025-01-02 22:05:39	28 °C	95 %
2025-01-02 22:05:38	28 °C	95 %
2025-01-02 22:05:37	28 °C	95 %
2025-01-02 22:05:36	28 °C	95 %

Items per page: 10 1 – 10 of 60

flow rate

Last update just now

3.6 L/min

Water Level

last 5 seconds

From Red Mark

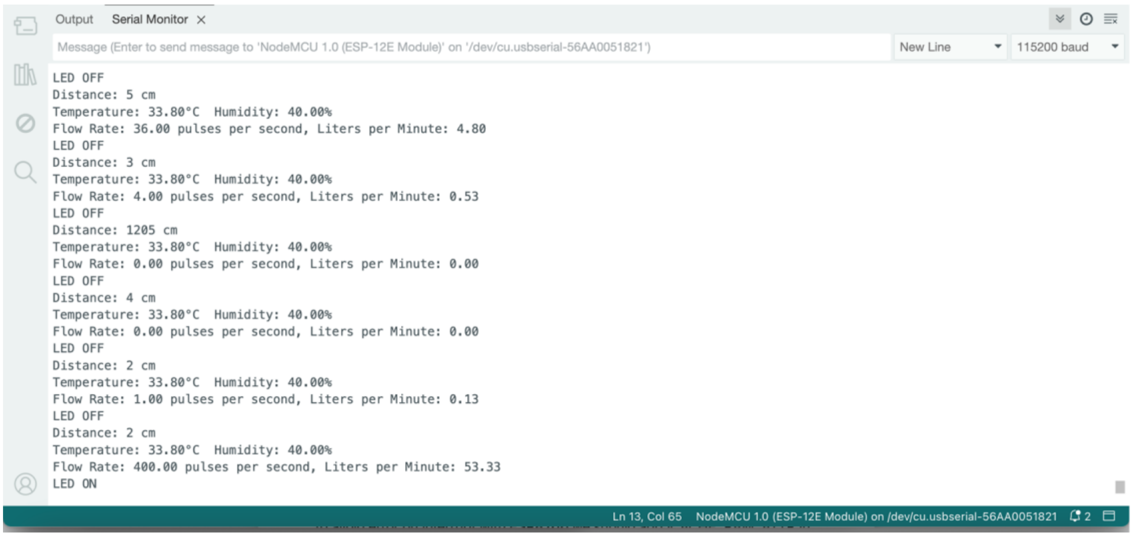
2 cm

Last update just now

Data are presented real-time.

It is shown that the flow rate exceeded the safe limit, alongside water level that has risen cutting the distance to 2 centimeters to the sensor. If these condition occurs, the LED will lit up as a sign of danger.

Serial Monitor on Arduino IDE.



Message (Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)' on '/dev/cu.usbserial-56AA0051821')

New Line 115200 baud

```
LED OFF
Distance: 5 cm
Temperature: 33.80°C  Humidity: 40.00%
Flow Rate: 36.00 pulses per second, Liters per Minute: 4.80
LED OFF
Distance: 3 cm
Temperature: 33.80°C  Humidity: 40.00%
Flow Rate: 4.00 pulses per second, Liters per Minute: 0.53
LED OFF
Distance: 1205 cm
Temperature: 33.80°C  Humidity: 40.00%
Flow Rate: 0.00 pulses per second, Liters per Minute: 0.00
LED OFF
Distance: 4 cm
Temperature: 33.80°C  Humidity: 40.00%
Flow Rate: 0.00 pulses per second, Liters per Minute: 0.00
LED OFF
Distance: 2 cm
Temperature: 33.80°C  Humidity: 40.00%
Flow Rate: 1.00 pulses per second, Liters per Minute: 0.13
LED OFF
Distance: 2 cm
Temperature: 33.80°C  Humidity: 40.00%
Flow Rate: 400.00 pulses per second, Liters per Minute: 53.33
LED ON
```

Ln 13, Col 65 NodeMCU 1.0 (ESP-12E Module) on /dev/cu.usbserial-56AA0051821 2

CONFIGURATION

Hardware	**:**	**ESP8266, Macbook Air M1, Macbook Pro**
Software	**:**	**Arduino IDE, Thingsboard**
Operating System	**:**	**MacOS Sonoma**