PROJECT 1
"Smart Aquarium System"

Arranged by : Group 9

1. Fakhri Ali Halamoan Sihombing (2120010053)
2. Muhammad Zidan Satrio (2120010105)

Class	: 3 ISA 3
Faculty	: Mr. Listyo Edi Prabowo

Continuing Education Center for Computing and Information Technology
Faculty of Engineering, University of Indonesia
Kampus Baru UI Depok 16424
2021

SMART AQUARIUM SYSTEM

Batch Code : 3 ISA 3

Start Date : January 16th, 2023

End Date : January 26th, 2023

Name of the Coordinator : Mr. Listyo Edi Prabowo

Names of Developer :

1. Fakhri Ali Halamoan Sihombing (2120010053)

2. Muhammad Zidan Satrio (2120010105))

CERTIFICATE

This is to certify the report, titled **Smart Aquarium System** embodies original work done by Fakhri Ali Halamoan Sihombing and Muhammad Zidan Satrio in fulfilling the project 4 assignment in CCIT-FT UI.

Coordinator,

Mr. Listyo Edi Prabowo

NIIT

ACKNOWLEDGEMENT

Praise be to the presence of Allah, SWT. who has helped us in making this project because without the pleasure of it we will not be able to make this project. we also thank to Mr. Listyo Edi Prabowo as our lecturer and mentor in making this project entitled **Smart Aquarium System.** Hopefully what we have made can be useful in the future.

Best Regards,

Authors

NIIT

SYSTEM ANALYSIS

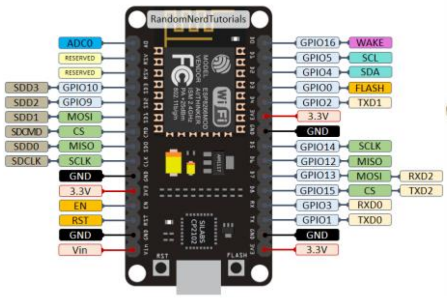
System Summary :

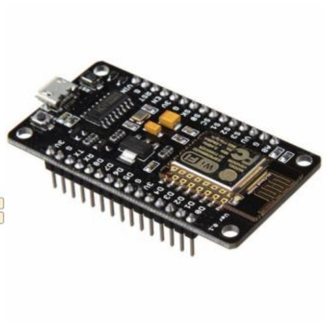
The Internet of Things (IoT) is the concept of computing everyday objects that are connected to the internet and can identify other devices. Today, the Internet of Things is not a strange thing anymore. There are so many IoT systems that are applied in our daily lives. Due to the confluence of technologies, such as ubiquitous computing, commodity sensors, increasingly powerful embedded systems, and machine learning, this field has grown. The Internet of Things is enabled by traditional domains such as embedded systems, wireless sensor networks, control systems, and automation (including home and building automation)

In this project we build a simple smart aqurium system based on nodemcu. The first sensor we use in this project is the DS18B20 temperature sensor which we will combine with an aquarium heater aiming to stabilize water temperature. In addition to stabilizing water temperature, the heater function also helps reduce the population of fungi and bacteria. besides that we also added an ultrasonic sensor to measure the depth of the water so we can monitor the volume of the water

COMPONENT

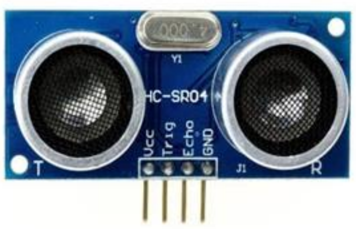
1.Nodemcu ESP 8266





2. Temperature Sensor DS18B20

3. Ultrasonic Sensor



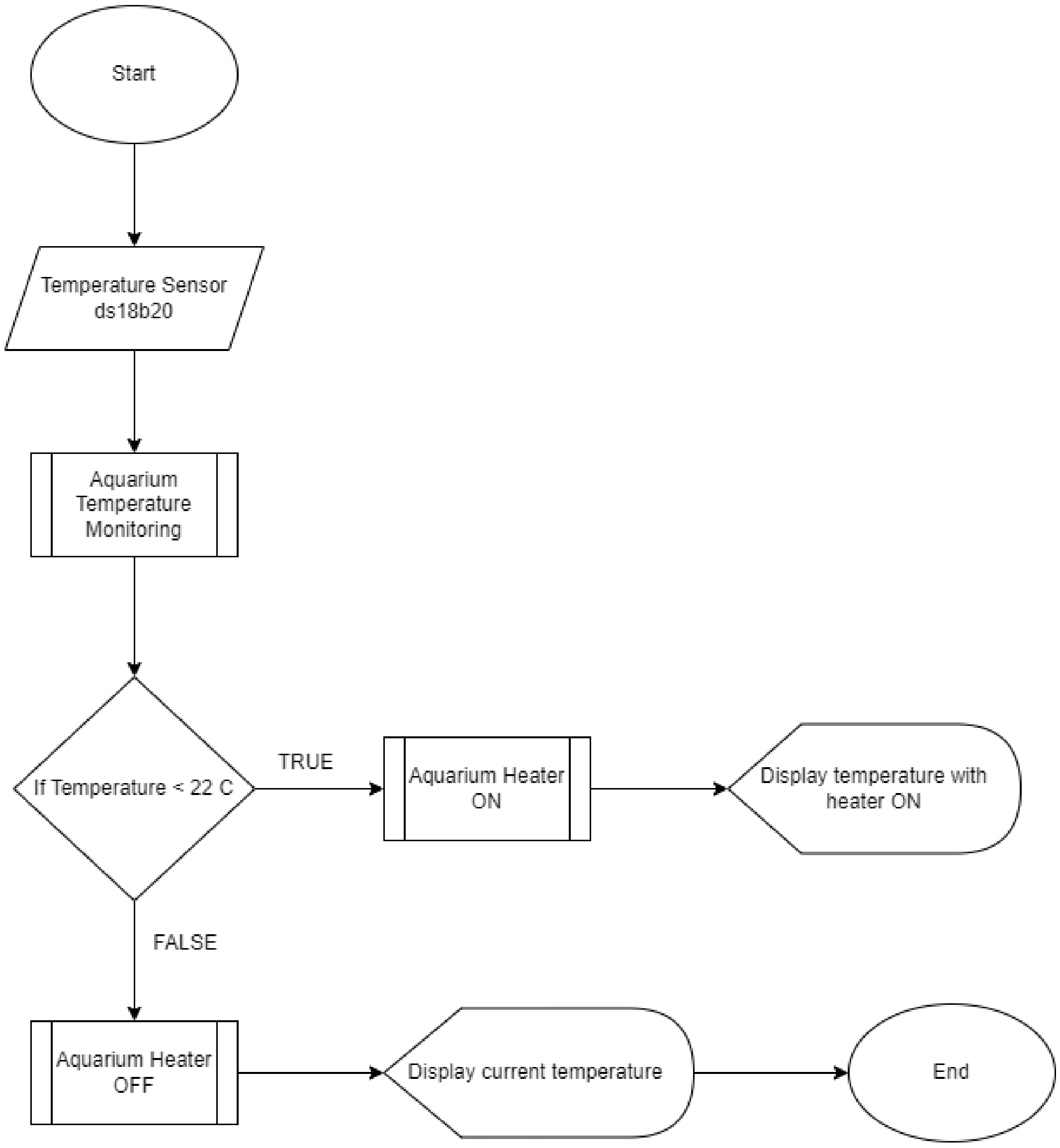
COMPONENT

4. Aquarium Heater

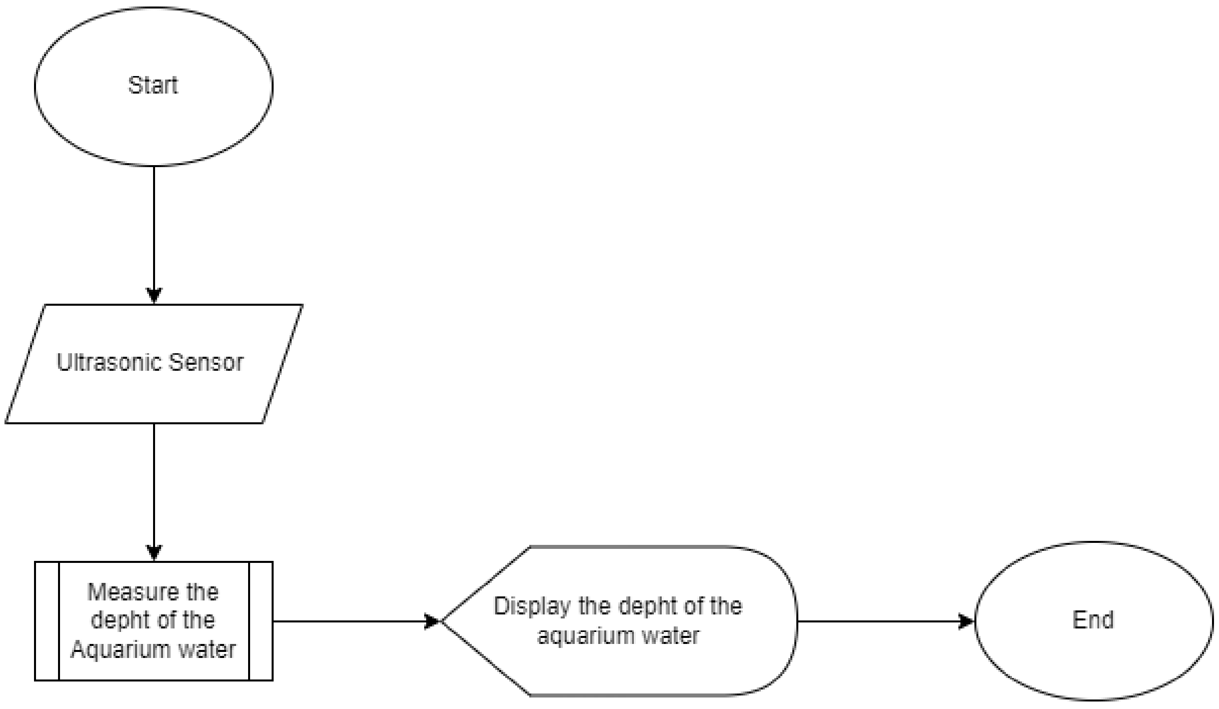
5. Relay

6. Jumper wire

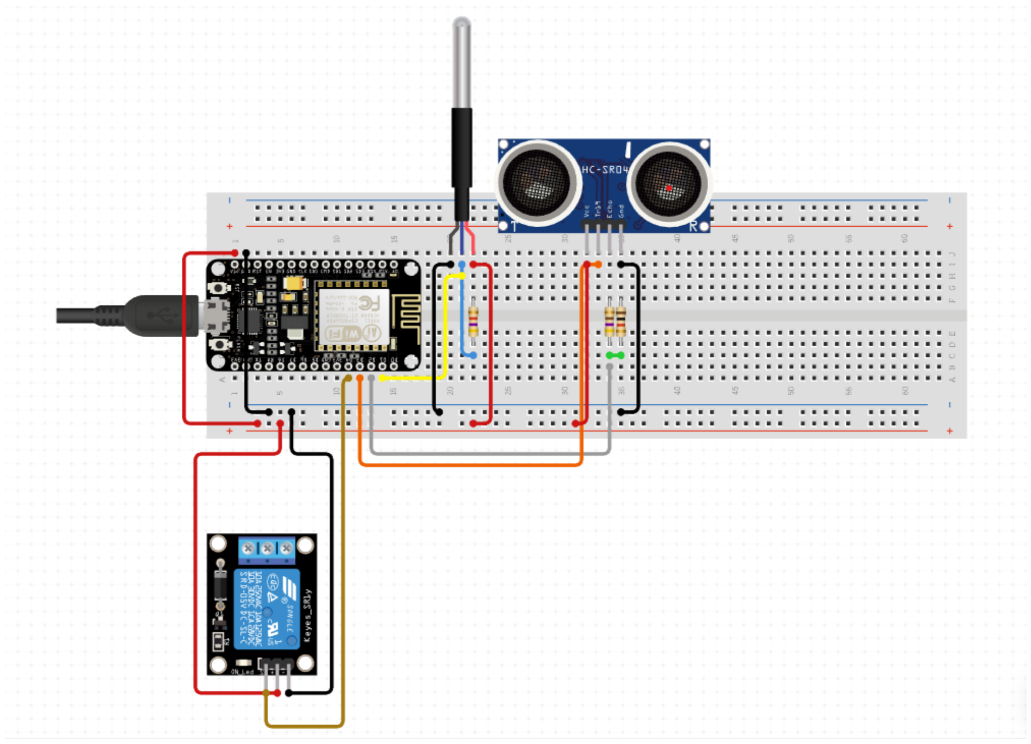
FLOWCHART


Start
Temperature Sensor ds18b20
Aquarium Temperature Monitoring
If Temperature < 22 C
TRUE
Aquarium Heater ON
Display temperature with heater ON
FALSE
Aquarium Heater OFF
Display current temperature
End

FLOWCHART


Start
Ultrasonic Sensor
Measure the depht of the Aquarium water
Display the depht of the aquarium water
End

SCHEMATIC DESIGN



SOURCE CODE

```
#include <OneWire.h>
#include <DallasTemperature.h>

#define SENSOR_PIN 5 // ESP32 pin GIOP21 connected to DS18B20 sensor's DQ
pin
#define RELAY_PIN 4 // ESP32 pin GIOP22 connected to relay's IN pin

//define sound velocity in cm/uS
#define SOUND_VELOCITY 0.034
#define CM_TO_INCH 0.393701



#include <ArduinoMqttClient.h>
#if defined(ARDUINO_SAMD_MKRWIFI1010) ||
defined(ARDUINO_SAMD_NANO_33_IOT) ||
defined(ARDUINO_AVR_UNO_WIFI_REV2)
  #include <WiFiNINA.h>
#elif defined(ARDUINO_SAMD_MKR1000)
  #include <WiFi101.h>
#elif defined(ARDUINO_ARCH_ESP8266)
  #include <ESP8266WiFi.h>
#elif defined(ARDUINO_ARCH_ESP32)
  #include <WiFi.h>
#endif
```

SOURCE CODE

```
char ssid[] = "ipok" ;   // your network SSID (name)
char pass[] = "apaluliat7"  ;  // your network password (use for WPA, or use as key for
WEP)

// To connect with SSL/TLS:
// 1) Change WiFiClient to WiFiSSLClient.
// 2) Change port value from 1883 to 8883.
// 3) Change broker value to a server with a known SSL/TLS root certificate
//    flashed in the WiFi module.

WiFiClient wifiClient;
MqttClient mqttClient(wifiClient);

const char broker[] = "test.mosquitto.org";
int        port     = 1883;
const char topic_suhu[]  = "home/suhu";
const char topic_jarak[] = "home/jarak";

const int trigPin = 12;
const int echoPin = 14;

OneWire oneWire(SENSOR_PIN);
DallasTemperature DS18B20(&oneWire);
```

SOURCE CODE

```
float tempC; // temperature in Celsius
float tempF; // temperature in Fahrenheit
long duration;
float distanceCm;
float distanceInch;


void ultrasonik() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);

  // Calculate the distance
  distanceCm = duration * SOUND_VELOCITY/2;

  // Convert to inches
  distanceInch = distanceCm * CM_TO_INCH;
```

SOURCE CODE

```
// Prints the distance on the Serial Monitor
  Serial.print("Distance (cm): ");
  Serial.println(distanceCm);
  Serial.print("Distance (inch): ");
  Serial.println(distanceInch);

  mqttClient.beginMessage(topic_jarak);
  mqttClient.print(distanceCm);
  mqttClient.endMessage();

  delay(3000);
}

void setup() {
    //Initialize serial and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }

  // attempt to connect to WiFi network:
  Serial.print("Attempting to connect to WPA SSID: ");
  Serial.println(ssid);
  while (WiFi.begin(ssid, pass) != WL_CONNECTED) {
    // failed, retry
    Serial.print(".");
    delay(5000);
  }
```

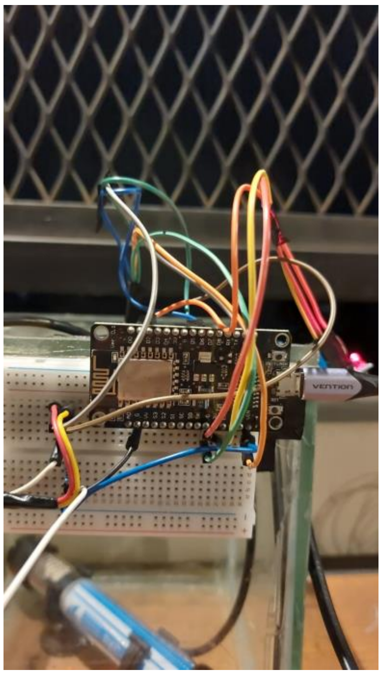
SOURCE CODE

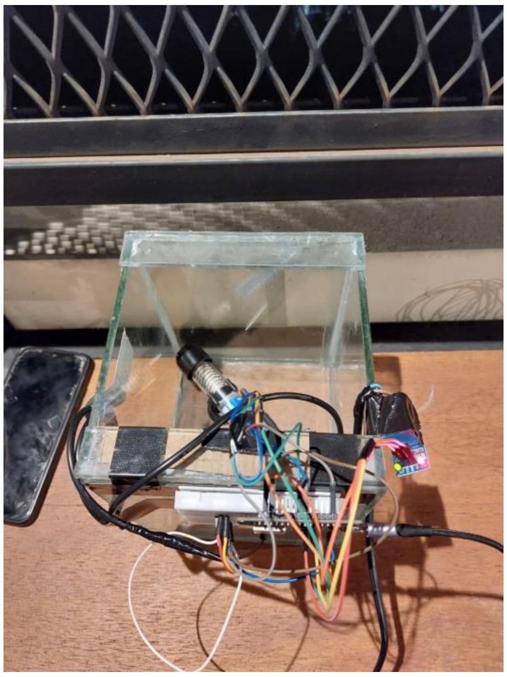
```
Serial.println("You're connected to the network");
  Serial.println();

  // You can provide a unique client ID, if not set the library uses Arduino-millis()
  // Each client must have a unique client ID
  // mqttClient.setId("clientId");

  // You can provide a username and password for authentication
  // mqttClient.setUsernamePassword("username", "password");

  Serial.print("Attempting to connect to the MQTT broker: ");
  Serial.println(broker);

 if (!mqttClient.connect(broker, port)) {
    Serial.print("MQTT connection failed! Error code = ");
    Serial.println(mqttClient.connectError());

    while (1);
  }

  Serial.println("You're connected to the MQTT broker!");
  Serial.println();

  DS18B20.begin();    // initialize the DS18B20 sensor
  pinMode(RELAY_PIN, OUTPUT); // set relay pin as output
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
}
```

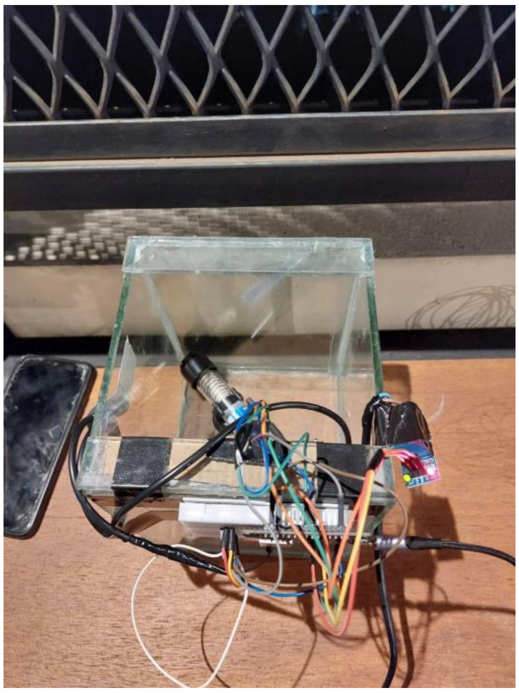
SOURCE CODE

```
void loop() {
 // call poll() regularly to allow the library to send MQTT keep alives which
 // avoids being disconnected by the broker
  mqttClient.poll();

  DS18B20.requestTemperatures();       // send the command to get temperatures
  tempC = DS18B20.getTempCByIndex(0);  // read temperature in °C
  tempF = tempC * 9 / 5 + 32; // convert °C to °F
  ultrasonik();

  Serial.print("Temperature: ");
  Serial.print(tempC);    // print the temperature in °C
  Serial.print("°C");
  Serial.print("  ~  ");  // separator between °C and °F
  Serial.print(tempF);    // print the temperature in °F
  Serial.println("°F");

  mqttClient.beginMessage(topic_suhu);
  mqttClient.print(tempC);
  mqttClient.endMessage();

  delay(500);
  if (tempC < 30) {
//      mqttClient.beginMessage(topic_suhu);
//      mqttClient.print("Heater ON");
//      mqttClient.endMessage();

      digitalWrite(RELAY_PIN, LOW); // turn on the relay
      Serial.println("Relay is ON");
      delay(10000);
```

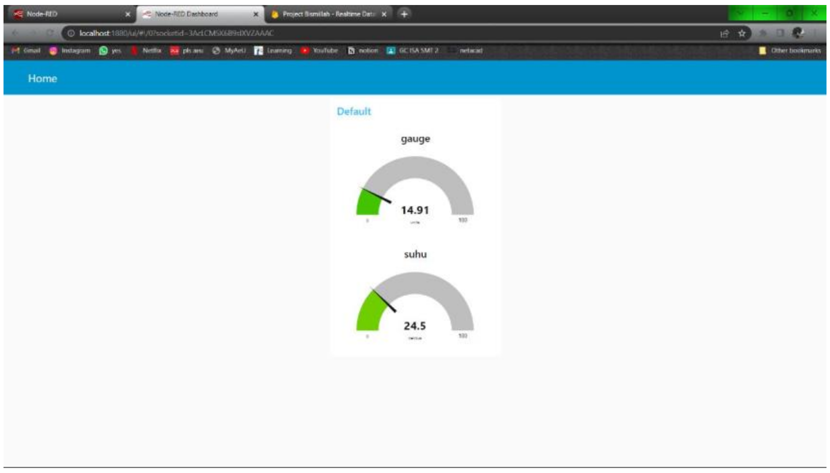
SCHEMATIC SYSTEM

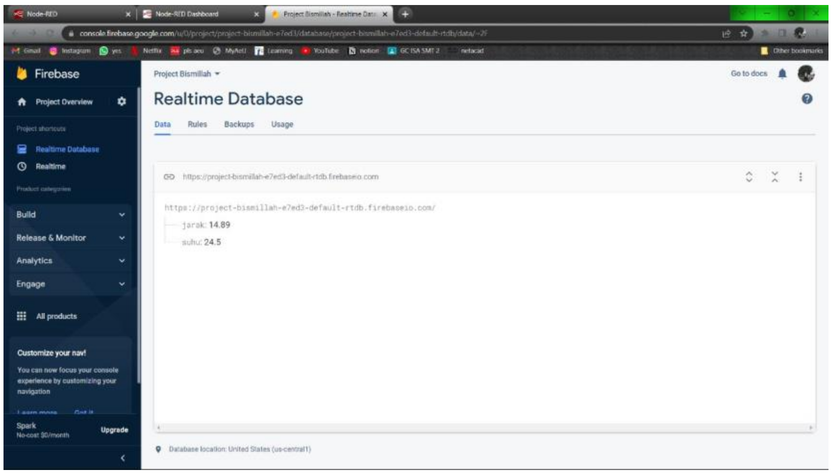






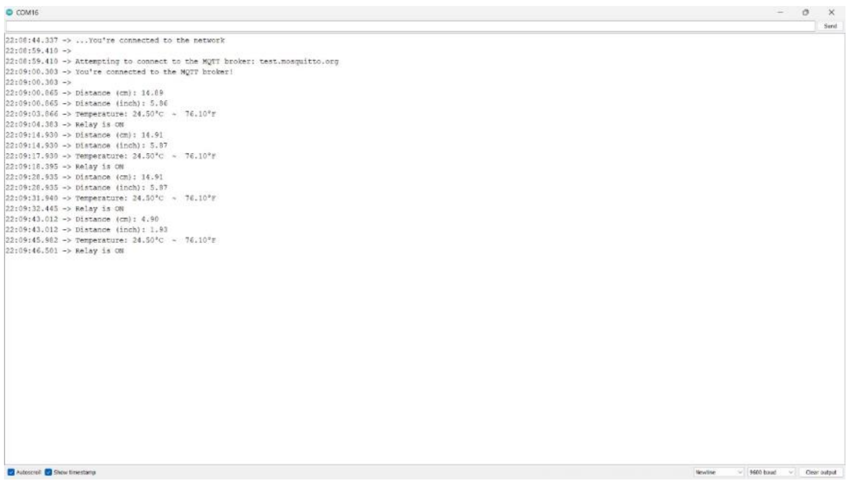
OUTPUT





https://project-bismillah-e7ed3-default-rtdb.firebaseio.com/
jarak: 14.89
suhu: 24.5

Database location: United States (us-central1)



```
22:08:44.337 -> ...You're connected to the network
22:08:59.410 ->
22:08:59.410 -> Attempting to connect to the MQTT broker: test.mosquitto.org
22:09:00.303 -> You're connected to the MQTT broker!
22:09:00.303 ->
22:09:00.865 -> Distance (cm): 14.89
22:09:00.865 -> Distance (inch): 5.86
22:09:03.866 -> Temperature: 24.50°C  ~  76.10°F
22:09:04.383 -> Relay is ON
22:09:14.930 -> Distance (cm): 14.91
22:09:14.930 -> Distance (inch): 5.87
22:09:17.930 -> Temperature: 24.50°C  ~  76.10°F
22:09:18.395 -> Relay is ON
22:09:28.935 -> Distance (cm): 14.91
22:09:28.935 -> Distance (inch): 5.87
22:09:31.940 -> Temperature: 24.50°C  ~  76.10°F
22:09:32.445 -> Relay is ON
22:09:43.012 -> Distance (cm): 4.90
22:09:43.012 -> Distance (inch): 1.93
22:09:45.982 -> Temperature: 24.50°C  ~  76.10°F
22:09:46.501 -> Relay is ON
```

CONFIGURATION

• Hardware	: AMD Ryzen 5 5500U, 16GB DDR4, 512GB SSD .
• Operating System	: Microsoft Windows 10 Home
• Software	: Red Node and Firebase