LAMPIRAN

1. Source code program untuk Box Controller

```
#include <WiFi.h>                //ESP8266
#include <PubSubClient.h>        //MQTT
#include <ArduinoJson.h>         //Parsing MQTT
#include <OneWire.h>             //DS18B20
#include <DallasTemperature.h>   //DS18B20
#include "DHT.h"                 //DHT22
#include <LiquidCrystal_I2C.h>   //LCD16x2
#include <EEPROM.h>              //EEPROM
#include <DFRobot_PH.h>          //pH


#define DHTTYPE DHT22            // pin sensor DHT22
#define PHPIN 34                //the esp gpio data pin number
#define TDSPIN 35              // pin sensor TDS
#define VREF 5.0                 // analog reference voltage(Volt) of the ADC
#define SCOUNT  30               // sum of sample point
#define ESPADC 4096.0            //the esp Analog Digital Convertion value
#define ESPVOLTAGE 5000          //voltage for ph

const char* ssid = "Nusanet Office";
const char* password = "1234554321";

const int mqttPort = 1883;
const char* mqttServer = "sibadaring.com";
const char* deviceID = "DS11X900";
const char* devicePasword = "";
const char* deviceUsername = "";

char message_buff[900];
const int pin_relay_nutrisi = 2; //FIX
const int pin_relay_air = 4; //FIX
const int pin_suhu_air = 32;
const int pin_suhu_kelembaban = 17;
int analogBuffer[SCOUNT];    // store the analog value in the array, read from ADC
int analogBufferTemp[SCOUNT];
int analogBufferIndex = 0,copyIndex = 0;
float averageVoltage=0, tdsValue = 0;
float temperature = 25;

//untuk sensor suhu air
OneWire oneWire(pin_suhu_air);
DallasTemperature sensors(&oneWire);

//untuk sensor suhu dan kelembaban
DHT dht(pin_suhu_kelembaban, DHTTYPE);

//untuk wifi client esp8266
```

```
WiFiClient espClient;
PubSubClient client(espClient);

//untuk sensor pH
DFRobot_PH ph;

//untuk LCD
LiquidCrystal_I2C lcd(0x27, 16, 2);


void callback(char* topic, byte* payload, unsigned int length) {
  String messageTemp;
  Serial.print("Message arrived in topic: ");
  Serial.println(topic);

  Serial.print("Message:");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
    messageTemp += (char)payload[i];
  }

  StaticJsonDocument<256> dataDevice;
  auto error = deserializeJson(dataDevice, messageTemp);
  Serial.println(messageTemp);
  if (error) {
    return;
  }


  Serial.println();
  Serial.println("-----------------------");
   if (String(topic) == "esp/test") {
      Serial.print("Changing output to ");
      if(dataDevice["led"] == "on"){
        Serial.println("on");
        digitalWrite(pin_relay_nutrisi, LOW);
        digitalWrite(pin_relay_air, HIGH);
      }
      else if(dataDevice["led"] == "off"){
        Serial.println("off");
        digitalWrite(pin_relay_nutrisi, HIGH);
        digitalWrite(pin_relay_air, LOW);
      }
    }
}

void setup(){
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  sensors.begin();
  dht.begin();
  lcd.init();
  lcd.backlight();
  pinMode (pin_relay_air, OUTPUT);
  pinMode (pin_relay_nutrisi, OUTPUT);
```

```
  pinMode(TDSPIN,INPUT);

  lcd.setCursor(0,0);
  lcd.print("Initializing : ");

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.println("Connecting to WiFi..");
    lcd.setCursor(0,1);
    lcd.print("Connecting to WiFi..");
    delay(1000);
  }
  Serial.println("Connected to the WiFi network");
  lcd.setCursor(0,1);
  lcd.print("WiFi Connected..");
  delay(1000);
  client.setServer(mqttServer, mqttPort);
  client.setCallback(callback);

  while (!client.connected()) {
    Serial.println("Connecting to MQTT...");
    lcd.setCursor(0,1);
    lcd.print("Connecting MQTT..");

    if (client.connect("ESP32Client")){
      Serial.println("connected");
      lcd.setCursor(0,1);
      lcd.print("MQTT Connected..");
      delay(1000);
    } else {

      Serial.print("failed with state ");
      Serial.print(client.state());
      lcd.setCursor(0,1);
      lcd.print("MQTT FailConnect");
      delay(2000);
    }
  }
  lcd.clear();
  client.subscribe("esp/test");

}

void loop(){
  client.loop(); //looping mqtt
  sensors.requestTemperatures();
  lcd.setCursor(0, 0);

  static unsigned long analogSampleTimepoint = millis();

  float suhu_lingkungan = dht.readTemperature();
  float kelembaban_lingkungan = dht.readHumidity();
  float suhu_air = sensors.getTempCByIndex(0);
  float nutrisi_air = 0;
  float ph = 0;
```

```
  if(millis()-analogSampleTimepoint > 40U){
    analogSampleTimepoint = millis();
    analogBuffer[analogBufferIndex] = analogRead(TDSPIN);
    analogBufferIndex++;
    if(analogBufferIndex == SCOUNT)
      analogBufferIndex = 0;
  }
  static unsigned long printTimepoint = millis();
  if(millis()-printTimepoint > 800U){
    printTimepoint = millis();
    for(copyIndex=0;copyIndex<SCOUNT;copyIndex++)
      analogBufferTemp[copyIndex]= analogBuffer[copyIndex];
    averageVoltage = getMedianNum(analogBufferTemp,SCOUNT) * (float)VREF / 4096.0;
    float compensationCoefficient=1.0+0.02*(temperature-25.0);
    float compensationVolatge=averageVoltage/compensationCoefficient;
    tdsValue=(133.42*compensationVolatge*compensationVolatge*compensationVolatge
-                255.86*compensationVolatge*compensationVolatge                +
857.39*compensationVolatge)*0.5; //convert voltage value to tds value

    //pengukuran pH
    int nilaiPengukuranPh = analogRead(PHPIN);
    double TeganganPh = 5 / 4095.0 * nilaiPengukuranPh;
    ph = 7.00 + ((3.593 - TeganganPh) / 0.381);

    Serial.print("TDS Value = "); Serial.print(tdsValue,0); Serial.println("ppm");
    Serial.print("pH = "); Serial.println(ph, 3);
    Serial.print("Suhu Lingkungan = ");Serial.println(suhu_lingkungan);
    Serial.print("Kelembaban                      Lingkungan                      =
");Serial.println(kelembaban_lingkungan);
    Serial.print("Suhu Air   = ");Serial.println(suhu_air);
    String tds_hasil = "TDS = "+String(tdsValue)+"ppm";
    String ph_hasil = "PH = "+String(ph);
    String humidity_hasil = "Humid = "+String(suhu_lingkungan)+"%";
    String suhu_udara_hasil = "Air Temp = "+String(suhu_lingkungan)+"C";
    String suhu_air_hasil = "Wtr Temp = "+String(suhu_air)+"C";


    lcd.setCursor(0,0);
    lcd.print(tds_hasil);
    lcd.setCursor(0,1);
    lcd.print(ph_hasil);
    delay(5000);
    lcd.setCursor(0,0);
    lcd.print(humidity_hasil);
    lcd.setCursor(0,1);
    lcd.print(suhu_udara_hasil);
    delay(5000);
    lcd.setCursor(0,0);
    lcd.print(suhu_air_hasil);
    delay(5000);
    lcd.clear();
  }
```

```
  String     pubString     =     "{'device':{'id':"+String(deviceID)+"},'sensors':
{'humid' :"+String(kelembaban_lingkungan)+",'wtemp':"+String(suhu_air)+",'atemp':
"+String(suhu_lingkungan)+",'ph':"+String(ph)+",'tds':"+String(tdsValue)+"}}";
  Serial.println(pubString);
  Serial.println(pubString.length());
  pubString.toCharArray(message_buff, pubString.length() + 1);
  client.publish("esp32/humidity", message_buff);

  delay(1000);
}

// Calculate TDS SENSOR
int getMedianNum(int bArray[], int iFilterLen){
  int bTab[iFilterLen];
  for (byte i = 0; i<iFilterLen; i++)
    bTab[i] = bArray[i];
    int i, j, bTemp;
    for (j = 0; j < iFilterLen - 1; j++) {
      for (i = 0; i < iFilterLen - j - 1; i++){
        if (bTab[i] > bTab[i + 1]){
          bTemp = bTab[i];
          bTab[i] = bTab[i + 1];
          bTab[i + 1] = bTemp;
        }
      }
    }
    if ((iFilterLen & 1) > 0)
      bTemp = bTab[(iFilterLen - 1) / 2];
    else
      bTemp = (bTab[iFilterLen / 2] + bTab[iFilterLen / 2 - 1]) / 2;
    return bTemp;}
```