LAMPIRAN

Lampiran 1. Program Gorong-gorong Otomatis

```
#include <SimpleTimer.h>
#include <NTPClient.h>
#include <ESP8266HTTPClient.h>
#include <DallasTemperature.h>
#include <GravityTDS.h>
#include <OneWire.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <WiFiUdp.h>

char auth[] = "fqMvEno5LtFrd_UcIpyyc6U03ehbgeQU";       // You should get
Auth Token in the Blynk App.                    // Your WiFi credentials.
char pass[] = "loranthus";

int DSPIN = D5; // Dallas Temperature Sensor

namespace pin
{
const byte tds_sensor = A0;   // TDS Sensor
}

namespace device
{
float aref = 3.3;
}

namespace sensor
{
float ec = 0;
```

```
unsigned int tds = 0;
float ecCalibration = 1;
}

//IP
//const char* host = "192.168.1.28";//KRISNA
//const char* host = "192.168.1.128";//DAYAT
//const char* host = "192.168.0.113";//KOS LOR
//const char* host = "192.168.43.25";//RUMAH
//const char* host = "192.168.0.126";//KOS LOR
//const char* host = "192.168.1.8";//KOS SUGLENG
//const char* host = "192.168.1.74";//RUMAH SATRIA
const char* host = "192.168.43.156";//RUMAH SATRIA

// WiFi
//const char* ssid = "Bocil 2"; // Enter your WiFi name
//const char* password = "sembarang";  // Enter WiFi password

//const char* ssid = "Kos Lor"; // Enter your WiFi name
//const char* password = "koskosan";  // Enter WiFi password

//const char* ssid = "EnemiesControl"; // Enter your WiFi name
//const char* password = "leboknosembarang";  // Enter WiFi password

//const char* ssid = "realme 3 Pro"; // Enter your WiFi name
//const char* password = "Whuyuare";  // Enter WiFi password

const char* ssid = "realme 5 Pro"; // Enter your WiFi name
const char* password = "Hello1234";  // Enter WiFi password

//const char* ssid = "Wifi Atas"; // Enter your WiFi name
//const char* ssid = "AjiTo-Net"; // Enter your WiFi name
```

```
//const char* password = "1sampai8";  // Enter WiFi password

//const char* ssid = "bismillah"; // Enter your WiFi name
//const char* password = "imenganteng";  // Enter WiFi password

//const char* ssid = "Wifi Id"; // Enter your WiFi name
//const char* password = "semangatbaru";  // Enter WiFi password

// MQTT Broker
const char* mqtt_broker = "maqiatto.com";
const char* topic = "zakariyaapw09@gmail.com/cobaMQTT";
const char* topic2 = "zakariyaapw09@gmail.com/suhu";
const char* topic3 = "zakariyaapw09@gmail.com/Fuzzy";
const char* topic4 = "zakariyaapw09@gmail.com/relayPipa";
const char* topic5 = "zakariyaapw09@gmail.com/relayPompa";
const char* topic6 = "zakariyaapw09@gmail.com/Manual";
const char* mqtt_username = "zakariyaapw09@gmail.com";
const char* mqtt_password = "Aji12345";
const int mqtt_port = 1883;
float fuzzy = 0;
float hasil = 0;
int pesanPipa = 0;
int pesanPompa = 0;
int pesanManualValue = 0;
String Status;
int cek = 0;

const long utcOffsetInSeconds = 25200;

#define relayPipa D6
#define relayPompa D7
```

```
SimpleTimer WaktuManual;

WiFiClient espClient;
PubSubClient client(espClient);
//DHT dht(DHTPIN, DHTTYPE);

WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "id.pool.ntp.org", utcOffsetInSeconds);

void setup() {
  // Set software serial baud to 115200;
  Serial.begin(115200);

  pinMode(relayPipa, OUTPUT);
  pinMode(relayPompa, OUTPUT);
  digitalWrite(relayPipa, HIGH);
  digitalWrite(relayPompa, HIGH);

  //  dht.begin();
  // connecting to a WiFi network
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.println("Connecting to WiFi..");
  }
  Serial.println("Connected to the WiFi network");
  //connecting to a mqtt broker
  client.setServer(mqtt_broker, mqtt_port);
  client.setCallback(callback);
  while (!client.connected()) {
    String client_id = "esp8266-client-";
    client_id += String(WiFi.macAddress());
```

```
    Serial.println("Connecting to public emqx mqtt broker.....");
    if (client.connect(client_id.c_str(), mqtt_username, mqtt_password)) {
      Serial.println("Public emqx mqtt broker connected");
    } else {
      Serial.print("failed with state ");
      Serial.print(client.state());
      delay(2000);
    }
  }
  //  publish and subscribe
  //  DHT.read11(sensor);
  //  int suhu = DHT.temperature;
  //  String suhu2 = String(suhu);
  //  client.publish(topic, suhu2.c_str());
  //  client.subscribe(topic);
  client.subscribe("zakariyaapw09@gmail.com/Fuzzy");
  client.subscribe("zakariyaapw09@gmail.com/relayPompa");
  client.subscribe("zakariyaapw09@gmail.com/relayPipa");
  client.subscribe("zakariyaapw09@gmail.com/Manual");
  client.subscribe("zakariyaapw09@gmail.com/statuspipa");
  timeClient.begin();
  WaktuManual.setInterval(2000);
}

void callback(char *topic, byte *payload, unsigned int length) {
  if (strcmp(topic, "zakariyaapw09@gmail.com/statuspipa") == 0) {
    for (int i = 0; i < length; i++) {
      Serial.print((char)payload[i]);
      //      Serial.println();
      payload[length] = '\0';
      fuzzy = atof((char* )payload);
    }
```

```
  Serial.println();
 }
 if (strcmp(topic, "zakariyaapw09@gmail.com/relayPipa") == 0) {
  for (int i = 0; i < length; i++) {
   Serial.print((char)payload[i]);
   //      Serial.println();
   payload[length] = '\0';
   pesanPipa = atof((char* )payload);
  }
  Serial.println();
 }
 if (strcmp(topic, "zakariyaapw09@gmail.com/relayPompa") == 0) {
  for (int i = 0; i < length; i++) {
   Serial.print((char)payload[i]);
   //      Serial.println();
   payload[length] = '\0';
   pesanPompa = atof((char* )payload);
  }
  Serial.println();
 }
 if (strcmp(topic, "zakariyaapw09@gmail.com/Manual") == 0) {
  for (int i = 0; i < length; i++) {
   Serial.print((char)payload[i]);
   //      Serial.println();
   payload[length] = '\0';
   pesanManualValue = atof((char* )payload);
  }
  Serial.println();
 }
 //  Serial.print("Message arrived in topic: ");
 //  Serial.println(topic);
 //  Serial.print("Message:");
```

```
  //  for (int i = 0; i < length; i++) {
  //    Serial.print((char) payload[i]);
  //    payload[length] = '\0';
  //    fuzzy = atof((char* )payload);
  //  }
  //  Serial.println();
  //  Serial.println("-----------------------");
}
float temp = 0.0;



void loop() {
  client.loop();
  if (WaktuManual.isReady()) {
    if (pesanManualValue == 1) {
      if (pesanPipa == 1) {
        digitalWrite(relayPipa, LOW);
      } else {
        digitalWrite(relayPipa, HIGH);
      }

      if (pesanPompa == 1) {
        digitalWrite(relayPompa, LOW);
      } else {
        digitalWrite(relayPompa, HIGH);
      }
    } else {
      timeClient.update();
      hasil = fuzzy;
      double waterTemp = TempRead();
      waterTemp  = waterTemp * 0.0625; // conversion accuracy is 0.0625 / LSB
```

```
    float rawEc = analogRead(pin::tds_sensor) * device::aref / 1024.0; // read the analog value more stable by the median filtering algorithm, and convert to voltage value
    float temperatureCoefficient = 1.0 + 0.02 * (waterTemp - 25.0); // temperature compensation formula: fFinalResult(25^C) = fFinalResult(current)/(1.0+0.02*(fTP-25.0));

    sensor::ec = (rawEc / temperatureCoefficient) * sensor::ecCalibration; // temperature and calibration compensation
    sensor::tds = (133.42 * pow(sensor::ec, 3) - 255.86 * sensor::ec * sensor::ec + 857.39 * sensor::ec) * 0.5; //convert voltage value to tds value
    int tdsvalue = (sensor::tds / 2.5);




    Serial.print(F("TDS:")); Serial.println(sensor::tds);
    Serial.print(F("EC:")); Serial.println(sensor::ec, 2);
    Serial.print(F("Temperature:")); Serial.println(waterTemp, 2);
    Serial.println(F(""));
    Serial.print("Hasil Fuzzy :");
    Serial.println(hasil);

    int AktuatorKeluar = ((int)hasil * 3);
    int AktuatorMasuk = (((int)AktuatorKeluar * 2)+14);
    Serial.print(timeClient.getHours());
    Serial.print(":");
    Serial.print(timeClient.getMinutes());
    Serial.print(":");
    Serial.println(timeClient.getSeconds());
    Serial.println(AktuatorKeluar);
    Serial.println(AktuatorMasuk);
```

```
    if (hasil < 2) {
      digitalWrite(relayPipa, HIGH);
      digitalWrite(relayPompa, HIGH);
    } else {
      // Air Keluar
      if (timeClient.getMinutes() == 49 && timeClient.getSeconds() == 0) {
        digitalWrite(relayPompa, LOW);
        Serial.println(timeClient.getMinutes());
        Serial.println(timeClient.getSeconds());
        Serial.println("Pipa Terbuka");
        Serial.println(" ");
      } else if (timeClient.getMinutes() == 49 && timeClient.getSeconds() ==
AktuatorKeluar) {
        digitalWrite(relayPompa, HIGH);
        Serial.println(timeClient.getMinutes());
        Serial.println(timeClient.getSeconds());
        Serial.println("Pipa Mati");
        Serial.println(" ");
      }

      // Air Masuk
      if (timeClient.getMinutes() == 49 && timeClient.getSeconds() ==
AktuatorKeluar) {
        digitalWrite(relayPipa, LOW);
        Serial.println(timeClient.getMinutes());
        Serial.println(timeClient.getSeconds());
        Serial.println("Pompa Terbuka");
        Serial.println(" ");
      } else if (timeClient.getMinutes() == 49 && timeClient.getSeconds() ==
AktuatorMasuk) {
        digitalWrite(relayPipa, HIGH);
```

```
        Serial.println(timeClient.getMinutes());
        Serial.println(timeClient.getSeconds());
        Serial.println("Pompa Mati");
        Serial.println(" ");
      }
    }

    // Air Keluar
//    if (timeClient.getMinutes() == 38 && timeClient.getSeconds() == 0) {
//      digitalWrite(relayPipa, LOW);
//      Serial.println(timeClient.getMinutes());
//      Serial.println(timeClient.getSeconds());
//      Serial.println("Pipa Terbuka");
//      Serial.println(" ");
//      } else if (timeClient.getMinutes() == 38 && timeClient.getSeconds() ==
AktuatorKeluar) {
//      digitalWrite(relayPipa, HIGH);
//      Serial.println(timeClient.getMinutes());
//      Serial.println(timeClient.getSeconds());
//      Serial.println("Pipa Mati");
//      Serial.println(" ");
//    }
//
//    // Air Masuk
//          if (timeClient.getMinutes() == 38 && timeClient.getSeconds() ==
AktuatorKeluar) {
//      digitalWrite(relayPompa, LOW);
//      Serial.println(timeClient.getMinutes());
//      Serial.println(timeClient.getSeconds());
//      Serial.println("Pompa Terbuka");
//      Serial.println(" ");
```

```
//      } else if (timeClient.getMinutes() == 38 && timeClient.getSeconds() ==
AktuatorMasuk) {
//        digitalWrite(relayPompa, HIGH);
//        Serial.println(timeClient.getMinutes());
//        Serial.println(timeClient.getSeconds());
//        Serial.println("Pompa Mati");
//        Serial.println(" ");
//      }

     if (timeClient.getMinutes() == 35 && timeClient.getSeconds() == 0) {
       if (hasil <= 1) {
         cek = 1;
       } else if ( hasil <= 3 ) {
         cek = 2;
       } else if ( hasil > 3 || hasil <= 7) {
         cek = 3;
       };

//        String Link;
//        HTTPClient http;
//                                                                  Link          =
"http://192.168.43.156/app_kolam_lele/admin/Simpan/sensor?suhu="          +
String(waterTemp) + "&kepadatan=" + String(tdsvalue) + "&fuzzy=" +
String(hasil) + "&Status=" + String(cek);
//        http.begin(Link);
//        //mode
//        http.GET();
//        http.end();
//        Serial.println(timeClient.getMinutes());
//        Serial.println(timeClient.getSeconds());
//        Serial.println("Database Masuk");
//        Serial.println(" ");
```

```
    } else if (timeClient.getMinutes() == 24 && timeClient.getSeconds() == 4)
{


    }
    //  String Link;
    //  HTTPClient http;
    //                                                  Link                    =
"http://192.168.0.113/app_kolam_lele/admin/Simpan/sensor?suhu="          +
String(waterTemp) + "&kepadatan=" + String(sensor::tds) + "&fuzzy=" +
String(hasil);
    //  http.begin(Link);
    //  //mode
    //  http.GET();
    //  http.end();

    String suhu2 = String(waterTemp);
    String tds2 = String(tdsvalue);
    //client.subscribe("zakariyaapw09@gmail.com/Fuzzy");
    client.publish(topic, suhu2.c_str());
    client.publish(topic2, tds2.c_str());
    delay(1000);
   }
  }
}


boolean DS18B20_Init()
{
  pinMode(DSPIN, OUTPUT);
  digitalWrite(DSPIN, HIGH);
  delayMicroseconds(5);
  digitalWrite(DSPIN, LOW);
  delayMicroseconds(750);//480-960
```

```
  digitalWrite(DSPIN, HIGH);
  pinMode(DSPIN, INPUT);
  int t = 0;
  while (digitalRead(DSPIN))
  {
    t++;
    if (t > 60) return false;
    delayMicroseconds(1);
  }
  t = 480 - t;
  pinMode(DSPIN, OUTPUT);
  delayMicroseconds(t);
  digitalWrite(DSPIN, HIGH);
  return true;
}

void DS18B20_Write(byte data)
{
  pinMode(DSPIN, OUTPUT);
  for (int i = 0; i < 8; i++)
  {
    digitalWrite(DSPIN, LOW);
    delayMicroseconds(10);
    if (data & 1) digitalWrite(DSPIN, HIGH);
    else digitalWrite(DSPIN, LOW);
    data >>= 1;
    delayMicroseconds(50);
    digitalWrite(DSPIN, HIGH);
  }
}

byte DS18B20_Read()
```

```
{
  pinMode(DSPIN, OUTPUT);
  digitalWrite(DSPIN, HIGH);
  delayMicroseconds(2);
  byte data = 0;
  for (int i = 0; i < 8; i++)
  {
    digitalWrite(DSPIN, LOW);
    delayMicroseconds(1);
    digitalWrite(DSPIN, HIGH);
    pinMode(DSPIN, INPUT);
    delayMicroseconds(5);
    data >>= 1;
    if (digitalRead(DSPIN)) data |= 0x80;
    delayMicroseconds(55);
    pinMode(DSPIN, OUTPUT);
    digitalWrite(DSPIN, HIGH);
  }
  return data;
}

int TempRead()
{
  if (!DS18B20_Init()) return 0;
  DS18B20_Write (0xCC); // Send skip ROM command
  DS18B20_Write (0x44); // Send reading start conversion command
  if (!DS18B20_Init()) return 0;
  DS18B20_Write (0xCC); // Send skip ROM command
  DS18B20_Write (0xBE); // Read the register, a total of nine bytes, the first two
bytes are the conversion value
  int waterTemp = DS18B20_Read (); // Low byte
  waterTemp |= DS18B20_Read () << 8; // High byte
```

```
  return waterTemp;
}
```

Lampiran 2. Fuzzy Sugeno

```
var bkCepat = 5;
var bkLama = 7;
var Tutup = 0;

var suhu = 0, tds = 0;
var Pipa, OutPipa = 0;
var average = 0;

var tdkKeruh, tdsNormal, Keruh;
var shDingin, shNormal, shPanas;

var rule1, rule2, rule3, rule4, rule5, rule6, rule7, rule8, rule9;

// var messagePayloadSuhu = 0;
// var messagePayloadKekeruhan = 0;
// var messagePayloadFuzzy = 0;
// var messagePayloadPakan = 0;
// var suhuAir = 0;
// var tdsAir = 0;



// // Called after form input is processed
// function startConnect() {
//   // Generate a random client ID
//   clientID = "krisna123";

//   // Fetch the hostname/IP address and port number from the form
//   host = "maqiatto.com";
```

```
//   port = 8883;

//   // Print output for the user in the messages div
//                //      document.getElementById("pesansuhu").innerHTML      += '<span>Connecting to: ' + host + ' on port: ' + port + '</span><br/>';
//   // document.getElementById("pesansuhu").innerHTML += '<span>Using the following client value: ' + clientID + '</span><br/>';

//   // Initialize new Paho client connection
//   client = new Paho.MQTT.Client(host, Number(port), "Aji");

//   // Connect the client, if successful, call onConnect function
//   client.connect({
//       userName: "zakariyaapw09@gmail.com",
//       password: "Aji12345",
//       onSuccess: onConnect,
//   });

//   // Set callback handlers
//   client.onConnectionLost = onConnectionLost;
//   client.onMessageArrived = onMessageArrived;
// //   client.ledState = ledState;
// }

// // Called when the client connects
// function onConnect() {
//   // Fetch the MQTT topic from the form
//   topic = "zakariyaapw09@gmail.com/cobaMQTT";
//   topicKekeruhan = "zakariyaapw09@gmail.com/suhu";
//   topicFuzzy = "zakariyaapw09@gmail.com/Fuzzy";
//   topicPakan = "zakariyaapw09@gmail.com/pakan";
```

```
//   // Print output for the user in the messages div
//                 //      document.getElementById("pesansuhu").innerHTML      +=
'<span>Subscribing to: ' + topic + '</span><br/>';

//   // Subscribe to the requested topic
//   client.subscribe(topic);
//   client.subscribe(topicKekeruhan);
//   client.subscribe(topicFuzzy);
//   client.subscribe(topicPakan);
// }

// // Called when the client loses its connection
// function onConnectionLost(responseObject) {
//   console.log("onConnectionLost: Connection Lost");
//   if (responseObject.errorCode !== 0) {
//      console.log("onConnectionLost: " + responseObject.errorMessage);
//   }
// }

// var suhu = "";
// var kekeruhan = "";

// // Called when a message arrives
// function onMessageArrived(message) {
//   if (message.destinationName == topic) {
//         console.log("onMessageArrived: " + message.payloadString);
//         document.getElementById("pesansuhu").innerHTML   =   '<span>'   +
message.payloadString + ' Celcius</span>';
//         messagePayloadSuhu = parseInt(message.payloadString);
//         console.log("Suhu "+messagePayloadSuhu);
//   }if (message.destinationName == topicKekeruhan) {
```

```
//         console.log("onMessageArrived: " + message.payloadString);
//         document.getElementById("tdsair").innerHTML = '<span>' +
message.payloadString + ' ppm</span>';
//         messagePayloadKekeruhan = parseInt(message.payloadString);
//         console.log("Kekeruhan "+messagePayloadKekeruhan);
//   }if (message.destinationName == topicFuzzy) {
//         console.log("onMessageArrived: " + message.payloadString);
//         document.getElementById("fuzzy").innerHTML = '<span>' +
message.payloadString + ' Fuzzy</span>';
//         messagePayloadFuzzy = parseInt(message.payloadString);
//         console.log("Fuzzy "+messagePayloadFuzzy);
//   }if (message.destinationName == topicPakan) {
//         console.log("onMessageArrived: " + message.payloadString);
//         document.getElementById("pakan").innerHTML = '<span>' +
message.payloadString + ' ||</span>';
//         messagePayloadPakan = parseInt(message.payloadString);
//         console.log("pakan "+messagePayloadPakan);
//   }

//   suhuAir = messagePayloadSuhu;
//   tdsAir = messagePayloadKekeruhan;
// }

// Fuzzyfikasi Suhu
function suhuDingin() {
   if (suhu <= 23) {
      shDingin = 1;
    }
    else if (suhu >= 23 && suhu <= 27) {
      shDingin = (27 - suhu) / (27 - 23);
    }
    else if (suhu > 27) {
```

```
      shDingin = 0;
    }
    return shDingin;
}
function suhuNormal() {
  if (suhu < 23 || suhu > 33 ) {
      shNormal = 0;
    }
    else if (suhu >= 23 && suhu <= 27) {
      shNormal = (suhu - 23) / (27 - 23);
    }
    else if (suhu > 27 && suhu <= 33) {
      shNormal = (33 - suhu) / (33 - 27);
    }
    return shNormal;
}
function suhuPanas() {
  if (suhu <= 27 ) {
      shPanas = 0;
    }
    else if (suhu > 27 && suhu <= 33) {
      shPanas = (suhu - 27) / (33 - 27);
    }
    else if (suhu > 33) {
      shPanas = 1;
    }
    return shPanas;
}
// Fuzzyfikasi Kekeruhan
function TidakKeruh() {
  if (tds <= 400 ) {
      tdkKeruh = 1;
```

```
    }
    else if (tds > 400 && tds <= 550) {
      tdkKeruh = (550 - tds) / (550 - 400);
    }
    else if (tds > 550) {
      tdkKeruh = 0;
    }
    return tdkKeruh;
}
function Normal() {
   if (tds < 400 || tds > 700) {
      tdsNormal = 0;
    }
    else if (tds >= 400 && tds  < 550) {
      tdsNormal = (tds - 400) / (550 - 400);
    }
    else if (tds >= 550 && tds <= 700) {
      tdsNormal = (700 - tds) / (700 - 550);
    }
    return tdsNormal;
}
function tdsKeruh() {
   if (tds <= 600) {
      Keruh = 0;
    }
    else if (tds > 600 && tds <= 750) {
      Keruh = (tds - 600) / (750 - 600);
    }
    else if (tds > 750) {
      Keruh = 1;
    }
    return Keruh;
```

```
}

function fuzzifikasi() {
        suhuDingin();
        suhuNormal();
        suhuPanas();
        TidakKeruh();
        Normal();
        tdsKeruh();
}

function fuzzy_rule_pipa() {
        var jml_rule = [];
        var SumA = 0;
        fuzzifikasi();
        if (shDingin >= 0 && tdkKeruh >= 0) {
                rule1 = Math.min(shDingin, tdkKeruh);
                jml_rule[0] = rule1;
        }
        if (shDingin >= 0 && tdsNormal >= 0) {
                rule2 = Math.min(shDingin, tdsNormal);
                jml_rule[1] = rule2;
        }
        if (shDingin >= 0 && Keruh >= 0) {
                rule3 = Math.min(shDingin, Keruh);
                jml_rule[2] = rule3;
        }
        if (shNormal >= 0 && tdkKeruh >= 0) {
                rule4 = Math.min(shNormal, tdkKeruh);
                jml_rule[3] = rule4;
        }
        if (shNormal >= 0 && tdsNormal >= 0) {
```

```
            rule5 = Math.min(shNormal, tdsNormal);
            jml_rule[4] = rule5;
        }
        if (shNormal >= 0 && Keruh >= 0) {
            rule6 = Math.min(shNormal, Keruh);
            jml_rule[5] = rule6;
        }
        if (shPanas >= 0 && tdkKeruh >= 0) {
            rule7 = Math.min(shPanas, tdkKeruh);
            jml_rule[6] = rule7;
        }
        if (shPanas >= 0 && tdsNormal >= 0) {
            rule8 = Math.min(shPanas, tdsNormal);
            jml_rule[7] = rule8;
        }
        if (shPanas >= 0 && Keruh >= 0) {
            rule9 = Math.min(shPanas, Keruh);
            jml_rule[8] = rule9;
        }

        //defuzifikasi
        var weight =
            rule1 * bkCepat +
            rule2 * bkCepat +
            rule3 * bkLama +
            rule4 * Tutup +
            rule5 * Tutup +
            rule6 * bkLama +
            rule7 * bkCepat +
            rule8 * bkCepat +
            rule9 * bkLama;
        average =
```

```
                    jml_rule[0] +
                    jml_rule[1] +
                    jml_rule[2] +
                    jml_rule[3] +
                    jml_rule[4] +
                    jml_rule[5] +
                    jml_rule[6] +
                    jml_rule[7] +
                    jml_rule[8];
        OutPipa = weight / average;
}

function startFuzzy() {
        suhu = suhuAir;
        tds = tdsAir;
   // suhu = 32;
   // tds = 1;
   fuzzy_rule_pipa();
   document.getElementById("pipa").innerHTML                    =
parseFloat(OutPipa).toFixed(2);
        console.log(OutPipa);
        console.log(suhu);
        console.log(tds);
        console.log(suhuDingin());
        console.log(suhuNormal());
        console.log(suhuPanas());
        console.log(TidakKeruh());
        console.log(Normal());
        console.log(tdsKeruh());
}

// function Publish(){
```

```
//          pipaMessage = new Paho.MQTT.Message(OutPipa.toString());
//          pipaMessage.destinationName                                    =
"zakariyaapw09@gmail.com/FuzzyWeb";
//          client.send(pipaMessage);
// }

setInterval(startFuzzy,1000);
// setInterval(Publish,2000);
```