

**LAPORAN PROYEK INKUBATOR TELUR
SISTEM PENGATURAN TERTANAM**



Dosen:

Mohamad Abdul Hady, S.T., M.T.

Mahasiswa:

Muhammad Faris Zuhairi – 07111940000164

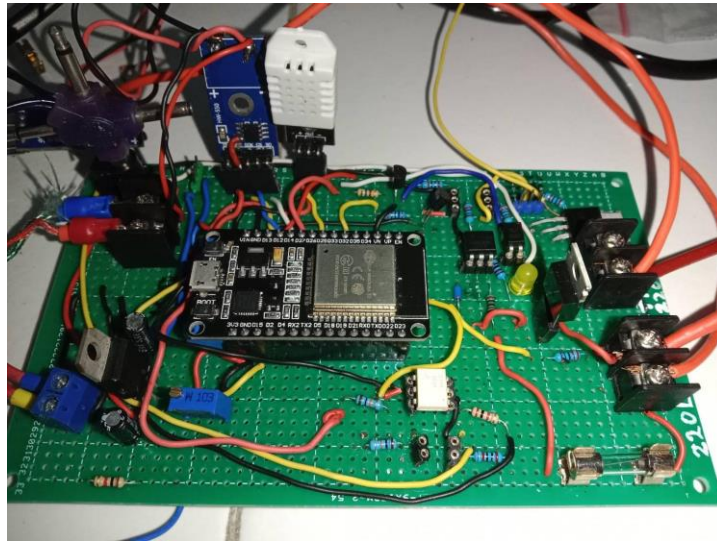
David Percival – 5022201242

**DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI ELEKTRO DAN INFORMATIKA CERDAS
INSTITUT TEKNOLOGI SEPULUH NOPEMBER SURABAYA
2022**

PROYEK INKUBATOR TELUR

Judul Proyek: Inkubator Telur menggunakan Kontroller PID Suhu dan IoT

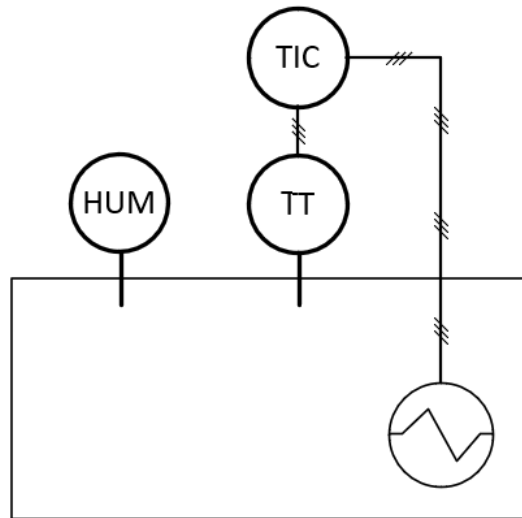
Secara garis besar sistem yang sudah dibuat tersusun atas sistem sourcing, sistem sensor, dan ESP32 beserta program di dalamnya. Berikut merupakan PCB yang telah dibuat.



Gambar 1. PCB dan komponennya.

PID Controller

PID (Proportional Integral Derivative) controller adalah kontrol yang paling sering digunakan pada aplikasi kontrol industri dengan menggunakan element feedback close-loop untuk meregulasikan dan mengontrol sistem untuk beberapa variabel proses. Seperti temperature, jalan angin, tekanan , kecepatan dan variabel proses. PID akan membandingkan output dari sistem dengan set point yang ada dan menghasilkan error signal dan akan melakukan pengontrolan output untuk minimalisir error dan menjalankan sistem secara efisien.

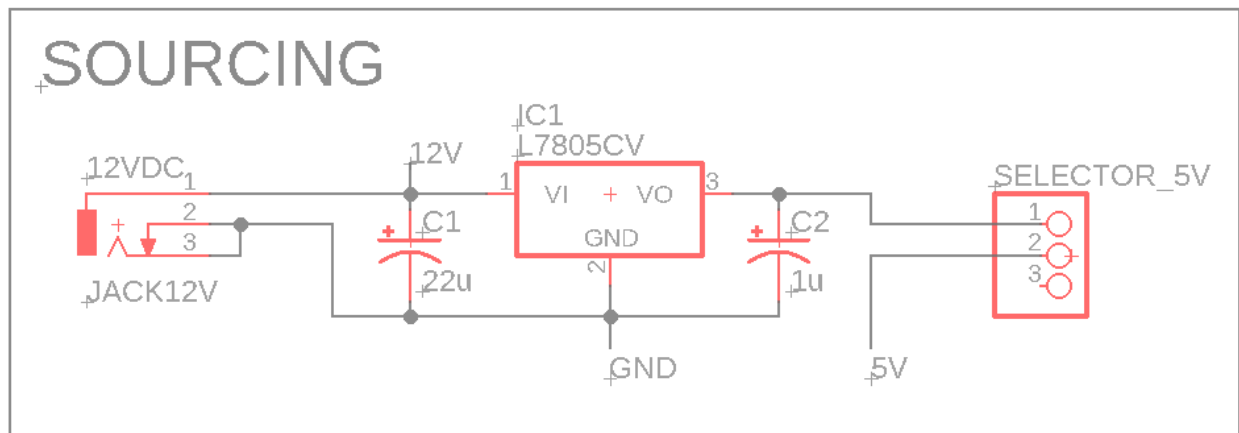


Gambar 2. Instrumentation diagram inkubator telur

1. Rangkaian Sourcing

Rangkaian sourcing merupakan rangkaian yang memberikan suplai 12VDC dan 5VDC ke rangkaian utama.

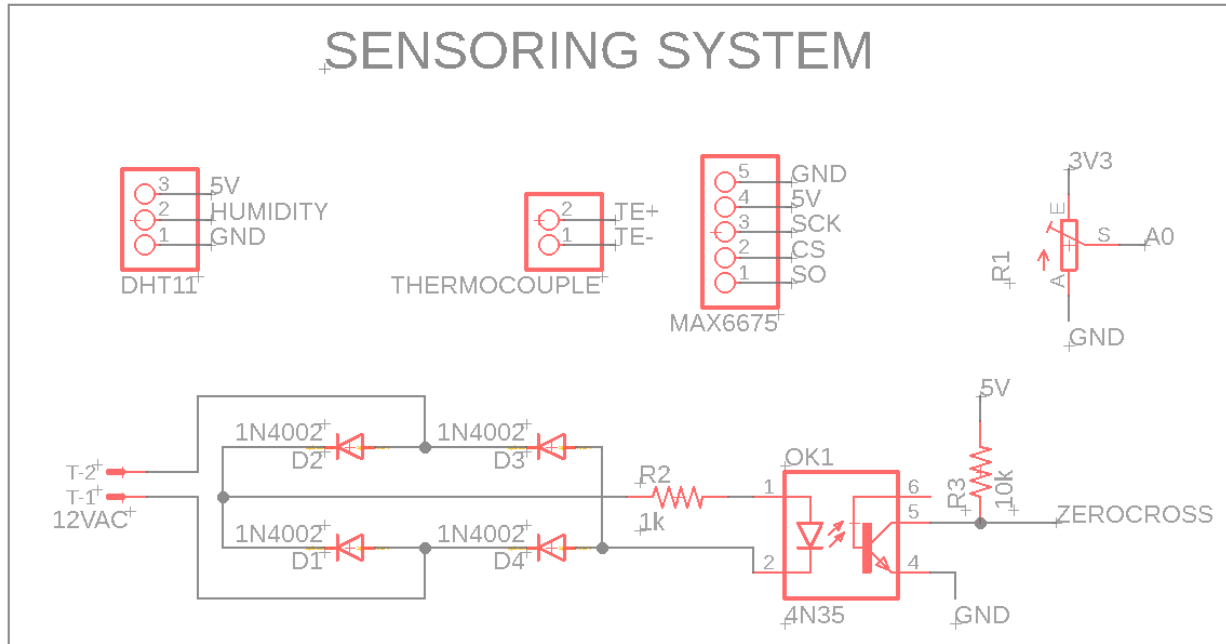
- Tegangan 12VDC digunakan untuk mentrigger optocoupler PC817-MOC3021 karena memiliki tegangan forward V_f yang mencukupi untuk memberikan arus forward I_f ke MOC3021.
- Tegangan 5VDC dihasilkan oleh voltage regulator L7805CV melalui kapasitor coupling 1uF dan 22uF. Header 3pin digunakan sebagai selector agar user dapat memilih tegangan yang dipakai tegangan dari jack 12VDC atau USB micro ESP32, agar closed loop single loop hukum kirchoff terpenuhi.



Gambar 3. Rangkaian sourcing menggunakan voltage regulator dan selector.

2. Rangkaian sensor

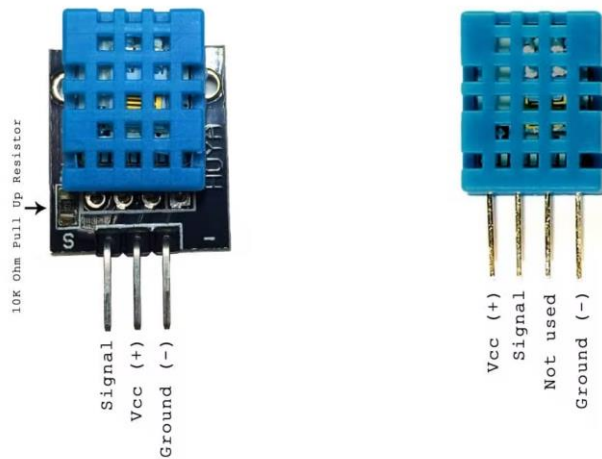
Rangkaian sensor tersusun atas 3 rangkaian, yakni sensor humidity DHT11, thermocouple tipe K, dan AC zero crossing detector.



Gambar 4. Rangkaian sensor humidity, temperature, dan zero-crossing detector.

2.1. DHT11

DHT11 adalah sensor tertanam yang digunakan untuk mengukur suhu dan humiditas dari lingkungan sekitar dan diberikan output dalam bentuk digital dimana dht11 dapat mengukur temperatur ruang dari 0°C- 50°C dengan $\pm 2^\circ\text{C}$ accuracy. Dengan humiditas dari 20%-80% dengan akurasi ± 5 . Dimana DHT11 dengan harga yang relatif murah dan mudah didapatkan dengan sampling rate 1 Hz.



Gambar 5. Pinout sensor DHT11

Inisialisasi pembacaan DHT11

```
#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

dht.begin();
```

Mengambil data humidity

```
float hum = dht.readHumidity(); //DHT11
```

2.2. K-type Termokopel

Termokopel adalah alat yang terdiri dari dua jenis konduktor listrik yang membentuk penghubung termal dipenghubung. Termokopel memiliki tipe berdasarkan range voltage, biaya dan sensitivitas. K-type Termokopel adalah salah satu termokopel yang terbuat dari konduktor chrome dan alumel yang bisa membaca temperatur dari range -200 sampai 1260°C. Termokopel tipe k tidak terdapat resistansi spesifik sehingga resistansi dari termokopel tipe k dapat bervariasi dari besar konduktor dan seberapa banyak konduktor tersebut digunakan.



Gambar 6. Modul penguat MAX6675

Termokopel ini membutuhkan modul penguatan MAX6675 agar sinyal thermocouple yang sangat kecil dapat terbaca microcontroller. MAX6675 adalah amplifier yang berfungsi untuk membantu memasukan output temperatur dari termokopel kedalam rangkaian listrik.. Output dari termokopel amplifier tergantung dari pembacaan voltase dari reference junction. MAX6675 terdapat sensor temperatur untuk mengukur reference junction dan menguatkan voltase kecil pada reference junction sehingga dapat terbaca oleh mikrocontroller. Pada umumnya max6675 sudah terdapat pada k-type termokopel.

Untuk mengambil data temperature oleh mikrocontroller, dibutuhkan library MAX6675. Inisialisasi dari program diberikan sebagai berikut.

```
#include <Thermocouple.h>
#include <MAX6675_Thermocouple.h>
#include <SmoothThermocouple.h>
#define SCK_PIN 13
#define CS_PIN 12
#define SO_PIN 14
#define SMOOTHING_FACTOR 5
Thermocouple* thermocouple = NULL;
Thermocouple* originThermocouple = new MAX6675_Thermocouple(SCK_PIN, CS_PIN,
SO_PIN);
thermocouple = new SmoothThermocouple(originThermocouple, SMOOTHING_FACTOR);
```

Pada looping, variabel temperature harus terus diupdate. Linearisasi thermocouple diberikan dengan memberikan gain feedback span-zero yang didefinisikan sebagai berikut.

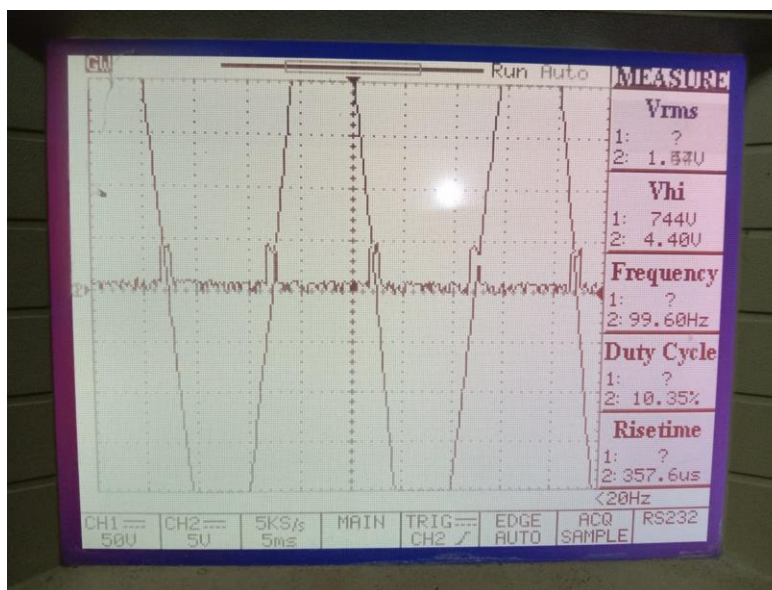
```
float y_span = 1;  
float y_zero = 0;  
float yTT = 0;
```

Lalu pembacaan temperature akan disubstitusikan ke $y=mx+c$ dengan asumsi nilai span zero ini ialah hasil kalibrasi.

```
//-----Feedback Thermocouple-----  
double celcius = thermocouple->readCelsius();  
yTT = y_span*celcius + y_zero; //gain sinyal thermocouple dlm celcius  
//Serial.print("Temperature: ");  
//Serial.print(yTT);  
//Serial.println(" C, ");  
delay(250);
```

2.3. Zero crossing detector

Zero Crossing detector adalah detektor Op-amp yang dapat mendeteksi perubahan sinyal gelombang sinusoidal dari positif ke negatif dan negatif ke positif, dan mengubah bentuk gelombang menjadi gelombang kotak. Zero crossing detector bekerja dengan mendeteksi perubahan voltase komparator pada output sinyal dengan input sinyal saat melewati nilai nol pada voltase referensi.

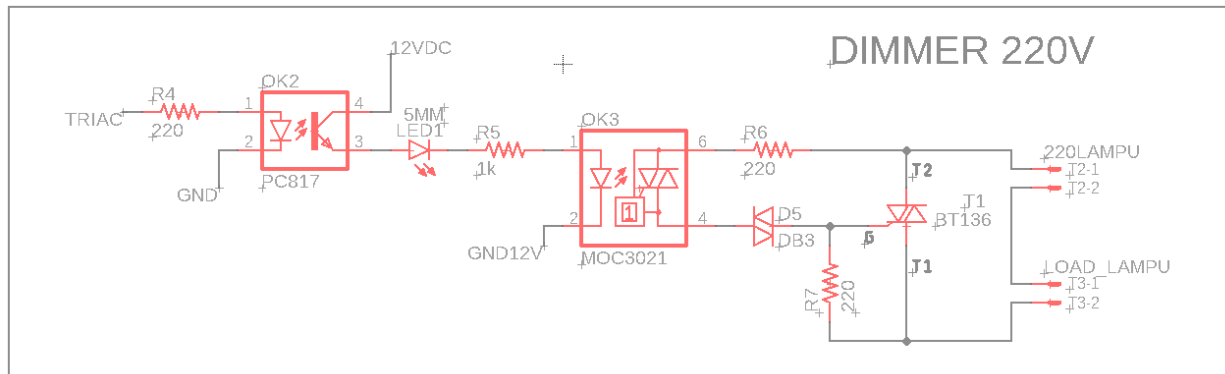


Gambar 7. Bentuk sinyal zero crossing dari sumber 12VAC

3. Dimmer 220V

Dimmer adalah suatu rangkaian elektronika yang bertujuan mengubah tegangan dan bentuk gelombang listrik. Dimmer dapat diaplikasikan untuk mengatur intensitas cahaya lampu dan mengatur kecepatan putaran berbagai rangkain listrik yang menggunakan kumparan motor. Dimmer memiliki konsep untuk memperlambat banyak kedip cahaya dalam satu detik sehingga bisa ditangkap secara visual oleh manusia.

Pada proyek ini dimmer 220V berfungsi untuk mengatur tingkat intensitas cahaya penerangan lampu pijar. Rangkaian ini bisa diatur oleh mikrokontroller mulai dari yang redup hingga ke remang-remang sampai ke nyala lampu yang terang.



Gambar 8. Rangkaian dimmer 220V

Komponen TRIAC BT136 berfungsi untuk mengatur besaran tegangan AC yang masuk ke perangkat lampu. Sedangkan, komponen DIAC DB3 berfungsi untuk mengatur bias gate TRIAC guna menentukan titik on dan off. Daya output rangkaian dimmer ini akan digunakan untuk mengendalikan intensitas cahaya lampu pijar dengan daya 5 Watt.

Rangkaian dimmer diaktifkan oleh pin digital output ESP32, tegangan yang diberikan berupa HIGH/LOW 3.3VDC. Tegangan ini perlu diberikan isolasi bertingkat menggunakan optocoupler PC817 untuk 12VDC dan MOC3021 untuk 220VAC. Trigger yang terlampaui kecil ini tidak bisa mengaktifkan triac driver MOC3021. $I_f = 5\text{VDC}/220\text{ohm} = 23\text{mA}$. Jika menggunakan tegangan 12VDC, $I_f = 12\text{VDC}/220\text{ohm} = 54.5\text{mA}$ cukup untuk mengaktifkan triac. LED kuning 5mm diberikan sebagai penanda rangkaian driver bekerja.

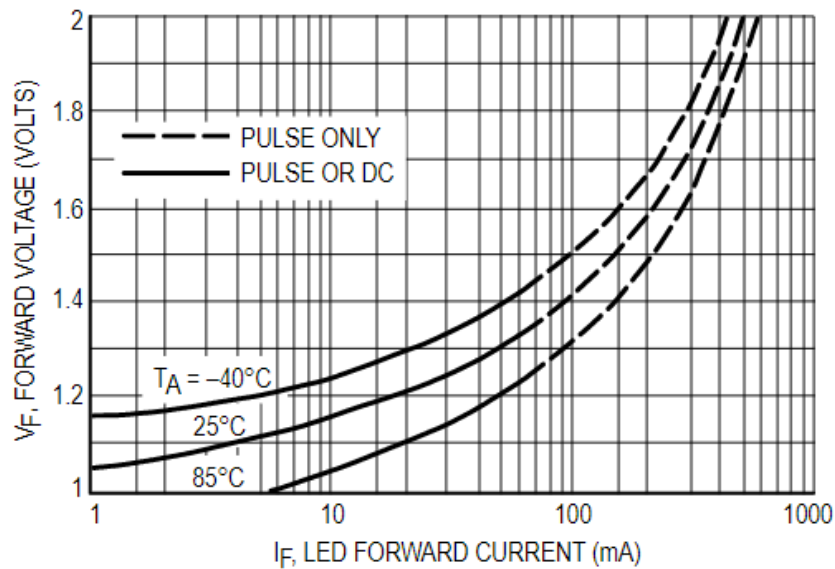


Figure 1. LED Forward Voltage versus Forward Current

Gambar 9. Kurva karakteristik MOC3021

Dimmer diprogram menggunakan library RBDdimmer yang berfungsi sebagai sinkronisasi sinyal digital PWM untuk pengaturan daya dan fase tegangan 220VAC. Terlebih dahulu perlu inialisasi program menggunakan kode berikut.

```
#include <RBDdimmer.h>
#define outputPin 27
#define zerocross 34
dimmerLamp dimmer(outputPin, zerocross);
dimmer.begin(NORMAL_MODE, ON);
```

Dimmer akan mulai mengatur tegangan sinus 220V dengan memanggil fungsi berikut disaat looping.

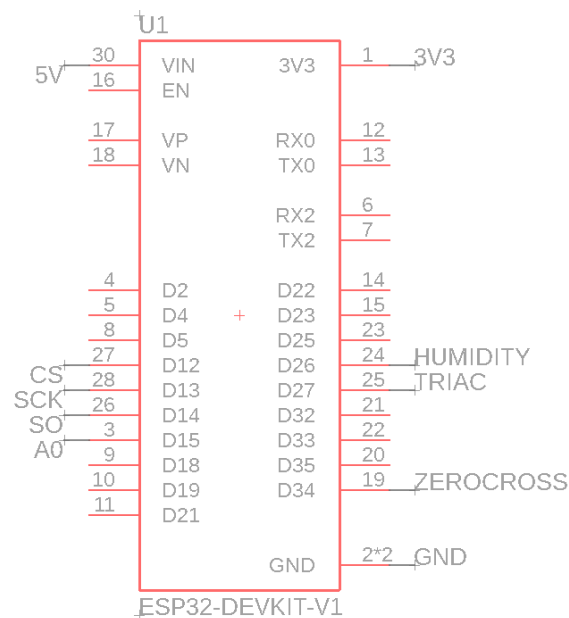
```
dimmer.setPower(mv_lamp);
```

4. ESP32 Devkit V1

ESP 32

ESP 32 adalah system pada chip mikrokontroler dari developer espressif sistem. Esp 32 terdapat 2 variasi core single dan dual core dengan, integrasi wifi dan bluetooth dan microporsesor tensillica's 32-bit Xtensa LX6 dimana biaya produksi untuk sistem ESP 32 termasuk rendah. ESP 32 memiliki integrasi komponen rf seperti power amplifier, Low-Noise Receive Amplifier, Antenna Switch, Filters, dan RF Balun. Sehingga komponen ekstra untuk meningkatkan kemampuan esp 32 sangatlah sedikit.

ESP 32 memiliki SRAM 520 KB, ROM 448 KB, dan 16 KB RTC SRAM dan ditambah dengan 34 GPIOs yang dapat diprogram sesuai keperluan serta micro USB untuk pembangkit dan memasukan Code, bahasa programming yang biasanya digunakan adalah Arduino IDE, Micropython, LUA, dan JavaScript.



Gambar 10. Pinout ESP32 yang digunakan

Programming PID

Program kontroller PID ditambahkan anti windup integrator agar nilai integral tidak overflow dan diberikan saturasi untuk membatasi sinyal kontrol PID ke actuator. Sebelum masuk ke actuator dimmer, sinyal ini dikalikan gain K_{uPID} agar intensitas lampu pijar sesuai.

```
//-----PID-----  
e = SP - yTT;
```

```

//anti-windup integrator
if(sigma_e > 1000){
    sigma_e = 1000;
}
else if(sigma_e < -1000){
    sigma_e = -1000;
}
else{
    sigma_e = sigma_e + e;
}

P = Kp*e;
I = Ki*(sigma_e);
D = Kd*(e - e_);
u_PID = P + I + D;
// updating
e_ = e;

// PID URV LRV SATURASI
if(u_PID > 10000){
    u_PID = 10000;
}
else if(u_PID < -10000){
    u_PID = -10000;
}

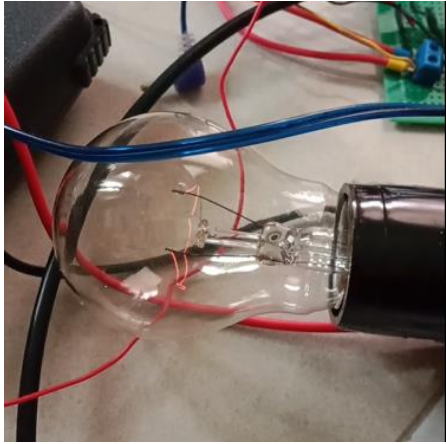
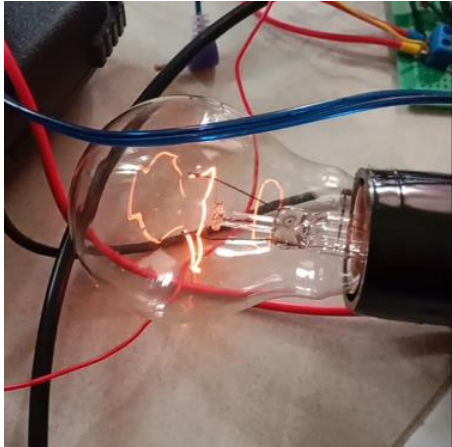
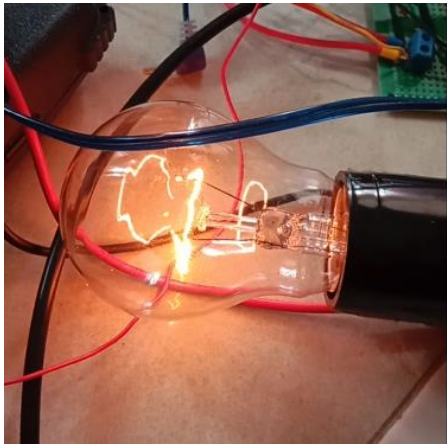
//int pot = analogRead(15);
float mv_lamp = u_PID*2; //gain, max agar tdk blink 94
dimmer.setPower(mv_lamp); // setPower(0-100%);
//delay(50);

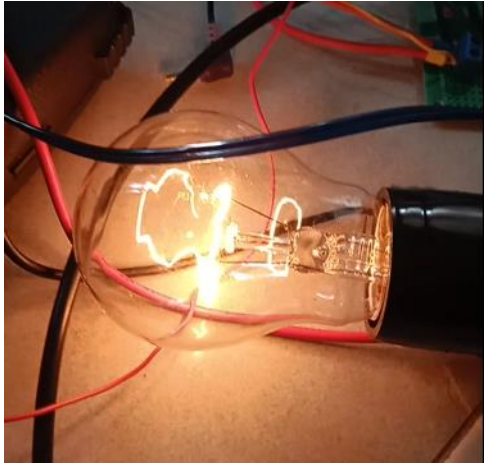
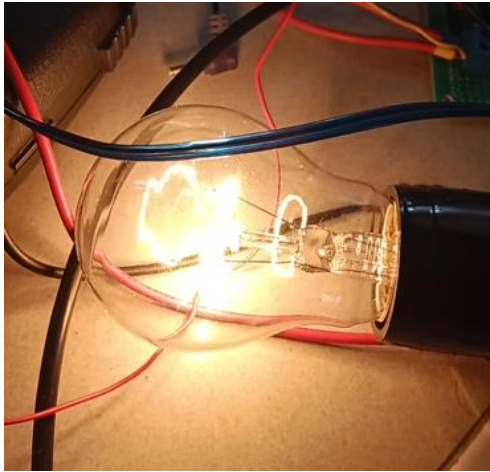
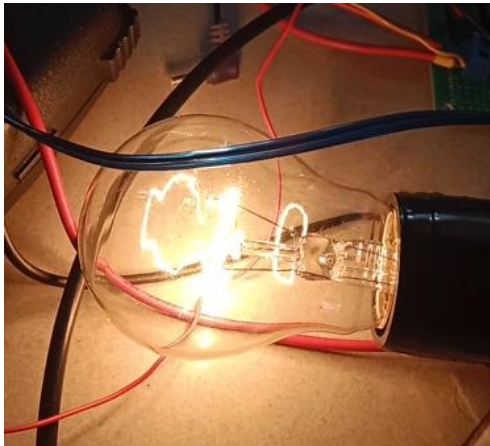
```

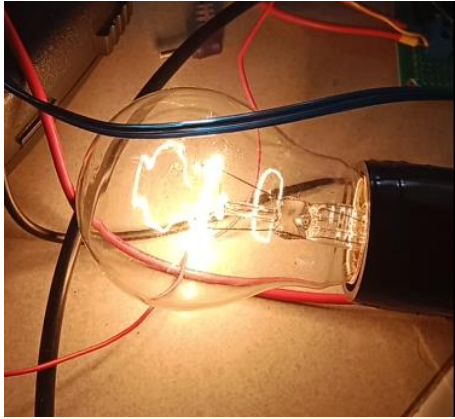
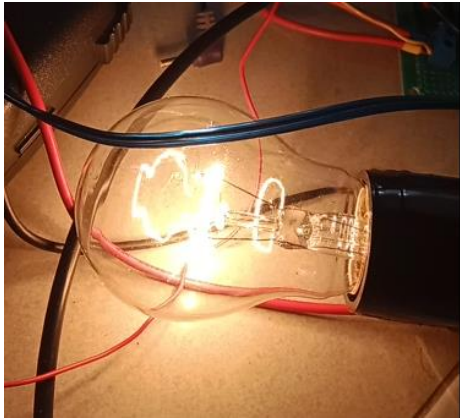
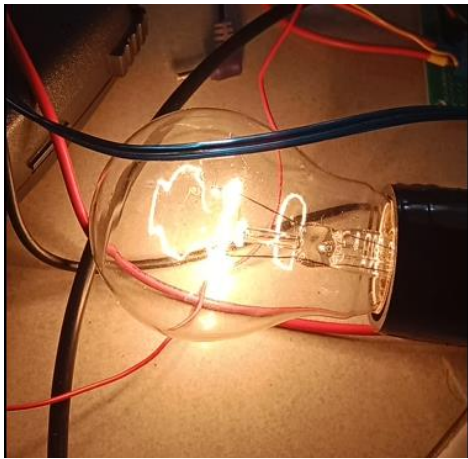
Tabel performansi dimmer manual open loop, $K_{uPID}=1$

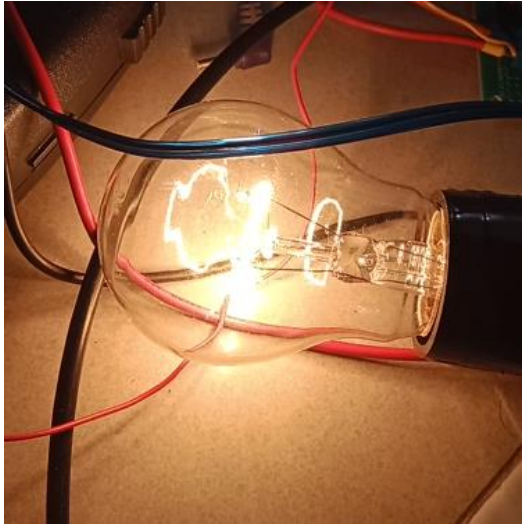
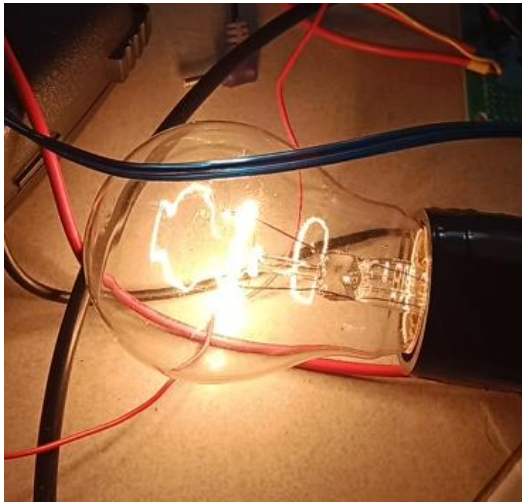
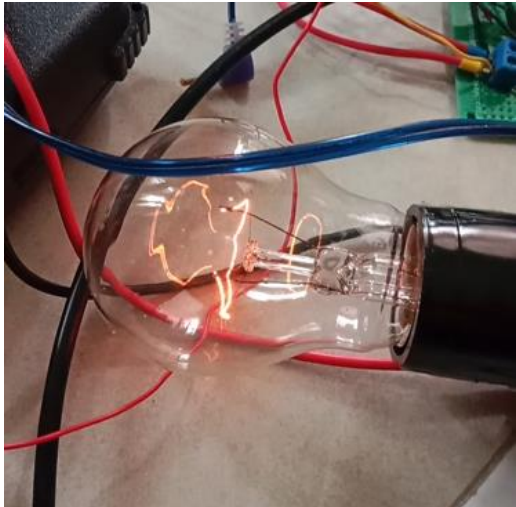
Digunakan LRV=0 dan URV=100

No.	u_{PID} ($K_p=1$, $K_i=0$, $K_d=0$)	Foto
-----	---	------

1	0	
2	10	
3	20	

4	30	
5	40	
6	50	

7	60	
8	70	
9	80	

10	90	
11	94 (max)	
12	100 (blinking)	

Pembuatan Database MySQL

mysql adalah open source Relational database management system yang sering digunakan untuk menyimpan data secara online (Open Source). MySQL digunakan dalam beberapa kasus, seperti pada automated system yang digunakan untuk mendapatkan data dan di input secara live. MySQL dapat disesuaikan dengan besar skala produk dan dapat melakukan penyesuaian sesuai dengan besar dari produk

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> data1bersih		499	InnoDB	utf8mb4_general_ci	80.0 KiB	-
<input type="checkbox"/> measurements		499	InnoDB	utf8mb4_unicode_ci	64.0 KiB	-
<input type="checkbox"/> measure_ayam		2	InnoDB	latin1_swedish_ci	16.0 KiB	-
<input type="checkbox"/> migrations		5	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> password_resets		0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
<input type="checkbox"/> personal_access_tokens		0	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-
<input type="checkbox"/> predicts		0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
<input type="checkbox"/> users		3	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
8 tables	Sum	1,008	InnoDB	latin1_swedish_ci	304.0 KiB	0 B

Gambar 11. List tabel dalam database u1588318_spt

	id	username	name	role	email	email_verified_at	password	remember_token	creat
<input type="checkbox"/>	1	hakhi	Hakhi Gya yektianto	admin	hakhiyektianto@gmail.com	NULL	\$2y\$10\$NyJYn2h86cvplVFJXBquOS7AlIkzIUeSkkH5c8.d8f...	NULL	2022-
<input type="checkbox"/>	2	pkm2022	PKM 2022	user	pkm2022@gmail.com	NULL	\$2y\$10\$3eztqhvc45Vuc0Do/bumOKn6pV0yDNq.C.LgOxRhRi...	NULL	2022-
<input type="checkbox"/>	3	faris	Faris Admin	admin		NULL	\$2y\$10\$xp9Pnt6PFaqYySudmttYyUwJqGNicGgcO8NVXIOWbHV...	NULL	NULL

Gambar 12. Tampilan tabel users

Server: localhost:3306 » Database: u1588318_spt » Table: measure_ayam

[Browse](#)
[Structure](#)
[SQL](#)
[Search](#)
[Insert](#)
[Export](#)
[Import](#)
[Operations](#)
[Triggers](#)

Showing rows 0 - 1 (2 total, Query took 0.0004 seconds.)

SELECT * FROM `measure_ayam`

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

		id	pv_suhu	pv_humidity	mv_dimmer	created_at
<input type="checkbox"/>	Edit Copy Delete	1	30	44	20	2022-12-02 15:54:37
<input type="checkbox"/>	Edit Copy Delete	2	30	44	20	2022-12-02 15:54:49

☐ Check all | With selected: Edit Copy Delete Export

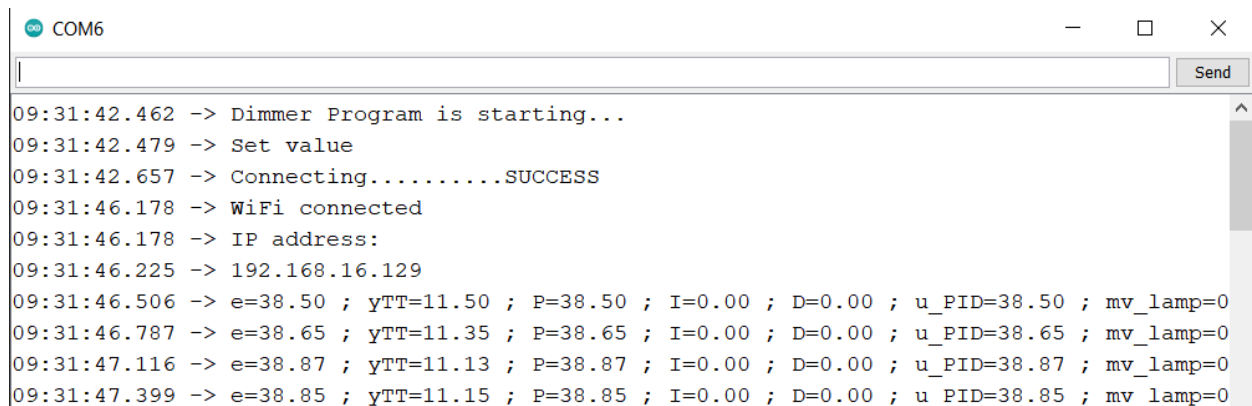
☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Gambar 13. Tampilan tabel measure_ayam

Protokol pengiriman data ke DB oleh ESP32

Inisialisasi program

```
#include <WiFi.h>
const char* host = "galium.ga";
const int httpPort = 80;
const char* ssid = "galium";
const char* password = "galium";
long last_millis_wifi;
String wifi="OFF";
long last_millis; //untuk interval send get DB
WiFiClient client;
```



```
COM6
09:31:42.462 -> Dimmer Program is starting...
09:31:42.479 -> Set value
09:31:42.657 -> Connecting.....SUCCESS
09:31:46.178 -> WiFi connected
09:31:46.178 -> IP address:
09:31:46.225 -> 192.168.16.129
09:31:46.506 -> e=38.50 ; yTT=11.50 ; P=38.50 ; I=0.00 ; D=0.00 ; u_PID=38.50 ; mv_lamp=0
09:31:46.787 -> e=38.65 ; yTT=11.35 ; P=38.65 ; I=0.00 ; D=0.00 ; u_PID=38.65 ; mv_lamp=0
09:31:47.116 -> e=38.87 ; yTT=11.13 ; P=38.87 ; I=0.00 ; D=0.00 ; u_PID=38.87 ; mv_lamp=0
09:31:47.399 -> e=38.85 ; yTT=11.15 ; P=38.85 ; I=0.00 ; D=0.00 ; u_PID=38.85 ; mv_lamp=0
```

Gambar xx. Tampilan connecting ke wifi

irim data tiap 5 detik

```
//-----send GET to MySQL per 5s-----
if ((millis()-last_millis)>5000){
    last_millis=millis();
    String protokol = "/update.php?e="+String(e)+
        "&pv_suhu="+String(yTT)+
        "&mv_dimmer="+String(mv_lamp)+
        "&pv_humidity="+String(hum);
    Serial.println( "Send Data To Server:"+send(protokol));
}
```

Pendefinisian protokol GET

```
String send(String url) {
    if (!client.connect(host, httpPort)) {
        return "Error Request Confirm";
    }
    client.print(String("GET ") + url + " HTTP/1.1\r\n" +
        "Host: " + host + "\r\n" +
        "User-Agent: BuildFailureDetectorESP8266\r\n" +
        "Connection: close\r\n\r\n");
}
```

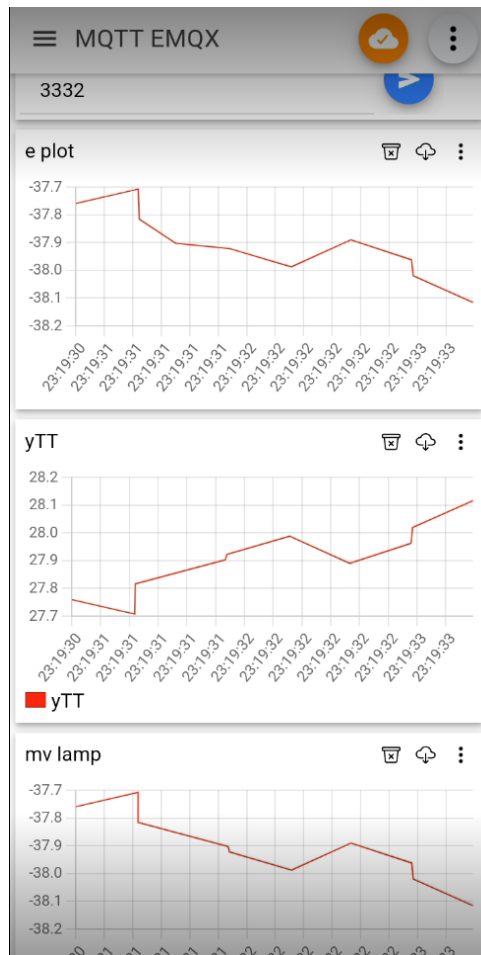
Memory used by program

```
145 }  
Done compiling.  
Sketch uses 654750 bytes (49%) of program storage space. Maximum is 1310720 bytes.  
Global variables use 39980 bytes (12%) of dynamic memory, leaving 287700 bytes for local  
142  
DOIT ESP32 DEVKIT V1, 80MHz, 921600, None on COM6
```

Gambar 15. Memory yang digunakan Program

Pembuatan data MQTT

MQTT adalah protokol perpesanan standar OASIS untuk Internet of Things (IoT). Ini dirancang sebagai transportasi perpesanan terbitkan / berlangganan yang sangat ringan yang ideal untuk menghubungkan perangkat jarak jauh dengan jejak kode kecil dan bandwidth jaringan minimal. MQTT saat ini digunakan di berbagai industri, seperti otomotif, manufaktur, telekomunikasi, minyak dan gas, dll.



Gambar 16. Statistik yang ditampilkan di Panel IoT Mqtt

Protokol ConnectWifi

```
void connectToWiFi() {  
  Serial.print("Connecting to ");  
  WiFi.begin(SSID, PWD);  
  Serial.println(SSID);  
  while (WiFi.status() != WL_CONNECTED) {  
    Serial.print(".");  
    delay(500);  
  }  
  Serial.print("Connected.");  
}
```

NOTE Bimbingan dengan Pak Hady 16 Desember 2022.

Protokol ConnectWiFi Esp32

```
void reconnect() {  
  Serial.println("Connecting to MQTT Broker...");  
  while (!mqttClient.connected()) {  
    Serial.println("Reconnecting to MQTT Broker..");  
    String clientId = "ESP32Client-";  
    //clientId += String(random(0xffff), HEX);  
    if (mqttClient.connect(clientId.c_str())) {  
      Serial.println("Connected.");  
      // subscribe to topic  
      mqttClient.subscribe("ewewew/swa/commands");  
    }  
  }  
}
```

Subs Mqtt

```
void callback(char* topic, byte* payload, unsigned int length) {  
  Serial.print("Callback - ");  
  Serial.print("Message:");  
  for (int i = 0; i < length; i++) {  
    Serial.print((char)payload[i]);  
  }  
}  
void setupMQTT() {  
  mqttClient.setServer(mqttServer, mqttPort);  
  // set the callback function  
  mqttClient.setCallback(callback);  
}
```

Pengiriman Data dan pengukuran

```
void loop() {  
  float hum = dht.readHumidity(); //DHT11  
  //-----Feedback Thermocouple----  
  double celcius = thermocouple->readCelsius();  
  yTT = y_span*celcius + y_zero; //gain sinyal thermocouple dlm celcius  
  //Serial.print("Temperature: ");  
  //Serial.print(yTT);  
  //Serial.println(" C, ");  
  delay(250);  
  if (!mqttClient.connected()){  
    reconnect();  
  }  
  mqttClient.loop();  
  long now = millis();  
  if (now - last_time > 60000) {  
    //-----PID-----  
    e = SP - yTT;  
    //anti-windup integrator  
    if(sigma_e > 1000){  
      sigma_e = 1000;  
    }  
    else if(sigma_e < -1000){  
      sigma_e = -1000;  
    }  
    else{  
      sigma_e = sigma_e + e;  
    }  
  }  
}
```

```

}
P = Kp*e;
I = Ki*(sigma_e);
D = Kd*(e - e_);
u_PID = P + I + D;
e_ = e;
if(u_PID > 10000){
    u_PID = 10000;
}
else if(u_PID < -10000){
    u_PID = -10000;
}
float mv_lamp = u_PID*2; //gain, max agar tdk blink 94
dimmer.setPower(mv_lamp); // setPower(0-100%);
Serial.print("e=");
Serial.print(e);
Serial.print(" ; yTT=");
Serial.print(yTT);
Serial.print(" ; P=");
Serial.print(P);
Serial.print(" ; I=");
Serial.print(I);
Serial.print(" ; D=");
Serial.print(D);
Serial.print(" ; u_PID=");
Serial.print(u_PID);
Serial.print(" ; mv_lamp=");
Serial.print(mv_lamp);
dtostrf(321.123,7, 3, buf_mv_lamp);
mqttClient.publish("ewewew/swa/temperature", buf_mv_lamp);
Serial.print(" ; Hum=");
Serial.println(hum);
dtostrf(123.099,7, 3, buf_hum);
mqttClient.publish("ewewew/swa/humidity", buf_hum);
last_time = now;
}
}

```

LAMPIRAN

Coding dengan IoT (Mqtt)

```
#include <PubSubClient.h>
#include <WiFi.h>
#include <Arduino.h>
#include <Adafruit_MQTT.h>

const char *SSID = "Zuto";//wifi
const char *PWD = "Zuto1234";//passwifi
//MQTT Client
WiFiClient wifiClient;
PubSubClient mqttClient(wifiClient);
char buf_mv_lamp[15];
char buf_hum[15];

char *mqttServer = "broker.hivemq.com";
int mqttPort = 1883;

long last_time = 0;
char data[100] ;

// -----Publish (Coding DHT11 mas
farris)-----
#include <RBDDimmer.h>
#define outputPin 27
#define zerocross 34 // for boards with CHANGEABLE input pins

dimmerLamp dimmer(outputPin, zerocross); //initialase port for dimmer
for ESP8266, ESP32, Arduino due boards
int outVal = 0;

#include <Thermocouple.h>
#include <MAX6675_Thermocouple.h>
#include <SmoothThermocouple.h>
#define SCK_PIN 13
#define CS_PIN 12
#define SO_PIN 14
```

```

#define SMOOTHING_FACTOR 5
Thermocouple* thermocouple = NULL;
float y_span = 1;
float y_zero = 0;
float yTT = 0;

#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

float sigma_e, e_, e = 0;
const int SP = 50;
const float Kp = 1;
const float Ki = 0;
const float Kd = 0;
float P, I, D, u_PID = 0;

void connectToWiFi() {
    Serial.print("Connecting to ");

    WiFi.begin(SSID, PWD);
    Serial.println(SSID);
    while (WiFi.status() != WL_CONNECTED) {
        Serial.print(".");
        delay(500);
    }
    Serial.print("Connected.");
}

//subs mqtt
void callback(char* topic, byte* payload, unsigned int length) {
    Serial.print("Callback - ");
    Serial.print("Message:");
    Serial.print(topic);
    Serial.print("] ");
    for (int i = 0; i < length; i++) {
        Serial.print((char)payload[i]);
    }
}

```



```

    Serial.println();
}

void setupMQTT() {
    mqttClient.setServer(mqttServer, mqttPort);
    // set the callback function
    mqttClient.setCallback(callback);
}

//Connect ESP32 to Mqtt
void reconnect() {
    Serial.println("Connecting to MQTT Broker...");
    while (!mqttClient.connected()) {
        Serial.println("Reconnecting to MQTT Broker..");
        String clientId = "ESP32Client-";

        if (mqttClient.connect(clientId.c_str())) {
            Serial.println("Connected.");
            // subscribe to topic
            mqttClient.subscribe("ewewew/swa/commands");
        }

    }
}

void setup() {
    Serial.begin(9600);
    dht.begin();
    dimmer.begin(NORMAL_MODE, ON); //dimmer initialisation:
name.begin(MODE, STATE)
    connectToWiFi();
    Serial.println("Setup berhasil konek :)");
    setupMQTT();
    Serial.println("Dimmer Program is starting...");
    Serial.println("Set value");

    Thermocouple* originThermocouple = new MAX6675_Thermocouple(SCK_PIN,
CS_PIN, SO_PIN);

```

```

    thermocouple = new SmoothThermocouple(originThermocouple,
SMOOTHING_FACTOR);
}

void loop() {
    float hum = dht.readHumidity(); //DHT11
    //-----Feedback Thermocouple----
    double celcius = thermocouple->readCelsius();
    yTT = y_span*celcius + y_zero; //gain sinyal thermocouple dlm celcius
    delay(250);
    if (!mqttClient.connected()){
        reconnect();
    }
    mqttClient.loop();

    long now = millis();
    if (now - last_time > 60000) {
        //-----PID-----
        e = SP - yTT;
        //anti-windup integrator
        if(sigma_e > 1000){
            sigma_e = 1000;
        }
        else if(sigma_e < -1000){
            sigma_e = -1000;
        }
        else{
            sigma_e = sigma_e + e;
        }

        P = Kp*e;
        I = Ki*(sigma_e);
        D = Kd*(e - e_);
        u_PID = P + I + D;
        // updating
        e_ = e;

        // PID URV LRV

```

```

if(u_PID > 10000){
    u_PID = 10000;
}
else if(u_PID < -10000){
    u_PID = -10000;
}

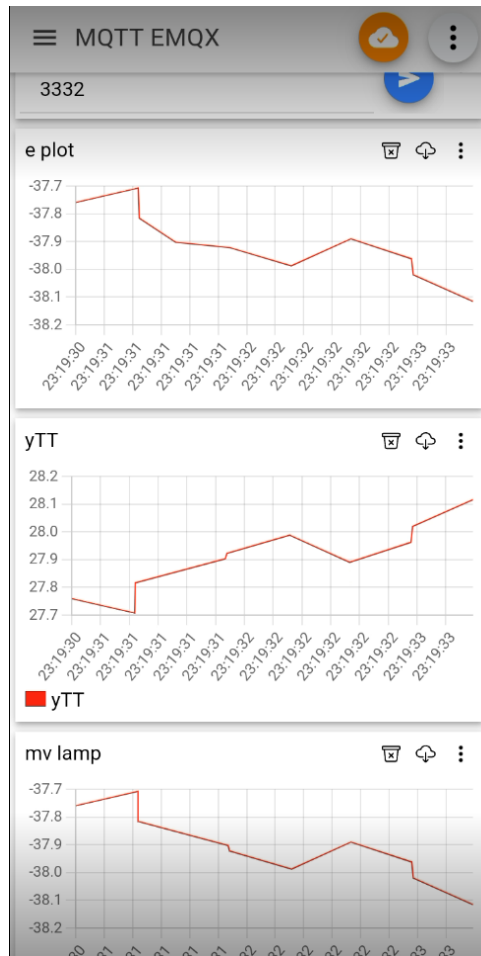
float mv_lamp = u_PID*2; //gain, max agar tdk blink 94
    dimmer.setPower(mv_lamp); // setPower(0-100%);
//delay(50);

Serial.print("e=");
Serial.print(e);
Serial.print(" ; yTT=");
Serial.print(yTT);
Serial.print(" ; P=");
Serial.print(P);
Serial.print(" ; I=");
Serial.print(I);
Serial.print(" ; D=");
Serial.print(D);
Serial.print(" ; u_PID=");
Serial.print(u_PID);
Serial.print(" ; mv_lamp=");
Serial.print(mv_lamp);

dtostrf(321.123,7, 3, buf_mv_lamp);

//gcvt(mv_lamp, 6, buf_mv_lamp);
mqttClient.publish("ewewew/swa/temperature", buf_mv_lamp);
Serial.print(" ; Hum=");
Serial.println(hum);
dtostrf(123.099,7, 3, buf_hum);
//gcvt(hum, 6, buf_hum);
mqttClient.publish("ewewew/swa/humidity", buf_hum);
last_time = now;
}
}

```



Gambar 17 Hasil Mqtt
Coding dengan IoT (Http)

```
#include <WiFi.h>
#include <PubSubClient.h>
const char* host = "ayam.galium.ga";
const int httpPort = 80;
// MQTT Broker
const char *mqtt_broker = "broker.emqx.io";
//const char *topic = "ayam/sensor";
const char *mqtt_username = "emqx";
const char *mqtt_password = "public";
const int mqtt_port = 1883;

const char* ssid      = "galium.ga";
const char* password = "galium.ga";
long last_millis_wifi;
String wifi="OFF";
long last_millis; //untuk interval send get DB
```

```

WiFiClient clientHTTP;
WiFiClient espClientMQTT;
PubSubClient client(espClientMQTT);

#include <RBDdimmer.h>
#define outputPin 27
#define zerocross 34 // for boards with CHANGEABLE input pins

dimmerLamp dimmer(outputPin, zerocross); //initialase port for dimmer for ESP8266, ESP32,
Arduino due boards
int outVal = 0;

#include <Thermocouple.h>
#include <MAX6675_Thermocouple.h>
#include <SmoothThermocouple.h>
#define SCK_PIN 13
#define CS_PIN 12
#define SO_PIN 14
#define SMOOTHING_FACTOR 5
Thermocouple* thermocouple = NULL;
float y_span = 1;
float y_zero = 0;
float yTT = 0;

#include "DHT.h"
#define DHTPIN 26
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

float sigma_e, e_, e = 0;
int SP = 50;
float Kp = 1;
float Ki = 0;
float Kd = 0;
float P, I, D, u_PID = 0;

void setup() {
  Serial.begin(9600);
  dht.begin();
  dimmer.begin(NORMAL_MODE, ON); //dimmer initialisation: name.begin(MODE,
STATE)
  Serial.println("Dimmer Program is starting...");
  Serial.println("Set value");

  Thermocouple* originThermocouple = new MAX6675_Thermocouple(SCK_PIN, CS_PIN,

```

```

SO_PIN);
thermocouple = new SmoothThermocouple(originThermocouple, SMOOTHING_FACTOR);

WiFi.mode(WIFI_STA);
WiFi.begin(ssid, password);
Serial.print("Connecting...");
//tunggu connect wifi
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("SUCCESS");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());

client.setServer(mqtt_broker, mqtt_port);
client.setCallback(callback);
while (!client.connected()) {
    String client_id = "esp32-client-";
    client_id += String(WiFi.macAddress());
    Serial.printf("The client %s connects to the public mqtt broker\n", client_id.c_str());
    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);
    }
}
}

void loop() {
    float hum = dht.readHumidity(); //DHT11
    //-----Feedback Thermocouple----
    double celcius = thermocouple->readCelsius();
    yTT = y_span*celcius + y_zero; //gain sinyal thermocouple dlm celcius
    //Serial.print("Temperature: ");
    //Serial.print(yTT);
    //Serial.println(" C, ");
    delay(250);

    //-----PID-----
    e = SP - yTT;
    //anti-windup integrator
    if(sigma_e > 1000){

```

```

        sigma_e = 1000;
    }
    else if(sigma_e < -1000){
        sigma_e = -1000;
    }
    else{
        sigma_e = sigma_e + e;
    }

    P = Kp*e;
    I = Ki*(sigma_e);
    D = Kd*(e - e_);
    u_PID = P + I + D;
    // updating
    e_ = e;

    // PID URV LRV
    if(u_PID > 10000){
        u_PID = 10000;
    }
    else if(u_PID < -10000){
        u_PID = 0;
    }

    float mv_lamp = u_PID*0.5; //gain, max agar tdk blink 94
    // jika mode manual
    //int pot = analogRead(15);
    //int mv_lamp = map(pot,0,4096,0,100);
    dimmer.setPower(mv_lamp); // setPower(0-100%);
    //delay(50);

    Serial.print("e=");
    Serial.print(e);
    Serial.print(" ; yTT=");
    Serial.print(yTT);
    Serial.print(" ; P=");
    Serial.print(P);
    Serial.print(" ; I=");
    Serial.print(I);
    Serial.print(" ; D=");
    Serial.print(D);
    Serial.print(" ; u_PID=");
    Serial.print(u_PID);
    Serial.print(" ; mv_lamp=");
    Serial.print(mv_lamp);
    Serial.print(" ; Hum=");

```

```

Serial.print(hum);
/*
e=1.69;
yTT=2.69;
mv_lamp=3.69;
hum=4.69;
*/
Serial.print(" ; SP=");
Serial.print(SP);
Serial.print(" ; Kp=");
Serial.print(Kp);
Serial.print(" ; Ki=");
Serial.print(Ki);
Serial.print(" ; Kd=");
Serial.println(Kd);

// publish and subscribe
char buf_e[15];
dtostrf(e,7, 7, buf_e);
client.publish("ayam/monitor/e", buf_e);
char buf_yTT[15];
dtostrf(yTT,7, 7, buf_yTT);
client.publish("ayam/monitor/yTT", buf_yTT);
char buf_mv_lamp[15];
dtostrf(e,7, 7, buf_mv_lamp);
client.publish("ayam/monitor/mv_lamp", buf_mv_lamp);
char buf_hum[15];
dtostrf(e,7, 7, buf_hum);
client.publish("ayam/monitor/hum", buf_hum);
//SP = client.subscribe("ayam/parameter/SP");
//Kp = client.subscribe("ayam/parameter/Kp");
//Ki = client.subscribe("ayam/parameter/Ki");
//Kd = client.subscribe("ayam/parameter/Kd");
client.loop();
//-----send GET to MySQL per 5s-----
if ((millis()-last_millis)>5000){
    last_millis=millis();
    String protokol = "/update.php?e="+String(e)+
"&pv_suhu="+String(yTT)+
"&mv_dimmer="+String(mv_lamp)+
"&pv_humidity="+String(hum);
    Serial.println( "Send Data To Server:"+send(protokol));
}
}
/*
//----PROTOKOL GET-----

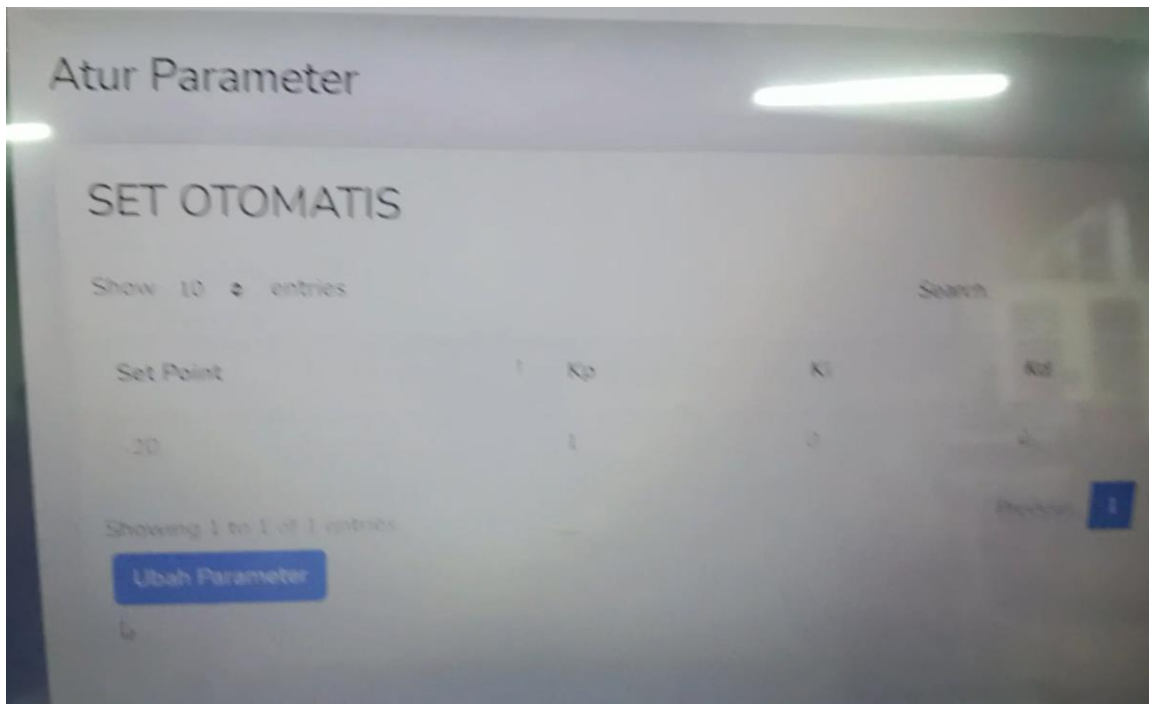
```



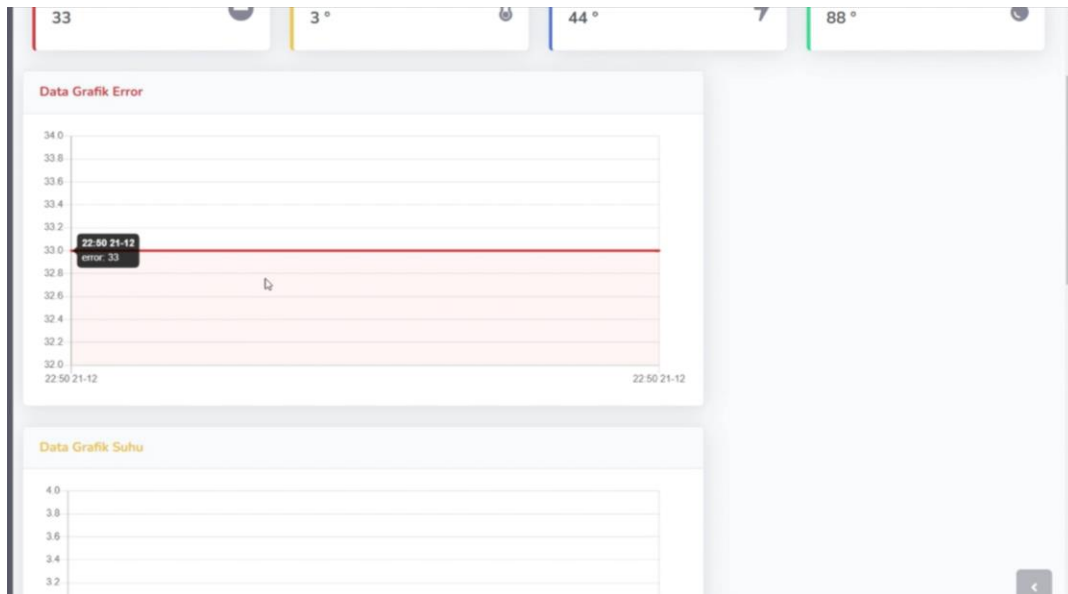
```

String send(String url) {
    if (!clientHTTP.connect(host, httpPort)) {
        return "Error Request Confirm";
    }
    clientHTTP.print(String("GET ") + url + " HTTP/1.1\r\n" +
        "Host: " + host + "\r\n" +
        "User-Agent: BuildFailureDetectorESP8266\r\n" +
        "Connection: close\r\n\r\n");
    clientHTTP.stop();
}
*/
void callback(char *topic, byte *payload, unsigned int length) {
    Serial.print("Message arrived in topic: ");
    Serial.println(topic);
    Serial.print("Message:");
    for (int i = 0; i < length; i++) {
        Serial.print((char) payload[i]);
    }
    Serial.println();
    Serial.println("-----");
}

```



Gambar 18. Hasil Http



Gambar 19. Hasil Web Ayam