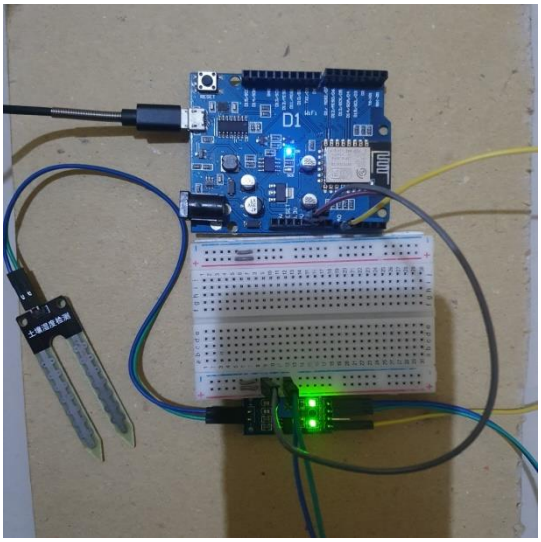
BAB IV
PEMBAHASAN

Pada bab ini akan menjelaskan tentang implementasi dari rancangan yang berupa perangkat keras, implementasi perangkat lunak, pengujian fungsional dan analisa hasil pengujian.

4.1 Implementasi Perangkat Keras

Pada sub bab ini akan menjelaskan tentang implementasi dari perancangan keseluruhan perangkat keras yang dibutuhkan pada sistem *greenhouse* berbasis IoT. Berikut penjelasan setiap rangkaian yang digunakan :

4.1.1 Implementasi Sensor *Soil Moisture*

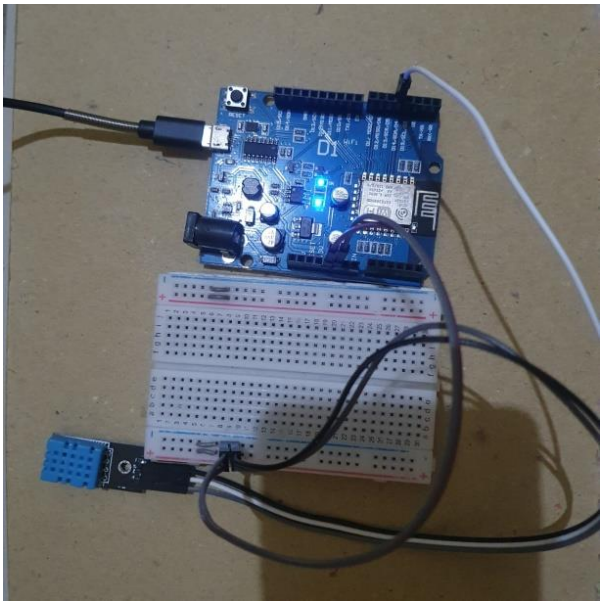


Gambar 4. 1 Rangkaian Mikrokontroller Dengan Sensor *Soil Moisture*

Pada Gambar 4.1 diatas merupakan rangkaian sensor *soil moisture* yang terhubung dengan mikrokontroller WEMOS D1R1 dan *Breadboard*. Pada sensor tersebut memiliki 3 buah kabel yang terhubung pada Wemos, penjelasan rangkaian seperti berikut. Pin VCC berwarna biru pada sensor terhubung dengan

pin 5V Wemos melalui *breadboard*, Pin AO (*Analog Ousput)* berwarna kuning pada sensor terhubung dengan pin AO Wemos, dan Pin GND berwarna hijau pada sensor terhubung dengan pin GND Wemos melalui *breadboard*. Pada sensor Soil Moisture terdapat dua lampu yang apabila lampu tersebut menyala maka sensor telah aktif dan siap digunakan.

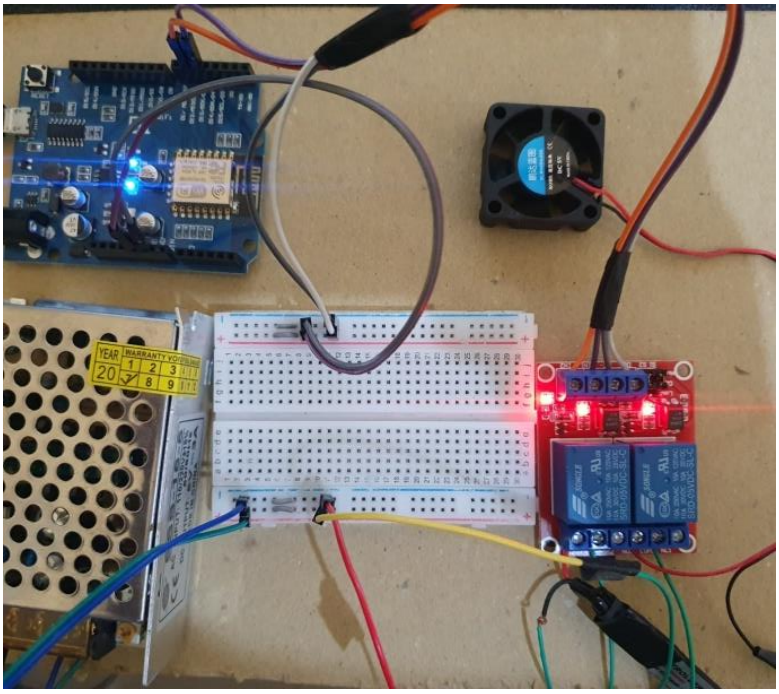
4.1.2 Implementasi Sensor Suhu DHT11



Gambar 4. 2 Rangkaian Mikrokontroller Dengan Sensor Suhu DHT11

Pada Gambar 4.2 diatas merupakan rangkaian sensor suhu yang terhubung dengan mikrokontroller Wemos D1R1 dan *Breadboard*. Pada sensor tersebut memiliki 3 buah kabel yang terhubung pada Wemos, penjelasan rangkaian seperti berikut. Pin VCC berwarna abu-abu pada sensor terhubung dengan pin 5V Wemos melalui *breadboard*, Pin Data berwarna putih pada sensor terhubung dengan pin D4 Wemos, dan Pin GND berwarna hitam pada sensor terhubung dengan pin GND Wemos melalui *breadboard*. Pada sensor suhu DHT11 tidak terdapat lampu yang menandakan sensor tersebut menyala tetapi jika kabel terhubung maka sensor akan aktif dan siap digunakan.

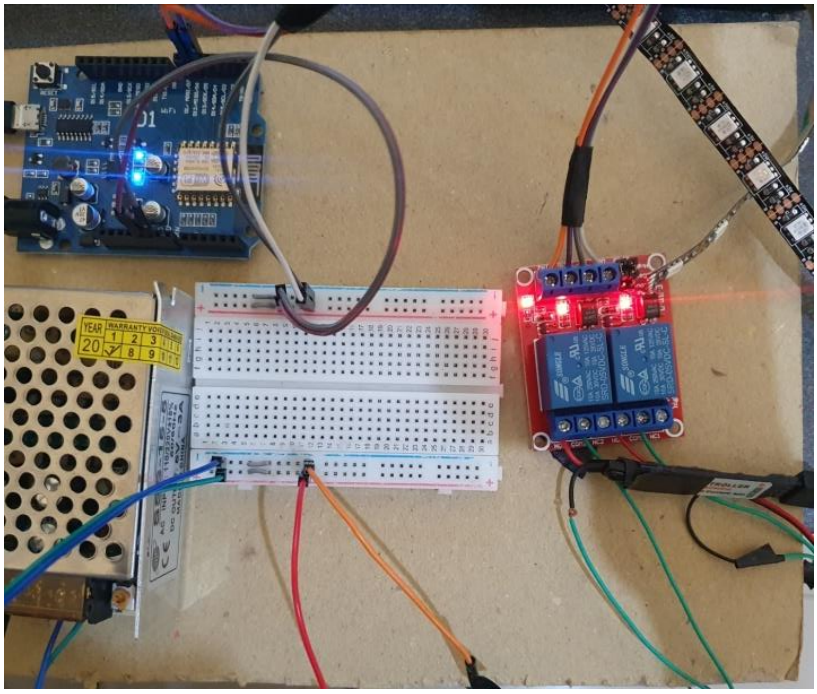
4.1.3 Implementasi Relay dan Kipas Angin



Gambar 4. 3 Rangkaian Mikrokontroller Dengan Relay dan Kipas Angin

Pada Gambar 4.3 diatas merupakan rangkaian *Relay* dan Kipas yang terhubung dengan mikrokontroller Wemos D1R1, *Breadboard dan Power Supply*. Pada relay tersebut memiliki 3 buah kabel yang terhubung pada Wemos dan 2 kabel terhubung pada Kipas dan *power supply*, penjelasan rangkaian seperti berikut. Pin VCC *relay* berwarna putih terhubung dengan pin 5V Wemos melalui *breadboard*, Pin GND *relay* berwarna hitam terhubung dengan pin GND Wemos melalui *breadboard*, Pin Data relay berwarna ungu terhubung dengan pin D5 Wemos, Pin NO (*Normally Open*) *relay* berwarna merah terhubung dengan kutub Positif kipas, Pin COM (*Common) relay* berwarna hijau kuning terhubung dengan kutub Positif *Power* melalui *breadboard* dan pada kutub Negatif kipas berwarna hijau merah terhubung pada kutub Negatif *Power* melalui *breadboard.*

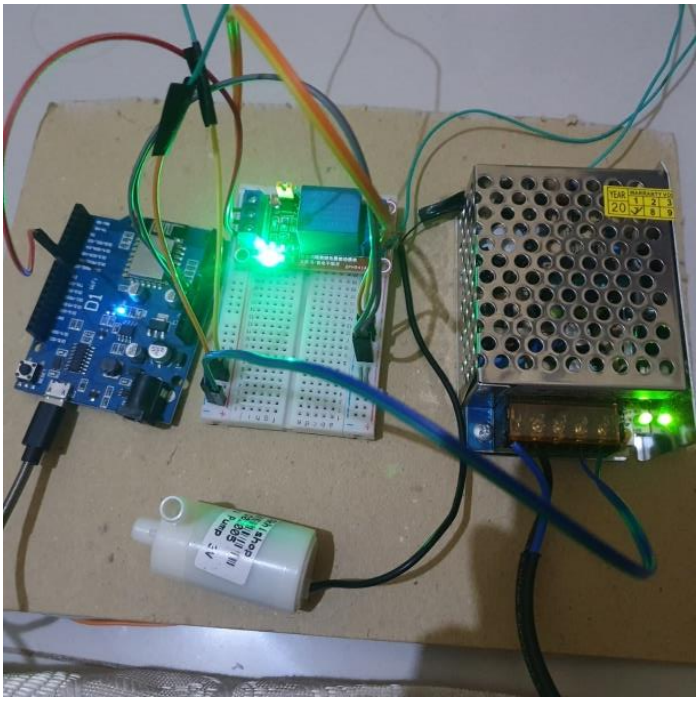
4.1.4 Implementasi Relay dan LED *Strip*



Gambar 4. 4 Rangkaian Mikrokontroller Dengan Relay dan LED *Strip*

Pada Gambar 4.4 diatas merupakan rangkaian *Relay* dan LED yang terhubung dengan mikrokontroller Wemos D1R1, *Breadboard dan Power Supply*. Pada relay tersebut memiliki 3 buah kabel yang terhubung pada Wemos dan 2 kabel terhubung pada Kipas dan *power supply*, penjelasan rangkaian seperti berikut. Pin VCC *relay* berwarna putih terhubung dengan pin 5V Wemos melalui *breadboard*, Pin GND *relay* berwarna hitam terhubung dengan pin GND Wemos melalui *breadboard*, Pin Data relay berwarna oranye terhubung dengan pin D6 Wemos, Pin NO (*Normally Open*) *relay* berwarna merah terhubung dengan kutub Positif kipas, Pin COM (*Common) relay* berwarna hijau merah terhubung dengan kutub Positif *Power* melalui *breadboard* dan pada kutub Negatif kipas berwarna hijau oranye terhubung pada kutub Negatif *Power* melalui *breadboard.*

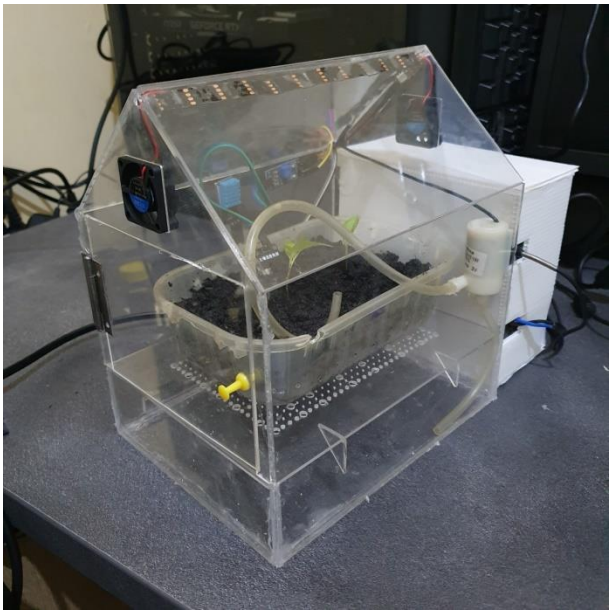
4.1.5 Implementasi Relay dan Pompa Air



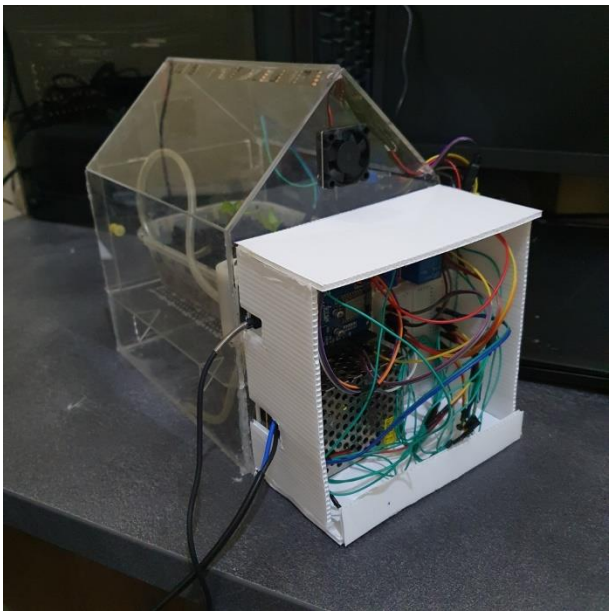
Gambar 4. 5 Rangkaian Mikrokontroller Dengan Relay dan Pompa Air

Pada Gambar 4.5 diatas merupakan rangkaian *Relay* dan Pompa yang terhubung dengan mikrokontroller Wemos D1R1, *Breadboard dan Power Supply*. Pada relay tersebut memiliki 3 buah kabel yang terhubung pada Wemos dan 2 kabel terhubung pada Pompa dan *power supply*, penjelasan rangkaian seperti berikut. Pin VCC *relay* berwarna kuning terhubung dengan pin 5V Wemos melalui *breadboard*, Pin GND *relay* berwarna oranye terhubung dengan pin GND Wemos melalui *breadboard*, Pin Data relay berwarna merah terhubung dengan pin D7 Wemos, Pin NO (*Normally Open*) *relay* berwarna merah terhubung dengan kutub Positif pompa, Pin COM (*Common) relay* berwarna hijau kuning terhubung dengan kutub Positif *Power* melalui *breadboard* dan pada kutub Negatif pompa berwanra hijau oranye terhubung pada kutub Negatif *Power* melalui *breadboard.* Pada *relay* terdapat satu lampu yang apabila lampu tersebut menyala maka relay telah aktif dan siap digunakan.

4.1.6 Implementasi Rangkaian Keseluruhan Alat



Gambar 4. 6 Tampak Depan Rangkaian Alat



Gambar 4. 7 Tampak Belakang Rangkaian Alat

Pada Gambar 4.6 dan Gambar 4.7 diatas merupakan tampilan denpan dan tampilan belakang dari keseluruhan rangkaian alat yang telah dibuat yaitu sistem *greenhouse* berbasis IoT. Alat ini menggunakan wadah berupa maket sederhana yang berbentuk seperti sebuah *greenhouse* dengan ukuran tinggi 17cm x lebar

15cm x panjang 20cm yang dilengkapi dengan tangki penyimpanan air berbentuk balok persegi panjang dibawahnya dengan ukuran lebar 15cm x panjang 20cm, diantara maket *greenhouse* dan tangki air terdapat sekat dimana sekat berbentuk persegi panjang yang berukuran tinggi 5cm x lebar 15cm x panjang 20cm dengan adanya lubang kecil di tengah persegi untuk jalur masuknya air yang dikeluarkan oleh pipa air masuk pada tanaman dan juga tetesan air dari atas yang di pasang pada *greenhouse*. Pada bagian belakang wadah *greenhouse* terdapat wadah komponen yang digunakan pada sistem ini dengan ukuran tinggi 14cm x lebar 8 cm x panjang 8 cm yang terdapat beberapa lubang pada bagian kanan dan kiri untuk jalur masuknya kabel pada komponen di dalamnya. Komponen yang digunakan pada sistem ini yaitu, sensor suhu DHT11, sensor *Soil Moisture*, Wemos D1R1, *Breadboard*, tiga *relay*, kipas angin, LED strip, pompa air, *power supply* 5V.

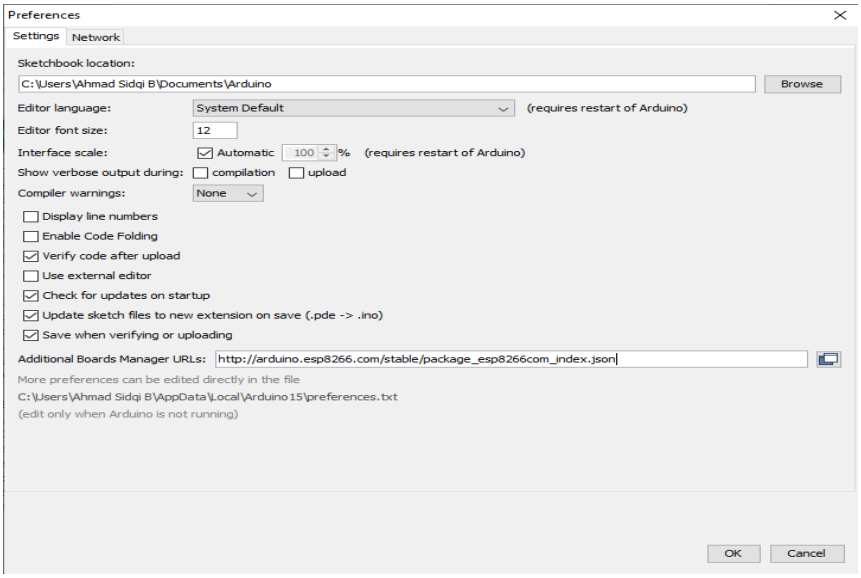
4.2 Implementasi Perangkat Lunak

Pada sub bab ini akan menjelaskan tentang implementasi sistem dari rancangan alat pada sistem yang bertujuan untuk menampilkan setiap tahapan yang akan dilakukan pada sistem yang telah dibuat. Berikut dibawah ini penjelasan implementasi sistem :

4.2.1 Instalasi Board Wemos D1R1 Arduino IDE

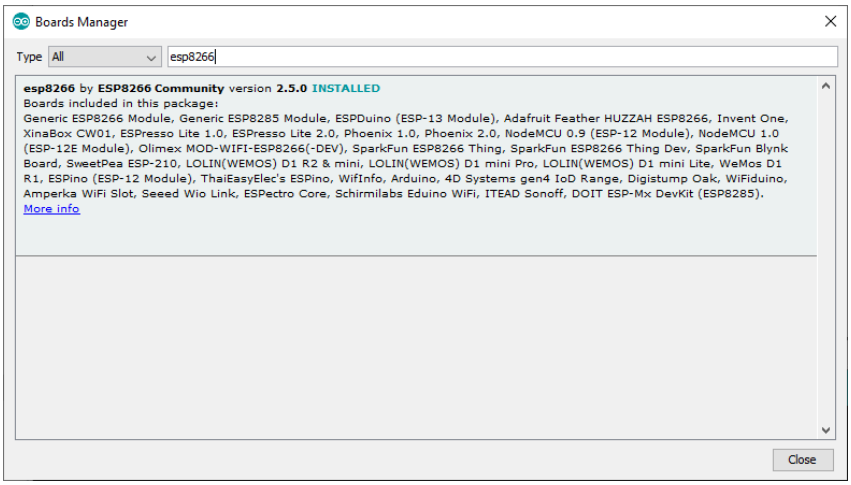
Pada tahap ini merupakan proses installasi board pada alat yang akan digunakan, karena alat ini menggunakan board Wemos D1R1 maka diperlukanya proses installasi pada program Arduino IDE agar program dapat dijalankan pada board. Langkah pertama yaitu pilih menu lalu masuk pada

pilihan Preferences lalu masukkan link yang telah ada, seperti pada Gambar 4.8 berikut.



Gambar 4. 8 Memasukkan Link Prefences

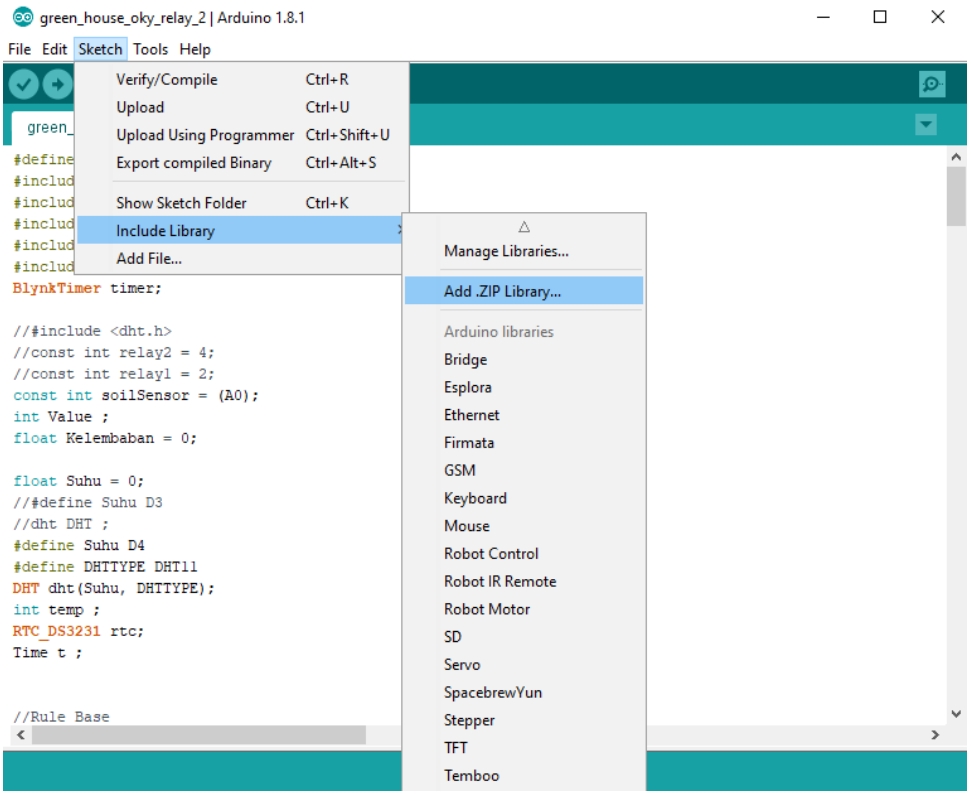
Dengan memasukkan link pada menu prefences akan membuka akses installasi pada board Wemos D1R1 yang menggunakan modul base esp8266 didalam menu boards manager, setelah selesai maka langkah berikutnya yaitu masuk pada menu Tools, lalu terdapat pilihan Board dan pilih Boards Managers. Cari dan install esp8266 untuk menentukan jenis mikrokontroller yang akan digunakan. Seperti pada Gambar 4.9 berikut.



Gambar 4. 9 Installasi Board esp8266

4.2.2 Instalasi Library Arduino IDE

Setelah melalui tahap installasi board esp8266, maka tahap berikutnya yaitu proses installasi library pada setiap komponen yang memerlukan penggunaan library. Proses program yang detail pada komponen yang menggunakan library dapat mempermudah pembacaan data komponen yang merupakan program *open source*. Terdapat beberapa cara untuk melakukan installasi, salah satu cara yang dapat dilakukan dengan pilih menu sketch lalu pilih include library lalu pilih Add .ZIP Library. Unduh terlebih dahulu file .zip library yang dibutuhkan pada rangkaian alat, lalu pilih open pada file .zip yang telah disiapkan. Seperti pada Gambar 4.10 berikut.



Gambar 4. 10 Installasi Library

Library yang digunakan pada rangkaian alat yaitu Library Sensor Suhu DHT, Library ESP8266 dan Library Blynk.

4.2.3 Pengaturan Jaringan Wemos D1R1 dengan Server Blynk

Agar program dari sistem *greenhouse* berbasis IoT dapat diproses dan terhubung pada server blynk, maka langkah pertama yang harus dilakukan yaitu menghubungkan mikrokontroller Wemos D1R1 dengan jaringan wifi yang dapat diatur dengan menentukan SSID serta Password agar dapat terhubung dengan jaringan internet. Agar dapat terhubung dengan server blynk maka masukkan kode Auth pada program yang akan digunakan bersamaan dengan memasukkan kode untuk menyambungkan jaringan internet pada platform Arduino IDE, berikut cara menghubungkan jaringan Wemos D1R1 pada jaringan internet dan server blynk. Seperti pada Gambar 4.11 berikut.

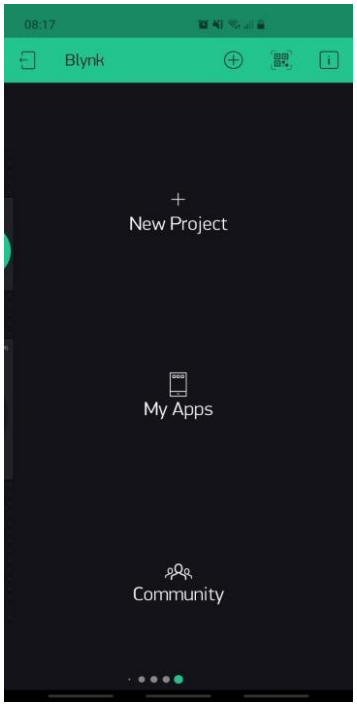
Gambar 4. 11 Penyambungan Jaringan Wifi dan Blynk pada IDE

Kode 4. 1 *Sourcecode* Penyambungan Jaringan Wifi dan Blynk

```
1  #define BLYNK_PRINT Serial
2  #include <ESP8266WiFi.h>
```

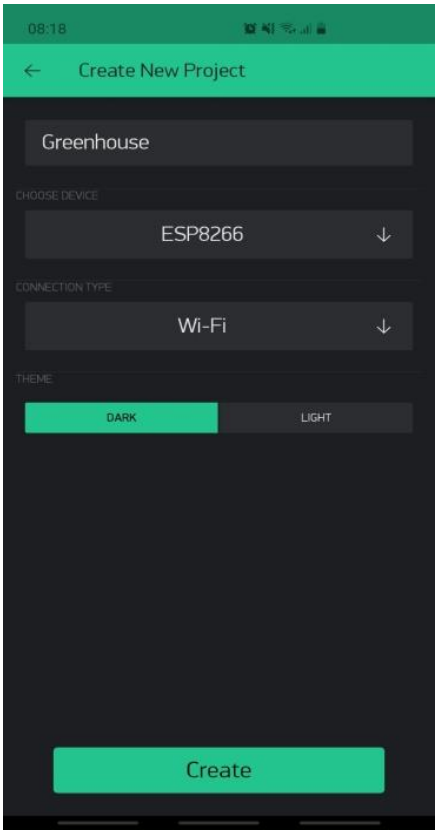
```
#include <Blynk.h>
#include <BlynkSimpleEsp8266.h>

// Kode yang dikirim pada email
 char auth[] = "oFhGOI3bm-3L_6sDLl6B1jemKnkYJYKM";

//Nama Wifi dan Password Wifi
 char ssid[] = "OKI";
 char pass[] = "12345678";

void setup()
{
  Serial.begin(115200);
  Blynk.begin(auth, ssid, pass);
}

void loop()
{
  Blynk.run();
}
```

Pada Kode 4.1 diatas menampilkan untuk mendapatkan kode Auth agar dapat menghubungkan sistem pada server blynk maka langkah pertama yaitu unduh dan install terlebih dahulu aplikasi blynk pada ponsel yang akan digunakan. Langkah berikutnya yaitu buka aplikasi dan daftar aku baru dengan pilih *sign up account* atau masuk menggunakan akun Facebook, jika sudah maka akan masuk pada menu awal yang terdapat 3 pilihan yaitu New Project, My Apps dan Community seperti pada Gambar 4.12 berikut.

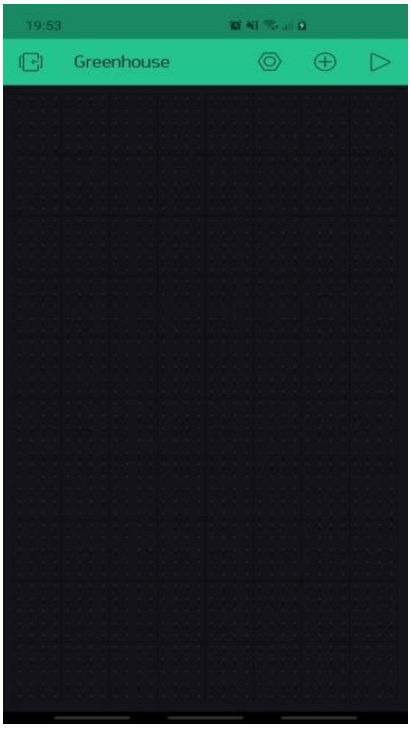


Gambar 4. 12 Tampilan Menu Awal Blynk

Langkah berikutnya pilih menu *New Project* lalu akan berpindah pada tampilan *Create New Project*, pada kolom *Project Name* dapat diisikan dengan nama sistem atau projek yang ingin dibuat. Pada kolom *Choose Device* yang merupakan kolom untuk memilih mikrokontroller apa yang digunakan, untuk sistem ini menggunakan mikrokontroller WEMOS D1R1 yang menggunakan modul *base* esp8266 maka pada kolom ini memilih ESP8266. Pada kolom *Connection Name* yang merupakan kolom untuk memilik jalur koneksi yang digunakan untuk menghubungkan mikrokontroller dengan server blynk, untuk sistem ini menggunakan koneksi wifi untuk menghubungkan mikrokontroller Wemos dengan server blynk. Pada kolom *Theme* terdapat 2 pilihan yaitu *Dark* untuk tampilan gelap dan *Light* untuk tampilan terang, setelah memilih maka tekan pilihan *Create*. Setelah terbuat maka blynk akan mengirimkan kode Auth untuk dimasukkan pada program pada email yang digunakan lalu akan berpindah pada halaman projek. Seperti pada Gambar 4.13 dan 4.14 berikut.



Gambar 4. 13 Tampilan Halaman *Create New Project*



Gambar 4. 14 Tampilan Halaman Projek

4.2.4 *Fuzzifikasi*

Fuzzifikasi merupakan langkah pertama untuk menentukan nilai inputan dimana tiap-tiap variabel yang digunakan perlu dilakukanya insialisasi, setelah inisialisasi maka langkah selanjutnya yaitu menentukan jumlah range pada setiap derajat keanggotaan pada variabel suhu dan kelembaban tanah. Berikut kode untuk proses *fuzzifikasi* pada setiap variabel.

a. *Fuzzifikasi* Suhu

Setelah nilai variabel telah didapatkan oleh pembacaan sensor suhu maka langkah berikutnya adalah membandingkan nilai yang ada dengan kondisi suhu dingin, sejuk, normal dan dingin untuk menentukan nilai derajat keanggotaanya. Berikut Kode 4.2 yang merupakan kode *fuzzifikasi* suhu.

Kode 4. 2 Sourcecode Fuzzifikasi *Variabel* Suhu

```
1  void Fuzzifikasi() {
2       fuzzySuhu();
3       fuzzyKelembaban();
4  }
5  int fuzzySuhu(){
6    // untuk suhu dingin
7    if (Suhu <= 20)
8     { suhu [0] = 1;}
9    else if (Suhu > 20 && Suhu < 24)
10    { suhu [0] = float (24 - Suhu)/(24 - 20); }
11   else
12    { suhu [0] = 0;}
13
14   // untuk suhu sejuk
15   if (Suhu == 24)
16    { suhu [1] = 1; }
17   else if (Suhu > 20 && Suhu < 24)
18    { suhu [1] = float (Suhu - 20)/(24 - 20); }
19   else if (Suhu > 24 && Suhu < 28)
20    { suhu [1] = float (28 - Suhu)/(28 - 24); }
21   else
22    { suhu [1] = 0;}
```

```
23
24    // untuk suhu normal
25    if (Suhu == 28)
26     { suhu [2] = 1;}
27    else if (Suhu > 24 && Suhu < 28)
28     { suhu [2] = float (Suhu - 24)/(28 - 24); }
29    else if (Suhu > 28 && Suhu < 32)
30     { suhu [1] = float (32 - Suhu)/(32 - 28); }
31    else
32     { suhu [2] = 0;}
33
34    // untuk suhu panas
35    if (Suhu >= 32)
36     { suhu [3] = 1;}
37    else if (Suhu > 28 && Suhu <= 32)
38     { suhu [3] = float (Suhu - 28)/(32 - 28); }
39    else
40     { suhu [3] = 0; }
41       }
```

b. *Fuzzifikasi* Kelembaban Tanah

Setelah nilai variabel telah didapatkan oleh pembacaan sensor soil moisture maka langkah berikutnya adalah membandingkan nilai yang ada dengan kondisi tanah kering, lembab dan basah untuk menentukan nilai derajat keanggotaanya. Berikut Kode 4.3 yang merupakan kode fuzzifikasi kelembaban tanah.

Kode 4. 3 Sourcecode Fuzzifikasi *Variabel* Kelembaban Tanah

```
1  int fuzzyKelembaban(){
2   // untuk  kondisi kering
3   if (Kelembaban < 60)
4    { kelembaban [0] = 1;}
5   else if (Kelembaban > 60 && Kelembaban < 65)
6    { kelembaban [0] = float (65-Kelembaban)/
7    (65 - 60); }
8   else
9    { kelembaban [0] = 0;}
10
11  // untuk kondisi lembab
12  if (Kelembaban == 65)
```

```
13    { kelembaban [1] = 1;}
14   else if (Kelembaban > 60 && Kelembaban < 65)
15    { kelembaban [1] = float (Kelembaban - 60)/
16    (65 - 60);}
17   else if (Kelembaban > 65 && Kelembaban < 70)
18    { kelembaban [1] = float (70-Kelembaban)/
19    (70 - 65);}
20   else
21    { kelembaban [1] = 0;}
22
23   // untuk kondisi basah
24   if (Kelembaban >= 70)
25    { kelembaban [2] = 1;}
26   else if (Kelembaban > 65 && Kelembaban < 70)
27    { kelembaban [2] = float (Kelembaban-65)/
28    (70-65);}
29   else
30    { kelembaban [2] = 0;}
31  }
```

4.2.5 *Fuzzy Rule*

Proses aturan *fuzzy* atau Inferensi adalah proses penggabungan aturan berdasarkan data yang didapat. Proses evaluasi aturan *fuzzy* pada kode yang digunakan akan berfungsi untuk menjelaskan hubungan antara variabel masukan dan keluaran, dimana variabel yang diproses akan menghasilkan variabel berbentuk *fuzzy*. Berikut Kode 4.4 yang merupakan kode *fuzzy rule*.

Kode 4. 4 *Sourcecode Fuzzy Rule*

```
1   void fuzzy_rule(){
2
3    int i, j;
4    for ( int i=0; i<=2; i=i+1)
5    {
6      for (int  j=0; j<=2; j=j+1)
7      {
8     Ap = min(suhu[i], kelembaban[j]);
9        rule [i][j] = Ap;
10     }
11   }
12   rule00 = rule [0][0]; // (dingin,kering = sedang)
```

```
13   rule01 = rule [0][1]; // (dingin,lembab = cepat)
14   rule02 = rule [0][2]; // (dingin,basah =  cepat)
15
16   rule10 = rule [1][0]; // (sejuk,kering = sedang)
17   rule11 = rule [1][1]; // (sejuk,lembab = cepat)
18   rule12 = rule [1][2]; // (sejuk,basah = cepat)
19
20   rule20 = rule [2][0]; // (normal,kering = lama)
21   rule21 = rule [2][1]; // (normal,lembab = sedang)
22   rule22 = rule [2][2]; // (normal,basah = Cepat)
23
24   rule30 = rule [3][0]; // (panas, kering = lama)
25   rule31 = rule [3][1]; // (panas, lembab = sedang)
26   rule32 = rule [3][2]; // (panas, basah = cepat)
27
28 }
```

4.2.6 *Defuzzifikasi*

Setelah dilakukanya perhitungan pada tiap *rule* dengan mengambil nilai dari setiap variabel yang sesuai dengan kondisi *rule* yang ada, proses perhitungan akan mengubah variabel berbentuk *fuzzy* menjadi data pasti yang akan menjadi penentu pada output sistem. Berikut Kode 4.6 yang merupakan kode *defuzzifikasi*.

Kode 4. 5 *Sourcecode Defuzzifikasi*

```
1  void Defuzzifikasi () {
2
3  waktu = (rule[0][0]* Sedang) + (rule[0][1] * Cepat) +
4  (rule[0][2] * Cepat) + (rule[1][0] * Sedang)
5  + (rule[1][1] * Cepat) + (rule[1][2] * Cepat)
6  + (rule[2][0] * Lama) + (rule[2][1] * Sedang)
7  + (rule[2][2] * Cepat) + (rule[3][0] * Lama)
8  + (rule[3][1] * Sedang) + (rule[3][2] * Cepat); {
9  }
10   defuz = 0;
11   int i, j;
12   for ( i=0; i<=2; i=i+1)
13   {
14     for ( j=0; j<=2; j=j+1)
```

```
15      {
16      defuz = defuz + rule [i][j];
17      }
18    }
19   waktu = waktu / defuz ;
20   keluaran = waktu ;
21  }
```

4.2.7 Penentuan Kondisi *Relay Output*

Pada sistem *greenhouse* berbasis IoT terdapat tiga buah *relay* yang digunakan untuk mengkontrol output yang digunakan seperti kipas angin, led *strip* dan pompa air. Berikut penjelasan penentuan kondisi *relay* untuk *output* yang digunakan.

a. Kipas angin

Pada nilai sensor suhu yang telah diambil sebelumnya akan digunakan sebagai acuan set nilai dimana akan diproses untuk menghidupkan output berupa kipas angin secara otomatis. Pada output kipas angin set nilai yang digunakan ketika suhu berada melebihi 30°C dan kurang dari 45°C maka relay yang terhubung dengan kipas angin akan menyala, ketika kondisi tersebut tidak terpenuhi maka relay yang terhubung dengan kipas angin akan mati. Berikut Kode 4.6 yang merupakan kode penentuan kondisi *relay* pada *output* kipas angin.

Kode 4. 6 *Sourcecode* Kondisi *Relay* pada *Output* Kipas Angin

```
1       void Config_Kipas(){
2
3         if ( temp > 28 && temp < 45){
4           digitalWrite(Kipas,HIGH);
5           }
6         else {
7           digitalWrite(Kipas,LOW);
```

```
8           }
9  }
```

b. LED *Strip*

Pada nilai sensor suhu yang telah diambil sebelumnya akan digunakan sebagai acuan set nilai dimana akan diproses untuk menghidupkan output berupa led *strip* secara otomatis. Pada output led *strip* set nilai yang digunakan ketika suhu berada melebihi 10°C dan kurang dari 30°C maka relay yang terhubung dengan led *strip* akan menyala, ketika kondisi tersebut tidak terpenuhi maka relay yang terhubung dengan led *strip* akan mati. Berikut Kode 4.7 yang merupakan kode penentuan kondisi *relay* pada *output* led *strip*.

Kode 4. 7 *Sourcecode* Kondisi *Relay* pada *Output* LED *Strip*

```
1       void Config_Lampu (){
2
3         if ( temp > 10 && temp < 30) {
4           digitalWrite(Lampu,HIGH);
5            Serial.print("Lampu Nyala ");
6           }
7         else {
8           digitalWrite(Lampu,LOW);
9           }
10 }
```

c. Pompa Air

Pada nilai *defuzzifikasi* yang merupakan hasil dari perhitungan metode *fuzzy* dari variabel suhu dan kelembaban tanah yang telah diambil sebelumnya akan digunakan sebagai acuan set nilai dimana akan diproses untuk menghidupkan output berupa pompa air secara otomatis. Pada output pompa air set nilai yang digunakan ketika hasil keluaran

defuzzifikasi berada pada nilai 3 maka relay yang terhubung dengan pompa air akan menyala selama 3 detik lalu relay yang terhubung dengan pompa air akan mati selama 3 detik, ketika kondisi hasil keluaran *defuzzifikasi* berada pada nilai 6 maka relay yang terhubung dengan pompa air akan menyala selama 6 detik lalu relay yang terhubung dengan pompa air akan mati selama 6 detik, dan ketika kondisi hasil keluaran *defuzzifikasi* berada pada nilai 9 maka relay yang terhubung dengan pompa air akan menyala selama 9 detik lalu relay yang terhubung dengan pompa air akan mati selama 9 detik. Berikut Kode 4.8 yang merupakan kode penentuan kondisi *relay* pada *output* pompa air.

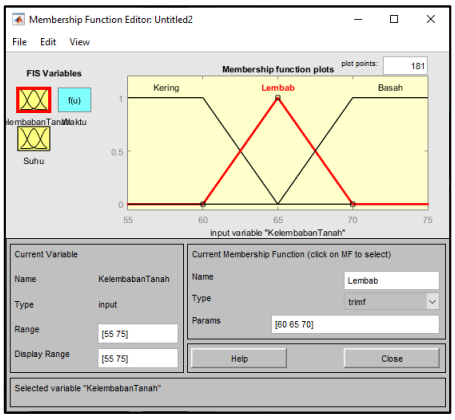
Kode 4. 8 *Sourcecode* Kondisi *Relay* pada *Output* Pompa Air

```
1       void Config_Pompa (){
2        if ( keluaran == 3){
3           digitalWrite(Pompa,HIGH);
4           delay(3000);
5           digitalWrite(Pompa,LOW);
6           delay(3000);
7           }
8       else if ( keluaran == 4){
9           digitalWrite(Pompa,HIGH);
10          delay(4000);
11          digitalWrite(Pompa,LOW);
12          delay(4000);
13          }
14       else if ( keluaran == 5){
15          digitalWrite(Pompa,HIGH);
16          delay(5000);
17          digitalWrite(Pompa,LOW);
18          delay(5000);
19          }
20       else if ( keluaran == 6){
21          digitalWrite(Pompa,HIGH);
22          delay(6000);
23          digitalWrite(Pompa,LOW);
24          delay(6000);
25          }
```

```
26        else if ( keluaran == 7){
27            digitalWrite(Pompa,HIGH);
28            delay(7000);
29            digitalWrite(Pompa,LOW);
30            delay(7000);
31            }
32         else if ( keluaran == 8){
33            digitalWrite(Pompa,HIGH);
34            delay(8000);
35            digitalWrite(Pompa,LOW);
36            delay(8000);
37            }
38
39         else if ( keluaran == 9){
40            digitalWrite(Pompa,HIGH);
41            delay(9000);
42            digitalWrite(Pompa,LOW);
43            delay(9000);
44            }
45  }
```

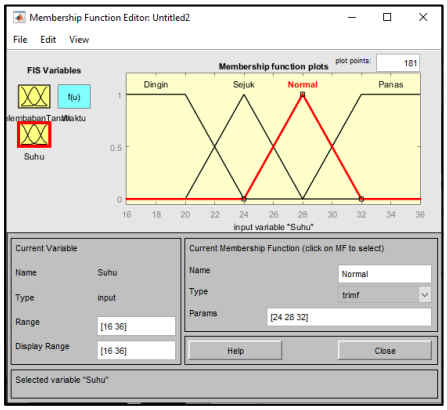
4.3 Pengujian *Fuzzy* Sugeno dengan Matlab

Pengujian metode *Fuzzy* Sugeno menggunakan Matlab untuk melihat proses metode fuzzy dari data yang telah didapatkan dan juga sebagai perbandingan pada simulasi pengujian dengan menggunakan alat yang telah dibuat.



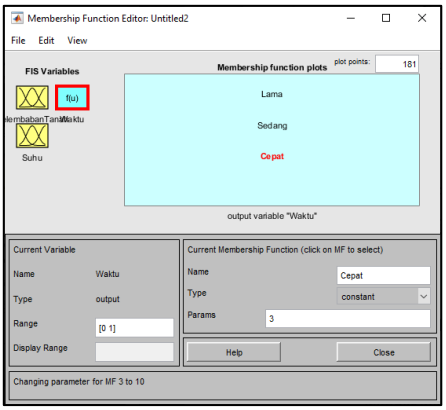
Gambar 4. 15 Keanggotaan Input Kelembaban Tanah pada Matlab

Pada Gambar 4.15 diatas merupakan tampilan fungsi keanggotaan inputan kelembaban tanah menggunakan aplikasi Matlab, pada inputan kelembaban tanah dijabarkan menjadi 3 buah *membership function* yaitu kering, lembab, dan basah. Pada fungsi keanggotaan inputan kelembaban tanah memiliki range pengukuran mulai dari 55 sampai 75 yang menjadi rata-rata kelembaban minimal dan maksimal.



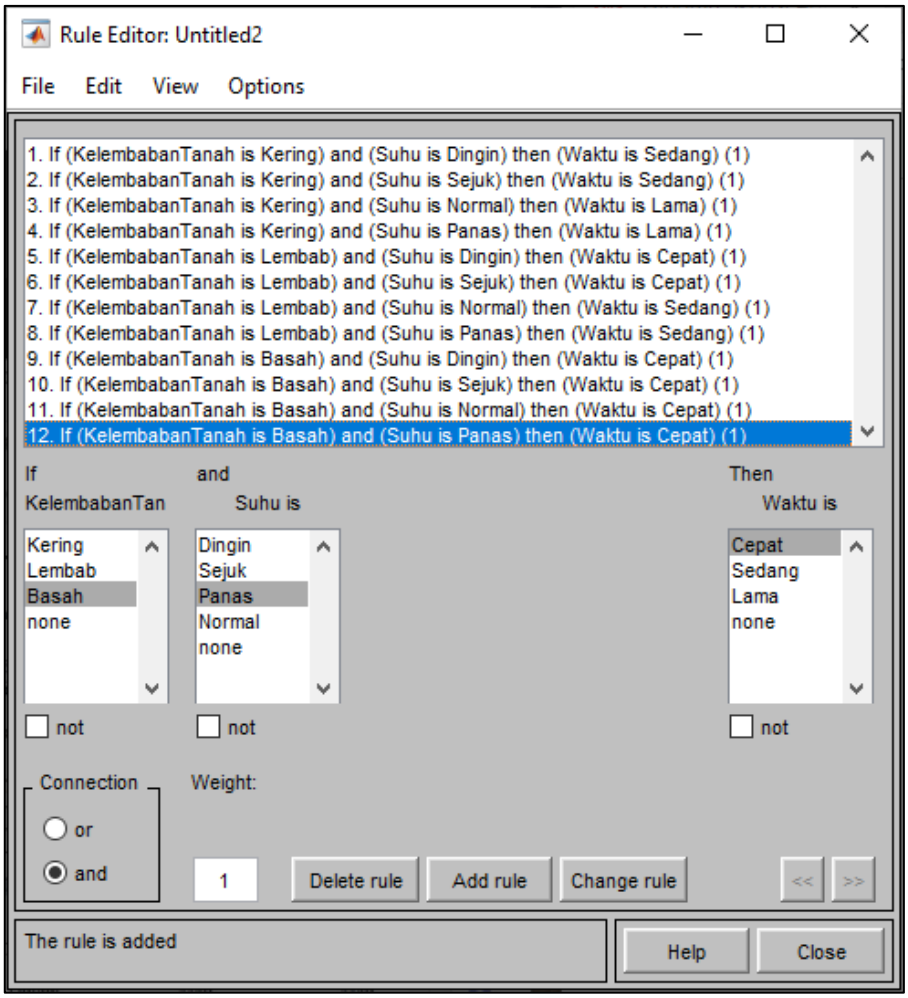
Gambar 4. 16 Keanggotaan Input Suhu pada Matlab

Pada Gambar 4.16 diatas merupakan tampilan fungsi keanggotaan inputan suhu menggunakan aplikasi Matlab, pada inputan suhu dijabarkan menjadi 4 buah *membership function* yaitu dingin, sejuk, normal dan dingin. Pada fungsi keanggotaan inputan suhu memiliki range pengukuran mulai dari 16 sampai 36 yang menjadi rata-rata suhu minimal dan maksimal.



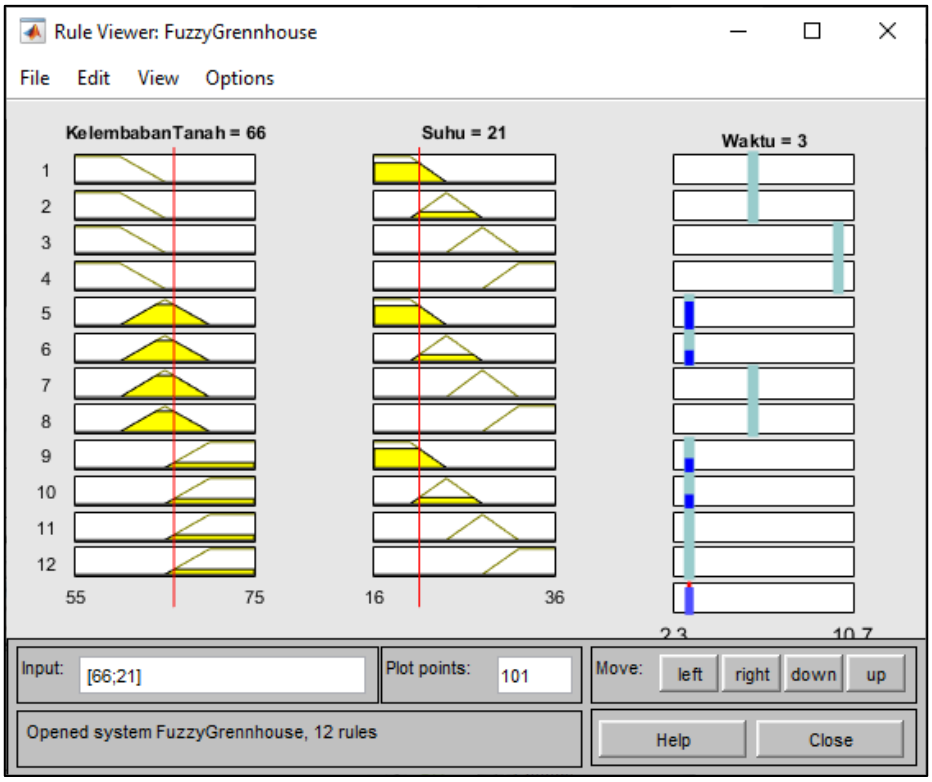
Gambar 4. 17 Keanggotaan Variabel Waktu pada Matlab

Pada Gambar 4.17 diatas merupakan tampilan fungsi keanggotaan variabel waktu menggunakan aplikasi Matlab, pada variabel waktu dibagi menjadi 3 range pengukuran yaitu 3 detik, 6 detik dan 9 detik. Nilai tersebut mempresentasikan tiap fungsi keanggotaan inputan suhu yang dijabarkan menjadi 3 buah *membership function* yaitu cepat, sedang, dan lama.



Gambar 4. 18 Aturan *Fuzzy* pada Matlab

Pada Gambar 4.18 diatas merupakan tampilan aturan fuzzy menggunakan aplikasi Matlab, Proses aturan *fuzzy* atau Inferensi merupakan proses penggabungan aturan berdasarkan data yang didapat dari hasil *fuzzifikasi* akan terbentuk 12 himpunan *fuzzy*.



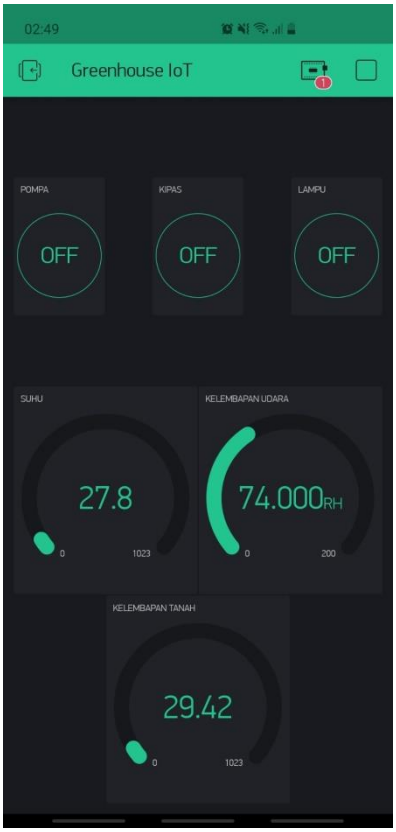
Gambar 4. 19 Hasil *output fuzzy* pada Matlab

Pada Gambar 4.19 diatas merupakan tampilan hasil *defuzzifikasi* menggunakan aplikasi Matlab. Dari hasil yang diperoleh menggunakan aplikasi Matlab dengan nilai kelembaban tanah sebesar 66% dan suhu sebesar 21°C maka akan mengeluarkan hasil waktu 3 detik. Jika dibandingkan dengan perhitungan yang telah dilakukan secara manual dengan penggunaan aplikasi, hasil yang dikeluarkan pada kedua perhitungan menunjukan hasil yang sama.

4.4 Pengujian Aplikasi Blynk

Pada pengujian aplikasi Blynk akan menguji fungsi dari pemantauan dan pengontrolan pada sistem *greenhouse* yang telah dibuat, tujuan dari penggunaan aplikasi ini untuk memantau kondisi lingkungan pada maket *greenhouse* yang telah terhubung dengan alat dan juga melakukan pengontrolan pada setiap keluaran sistem yang telah dibuat seperti kipas angin, led *strip* dan pompa air.

4.4.1 Tampilan Awal Projek Aplikasi Blynk

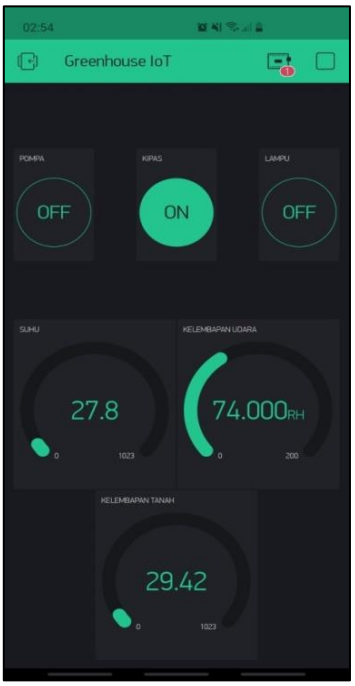


Gambar 4. 20 Tampilan Awal Projek *Greenhouse* Aplikasi Blynk

Pada Gambar 4.20 diatas merupakan tampilan halaman awal projek *greenhouse* berbasis IoT yang menggunakan aplikasi Blynk. Pada halaman tersebut terdapat beberapa tampilan pemantauan kondisi lingkungan yang dilakukan secara *real time*, tampilan pemantauan kondisi lingkungan yang dicakupi yaitu suhu udara pada *greenhouse*, kelembaban udara dan kelembaban pada tanah. Data yang ditampilkan pada tampilan suhu udara dan kelembaban udara diambil menggunakan sensor suhu DHT11 sedangkan data pada tampilan kelembaban tanah menggunakan sensor *soil moisture*. Pada tampilan halaman awal projek *greenhouse* berbasis IoT juga terdapat tiga tombol pengontrolan *output* pada rangkaian sistem yang dibuat yaitu kipas angin yang digunakan

pada sistem pendinginan suhu, led *strip* yang digunakan pada sistem pencahayaan dan pompa air pada sistem penyiraman.

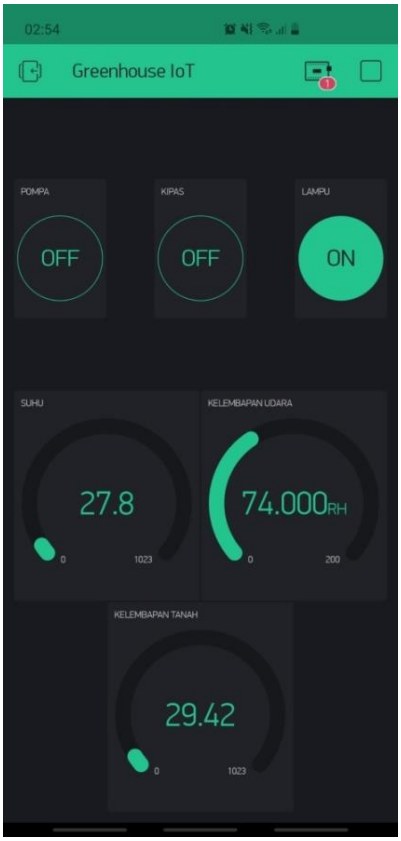
4.4.2 Pengontrolan Sistem Pendinginan Suhu



Gambar 4. 21 Tombol Pada *Output* Kipas

Pada Gambar 4.21 diatas merupakan tombol pengontrolan pada sistem pendinginan suhu yang terdapat pada tampilan halaman awal projek *greenhouse* berbasis IoT yang menggunakan aplikasi Blynk. Penggunaan tombol tersebut merupakan tombol ON dan OFF pada *outputan* kipas yang terhubung langsung pada perangkat yang dibuat. Untuk menyalakan *outputan* kipas, langkah yang perlu dilakukan yaitu menekan tombol hijau yang bertuliskan on lalu tombol tersebut akan menjadi hitam dan bertuliskan off. Pada saat menyala maka proses penentuan kondisi relay pada *outputan* kipas akan berjalan sesuai dengan kondisi yang ditentukan. Pada saat menekan tombol hitam yang bertuliskan off lalu tombol tersebut akan menjadi hijau dan bertuliskan on maka *outputan* kipas akan mati.

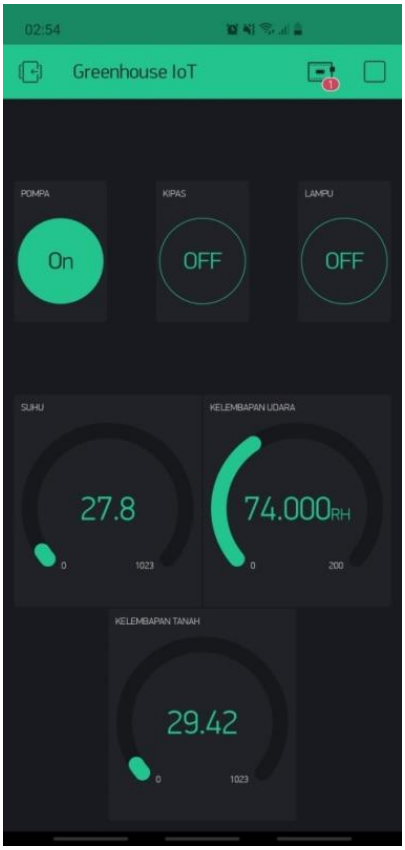
4.4.3 Pengontrolan Sistem Pencahayaan



Gambar 4. 22 Tombol Pada Output LED *Strip*

Pada Gambar 4.22 diatas merupakan tombol pengontrolan pada sistem pencahayaan yang terdapat pada tampilan halaman awal projek *greenhouse* berbasis IoT yang menggunakan aplikasi Blynk. Penggunaan tombol tersebut merupakan tombol ON dan OFF pada *outputan* led *strip* yang terhubung langsung pada perangkat yang dibuat. Untuk menyalakan *outputan* led *strip*, langkah yang perlu dilakukan yaitu menekan tombol hijau yang bertuliskan on lalu tombol tersebut akan menjadi hitam dan bertuliskan off. Pada saat menyala maka proses penentuan kondisi relay pada *outputan* led *strip* akan berjalan sesuai dengan kondisi yang ditentukan. Pada saat menekan tombol hitam yang bertuliskan off lalu tombol tersebut akan menjadi hijau dan bertuliskan on maka *outputan* led *strip* akan mati.

4.4.4 Pengontrolan Sistem Penyiraman



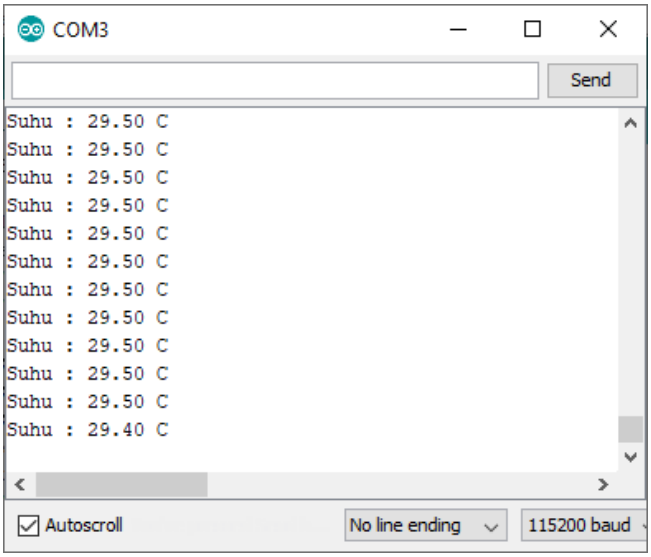
Gambar 4. 23 Tombol Pada Output Pompa Air

Pada Gambar 4.23 diatas merupakan tombol pengontrolan pada sistem penyiraman yang terdapat pada tampilan halaman awal projek *greenhouse* berbasis IoT yang menggunakan aplikasi Blynk. Penggunaan tombol tersebut merupakan tombol ON dan OFF pada *outputan* pompa air yang terhubung langsung pada perangkat yang dibuat. Untuk menyalakan *outputan* pompa air, langkah yang perlu dilakukan yaitu menekan tombol hijau yang bertuliskan on lalu tombol tersebut akan menjadi hitam dan bertuliskan off. Pada saat menyala maka proses penentuan kondisi relay pada *outputan* kipas angin akan berjalan sesuai dengan kondisi yang ditentukan. Pada saat menekan tombol hitam yang bertuliskan off lalu tombol tersebut akan menjadi hijau dan bertuliskan on maka *outputan* pompa air akan mati.

4.5 Pengujian Fungsional Alat

Pengujian fungsional merupakan pengujian pada tiap bagian alat yang telah dirangkai yang digunakan pada penelitian kali ini. Pengujian fungsional juga memiliki tujuan untuk mengetahui apakah alat atau komponen yang telah dirangkai dan disatukan bekerja sesuai dengan yang diinginkan.

4.5.1 Pengujian Sensor Suhu DHT11



Gambar 4. 24 Deteksi Suhu dengan Sensor Suhu DHT11 pada Aplikasi Arduino IDE

Gambar 4. 25 Deteksi Suhu dengan *Thermometer*

Table 4. 1 Hasil Pengujian Suhu dengan Sensor Suhu DHT11 dan *Thermometer*

No	Data Sensor Suhu DHT11 (°C)	Thermometer (°C)	Error (%)
1	28.31	28	1.10
2	31.28	31	0.90
3	33.63	33	1.9
4	30.15	30	0.5
5	26.07	26	0.26
6	27.25	27	0.92
7	30.08	30	0.26
8	31.28	31	0.90
9	29. 12	29	0.41
10	27.74	27	2.74
Rata – rata Error			0.989 %

Pada tabel 4.1 diatas dapat dilihat bahwa dari hasil pengujian sensor suhu yang menggunakan sensor suhu DHT11 pada Arduino IDE seperti Gambar 4.24 yang memiliki hasil pembacaan yang kurang lebih sama dengan penggunaan *thermometer* seperti Gambar 4.25 sebagai pembanding suhu yang dilakukan dengan waktu yang bersamaan. Hasil dari pengujian akan dimasukkan kedalam tabel dimana pengujian pembacaan dengan menggunakan sensor suhu DHT11 dan *thermometer* yang dilakukan sebanyak 10 kali, setelah mendapatkan nilai suhu maka akan dilakukan perhitungan presentasi *error* pada setiap pengujian lalu hasil kelesuruhan *error* akan dibuat rata-rata *error*. Berikut merupakan perhitungan presentase *error* pada pengujian sensor suhu DHT11, perhitungan sensor suhu DHT11 bernilai 28.31 dan *thermometer* bernilai 28.

$$Error\ (\%) = \frac{\text{Nilai DHT11} - \text{Nilai } Thermometer}{\text{Nilai } Thermometer}\ x\ 100\%$$

$$Error\ (\%) = \frac{28.31 - 28}{28}\ x\ 100\%$$

$$Error\ (\%) = \frac{0.31}{28}\ x\ 100\%$$

$Error$ (%) = 1.10%

Perhitungan rata – rata *error rate* saat melakukan pengujian sensor suhu DHT11

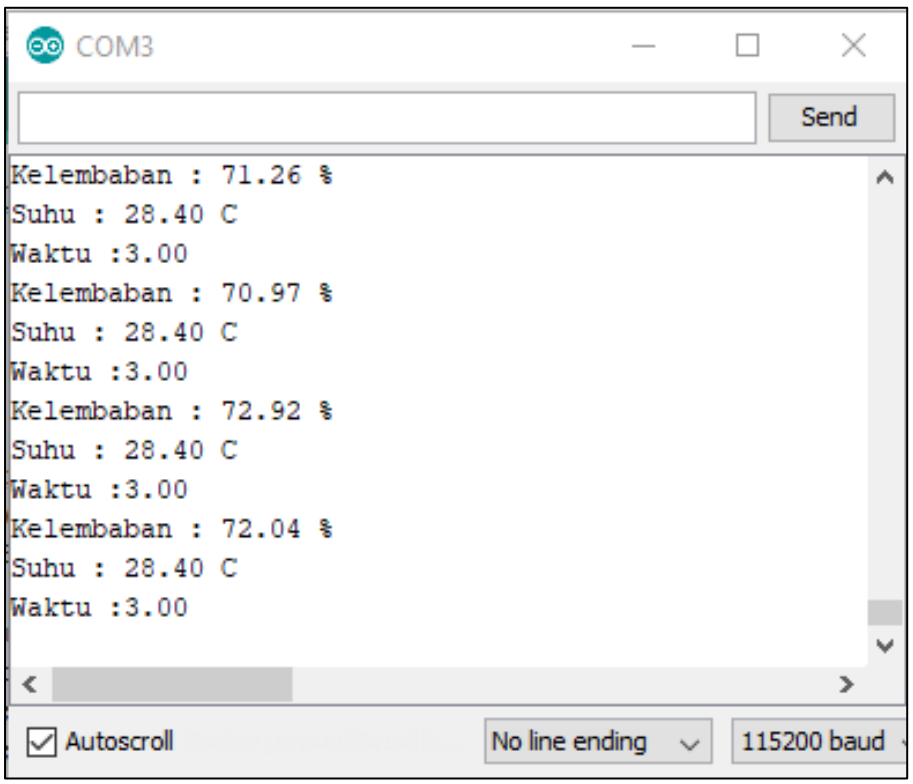
Rata – rata $Error$ (%) = $\frac{\sum error}{\sum uji\ coba}$

Rata – rata $Error$ (%) = $\frac{9.89}{10}$

Rata – rata $Error$ (%) = 0.989 %

Dari hasil perhitungan telah dilakukan pada pengujian tersebut maka dapat disimpulkan sensor suhu DHT11 memiliki nilai rata - rata *error* sebesar 0.989 %**,** maka dapat disimpulkan sensor suhu DHT11 berjalan baik dan sesuai dengan yang diinginkan.

4.5.2 Pengujian Sensor Kelembaban Tanah *Soil Moisture*



Gambar 4. 26 Deteksi Kelembaban Tanah dengan Sensor *Soil Moisture* pada Aplikasi Arduino IDE

Table 4. 2 Hasil Pengujian Kelembaban dengan Sensor Kelembaban Tanah

No	Volume Air (ml)	Nilai Sensor Kelembaban Tanah	Kelembaban Tanah (%)
1	30	788.5	22.9
2	60	648.3	36.6
3	90	503	49.5
4	120	439.7	57
5	150	362	64.6
6	180	302.7	70.4
7	210	261.4	74.4

Pada tabel 4.2 diatas dapat dilihat bahwa dari hasil pengujian sensor kelembaban tanah yang menggunakan sensor *soil moisture* pada Arduino IDE seperti pada Gambar 4.26 diatas. Hasil dari pengujian akan dimasukkan kedalam tabel dimana pengujian pembacaan dengan menggunakan sensor *soil moisture* dengan wadah pot berisi tanah lalu dilakukanya pemberian air dengan kelipatan 30ml pada pot tersebut dimana pengujian dilakukan sebanyak 7 kali, setelah mendapatkan nilai kelembaban tanah maka akan dilakukan perhitungan perubahan data sensor menjadi data presentase pada setiap pengujian. Berikut merupakan perhitungan presentase kelembaban tanah pada pengujian dengan volume air sebanyak 30ml dengan nilai sensor kelembaban tanah bernilai 788.5 dan nilai ADC dari sensor kelembaban tanah menggunakan 10 bit bernilai 1023.

$$\text{Kelembaban Tanah (\%)} = \frac{\text{Nilai ADC} - \text{Nilai Sensor Kelembaban Tanah}}{\text{Nilai ADC}}$$

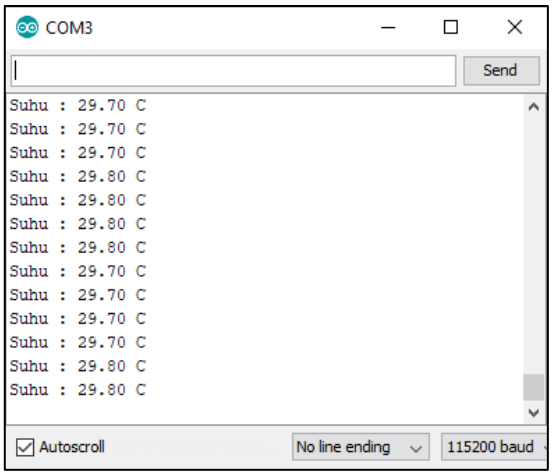
$$\text{Kelembaban Tanah (\%)} = \frac{1023 - 788.5}{1023}$$

Kelembaban Tanah (%) = 22.9%

Dari hasil perhitungan telah dilakukan pada pengujian tersebut maka dapat disimpulkan nilai sensor kelembaban tanah menjadi nilai presentasi yang

nantinya dapat di proses pada sistem**,** maka dapat disimpulkan sensor kelembaban tanah *soil moisture* berjalan baik dan sesuai dengan yang diinginkan.

4.5.3 Pengujian Sistem Kontrol Pendinginan Suhu



Gambar 4. 27 Deteksi Suhu dengan Sensor Suhu DHT11 pada Aplikasi Arduino IDE

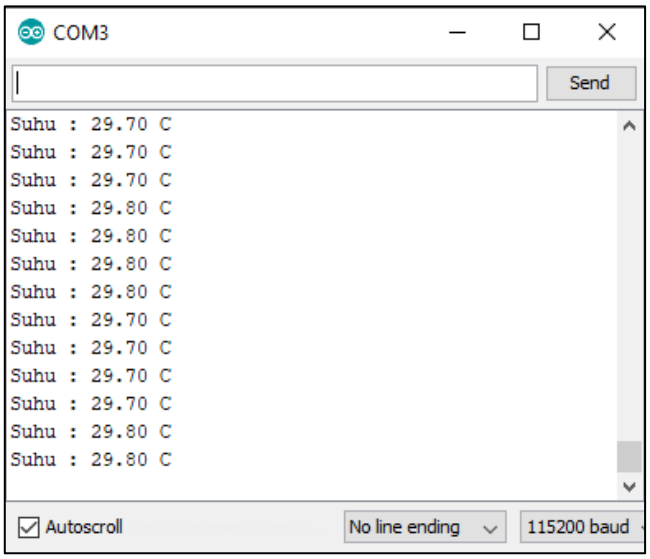
Table 4. 3 Hasil Pengujian Sistem Kontrol Pendinginan Suhu dengan Sensor Suhu DHT11 pada Alat yang telah dibuat

Pengujian	No. pengujian	Waktu Pengujian	Sensor Suhu DHT11 (°C)	Kondisi Kipas Angin
1	1	07.00	28	Mati
	2	12.00	33	Nyala
	3	16.00	30	Nyala
	4	20.00	26	Mati
2	1	07.00	27	Mati
	2	12.00	31	Nyala
	3	16.00	29	Nyala
	4	20.00	27	Mati

Pada Table 4.3 diatas dapat dilihat bahwa dari hasil pengujian sistem kontrol pendinginan suhu menggunakan sensor suhu DHT11 pada Arduino IDE seperti Gambar 4.27 diatas. Kondisi kipas angin telah berjalan secara otomatis dan bekerja dengan baik, kondisi kipas angin berjalan sesuai dengan kondisi

yang telah di tentukan dari data yang telah diambil oleh sensor suhu. Hasil dari pengujian akan dimasukkan kedalam tabel dimana pengujian yang dilakukan selama 2 hari dengan pembagian waktu yaitu pagi pukul 07.00 WIB, siang pukul 12.00 WIB, sore pukul 16.00 WIB dan 20.00 WIB. Dari hasil yang telah didapat dari pengujian tersebut maka dapat disimpulkan sistem kontrol pendinginan suhu berjalan baik pada *greenhouse* dan sesuai dengan yang diinginkan.

4.5.4 Pengujian Sistem Kontrol Pencahayaan



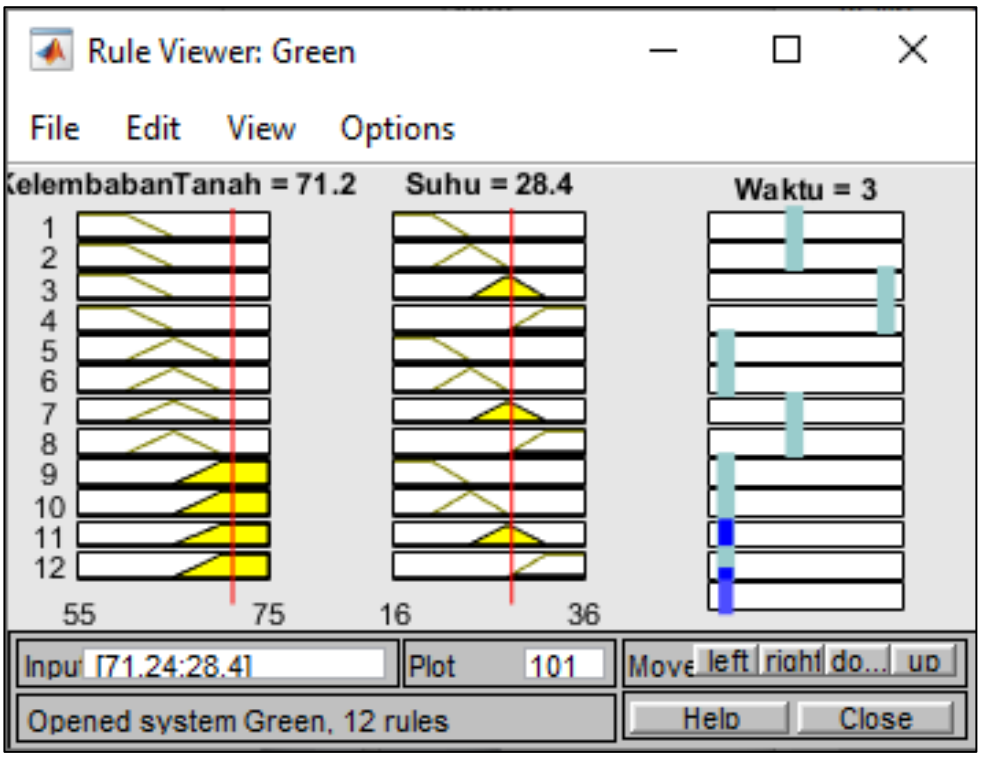
Gambar 4. 28 Deteksi Suhu dengan Sensor Suhu DHT11 pada Aplikasi Arduino IDE

Table 4. 4 Hasil Pengujian Sistem Kontrol Pencahayaan dengan Sensor Suhu DHT11 pada Alat yang telah dibuat

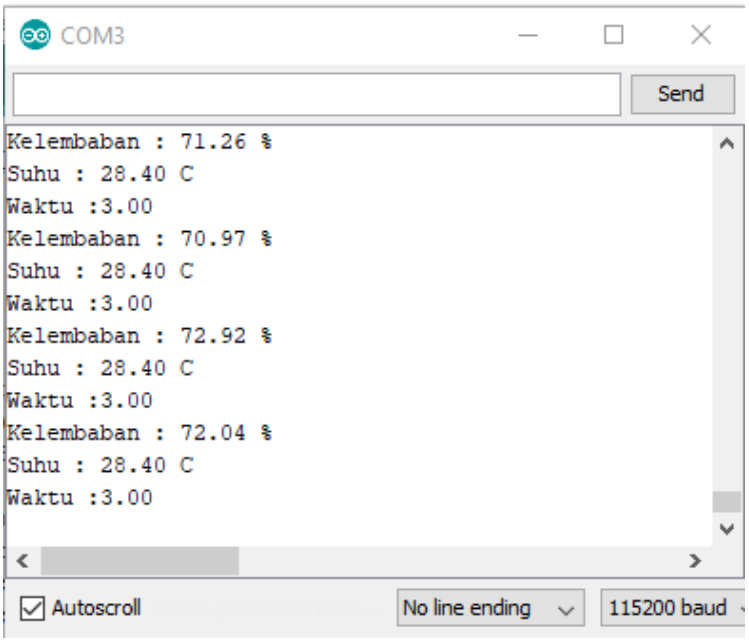
Pengujian	No. pengujian	Waktu Pengujian	Sensor Suhu DHT11 (°C)	Kondisi LED *Strip*
1	1	07.00	28	Nyala
	2	12.00	33	Mati
	3	16.00	30	Mati
	4	20.00	26	Nyala
2	1	07.00	27	Nyala
	2	12.00	31	Mati
	3	16.00	29	Nyala
	4	20.00	27	Nyala

Pada Table 4.4 diatas dapat dilihat bahwa dari hasil pengujian sistem kontrol pencahayaan menggunakan sensor suhu DHT11 pada Arduino IDE seperti Gambar 4.28 diatas. Kondisi led *strip* telah berjalan secara otomatis dan bekerja dengan baik, kondisi led *strip* berjalan sesuai dengan kondisi yang telah di tentukan dari data yang telah diambil oleh sensor suhu. Hasil dari pengujian akan dimasukkan kedalam tabel dimana pengujian yang dilakukan selama 2 hari dengan pembagian waktu yaitu pagi pukul 07.00 WIB, siang pukul 12.00 WIB, sore pukul 16.00 WIB dan 20.00 WIB. Dari hasil yang telah didapat dari pengujian tersebut maka dapat disimpulkan sistem kontrol pencahayaan berjalan baik pada *greenhouse* dan sesuai dengan yang diinginkan.

4.5.5 Pengujian Sistem Kontrol Penyiraman



Gambar 4. 29 Hasil Perhitungan Simulasi *Fuzzy* pada Aplikasi Matlab



Gambar 4. 30 Hasil Perhitungan Simulasi *Fuzzy* pada Aplikasi Arduino IDE

Table 4. 5 Hasil Pengujian Sistem Kontrol Penyiraman dengan membandingkan Hasil Simulasi Matlab dengan Alat yang telah dibuat dengan Arduino IDE

Pengujian	Jam	Hasil Pengukuran		Status Pompa Air	Hasil Perhitungan Simulasi Arduino	Hasil Keluaran Simulasi Matlab	Error (%)
		Suhu (°C)	Kelembaban Tanah (%)				
Hari ke 1	08.00	29	65	5 detik	5.63	5.79	2.7
	12.00	33	62	7 detik	7.86	7.91	0.6
	17.00	28	67	4 detik	4.37	4.39	0.4
Hari ke 2	08.00	28	64	6 detik	6.16	6.23	1.1
	12.00	32	65	6 detik	6.00	6.00	0
	17.00	30	66	4 detik	4.77	4.89	2.4
Hari ke 3	08.00	27	68	3 detik	3.59	3.61	0.5
	12.00	31	64	7 detik	6.84	7.07	3.2
	17.00	28	69	3 detik	3.56	3.62	1.6
Rata – rata Error							1.3%

Pada Table 4.5 diatas dapat merupakan hasil pengujian sistem kontrol penyiraman menggunakan logika *fuzzy* yang bertujuan untuk mengetahui apakah kerja dari sistem logika *fuzzy* pada penyiraman berjalan atau tidak. Pengujian tersebut dimodelkan sebagai ketepatan lama waktu penyiraman yang merupakan *output* pada sistem yang telah dibuat dan diterapkan pada mikrokontroller, hasil yang didapat dari logika *fuzzy* yang telah diterapkan pada Arduino IDE seperti pada Gambar 4.29 akan dibandingkan dengan hasil keluaran pada simulasi menggunakan matlab seperti Gambar 4.30 diatas. Hasil dari pengujian akan dimasukkan kedalam tabel dimana pengujian yang dilakukan selama 3 hari dengan pembagian waktu yaitu pagi pukul 08.00 WIB, siang pukul 12.00 WIB dan sore pukul 17.00 WIB, setelah mendapatkan nilai keluaran sistem dan nilai keluaran matlab maka akan dilakukan perhitungan presentasi *error* pada setiap pengujian lalu hasil kelesuruhan *error* akan dibuat rata-rata *error*. Berikut merupakan perhitungan presentase *error* pada pengujian sistem kontrol penyiraman menggunakan metode *fuzzy*, perhitungan keluaran data sistem yang bernilai 5.63 dan keluaraan data matlab yang bernilai 5.79

$$Error\ (\%) = \frac{\text{Nilai Arduino} - \text{Nilai Matlab}}{\text{Nilai Matlab}}\ x\ 100\%$$

$$Error\ (\%) = \frac{5.63 - 5.79}{5.79}\ x\ 100\%$$

$$Error\ (\%) = \frac{0.16}{5.79}\ x\ 100\%$$

$$Error\ (\%) = 2.7\%$$

Perhitungan rata – rata *error rate* saat melakukan pengujian sistem kontrol penyiraman

$$\text{Rata – rata } Error\ (\%) = \frac{\sum error}{\sum uji\ coba}$$

Rata – rata $Error$ (%) = $\frac{12.5}{9}$

Rata – rata $Error$ (%) = 1,3%

Dari hasil perhitungan telah dilakukan pada pengujian tersebut maka dapat disimpulkan pengujian sistem kontrol penyiraman dengan menggunakan metode *fuzzy* memiliki nilai *error* sebesar 1,3%, maka dapat disimpulkan sistem kontrol penyiraman dengan menggunakan metode *fuzzy* masih dapat dikatakan berjalan dengan baik dan normal sehingga sesuai dengan yang diinginkan.