

BAB IV

HASIL PENGUJIAN DAN PENGAMATAN

Dalam bab ini penulis akan menguraikan dan menjelaskan hasil analisis pengujian dari hasil penelitian tugas akhir ini yang telah dilakukan, pengujian dilakukan dalam beberapa bagian yang disusun dalam urutan dari yang sederhana menuju sistem yang lengkap. Pengujian dilakukan meliputi pengujian perangkat lunak (*software*) aplikasi *Android* dan perangkat keras (*hardware*) meliputi *Arduino Uno* dan *wireless module ESP8266*, diharapkan didapat suatu sistem yang dapat menjalankan rancangan alat berjalan dengan baik, optimal, dan bermanfaat.

4.1 Pengujian Arduino Uno

4.1.1 Tujuan Pengujian

Pengujian perangkat *Arduino Uno* bertujuan untuk mengetahui apakah dapat berjalan dengan baik dan memastikan *Arduino Uno* yang dipakai tidak ada kerusakan sehingga dapat digunakan sesuai yang diharapkan.

4.1.2 Alat Yang Dibutuhkan

1. Komputer/Laptop
2. *Wireless Module ESP8266*
3. *Arduino Uno R3*
4. Kabel serial

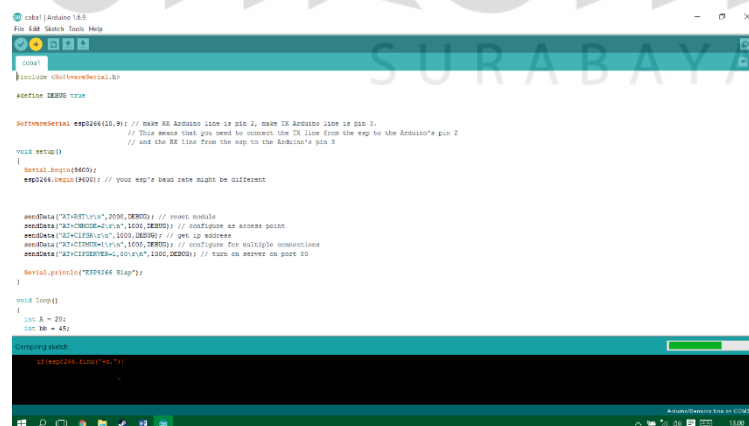
4.1.3 Prosedur Pengujian

1. Hubungkan *Arduino Uno* dengan *ESP8266* sesuai dengan pin yang telah dibuat.

2. Selanjutnya nyalakan komputer/laptop kemudian jalankan program Arduino IDE.
3. Sambungkan kabel serial dengan komputer/laptop.
4. Sambungkan Arduino Uno dengan kabel kabel serial.
5. Buka sketch yang akan digunakan untuk di *upload* kedalam Arduino Uno.
6. Setting board, serial port dan programmer sesuai dengan yang digunakan.
7. Kemudian upload sketch dan tunggu hingga selesai.
8. Setelah upload selesai akan diketahui apakah program berhasil di *download* atau tidak oleh Arduino Uno.

4.1.4 Hasil Pengujian

Dari percobaan di atas dapat diperoleh apabila terjadi proses *upload* program tidak ada *comment* yang menunjukkan kegagalan atau tidak ada *comment error* dalam sambungan antara kabel serial dan Arduino Uno dapat dilihat pada gambar 4.1. Apabila proses *upload* program berjalan dengan baik maka di tandai dengan tampil *comment* seperti yang di tunjukan pada gambar 4.2.



```

coba1 | Arduino 1.8.2
File Edit Sketch Tools Help

code
#include <SoftwareSerial.h>

#define DEBUG true

SoftwareSerial esp2466(10, 9); // make RX Arduino line 10 pin 2, make TX Arduino line 10 pin 3.
// This means that you need to connect the TX line from the esp to the Arduino's pin 2
// and the RX line from the esp to the Arduino's pin 3

void setup()
{
  Serial.begin(9600);
  esp2466.begin(9600); // your esp's baud rate might be different

  pinMode(13, OUTPUT); // reset module
  pinMode(12, INPUT); // configure as access point
  pinMode(11, INPUT); // get ip address
  pinMode(10, INPUT); // configure for multiple connections
  pinMode(9, INPUT); // turn on server on port 80
  Serial.println("ESP8266 Start");
}

void loop()
{
  int A = 20;
  int B = 40;
}
Compiling sketch
11:00:00, 11:00:01

```

Gambar 4.1 Tampilan Proses *Upload* dari Arduino IDE

```

cobat | Arduino 1.6.9
File Edit Sketch Tools Help

cobat

// This means that you need to connect the TX line from the esp to the Arduino's pin 2
// and the RX line from the esp to the Arduino's pin 3

void setup()
{
  Serial.begin(9600);
  esp266.begin(9600); // your esp's baud rate might be different

  sendData("AT+RESTORE",2000,DEBUG); // reset module
  sendData("AT+DNSMODE=1",1000,DEBUG); // configure as access point
  sendData("AT+CWMODE=1",1000,DEBUG); // get ip address
  sendData("AT+CIPMUX=1",1000,DEBUG); // configure for multiple connections
  sendData("AT+CIPSERVER=1,80",1000,DEBUG); // turn on server on port 80
  Serial.println("ESP266 Slap");
}

void loop()
{
  int A = 20;
  int B = 40;
  float C = 40;
  float D = 80;
  float E = 30;
  float F = 50;
  if(esp266.available()) // check if the esp is sending a message
  {
    // Data uploading
    Serial.println("Found 10. C:\Users\Doan\Documents\Arduino\libraries\esp266-relay-master\ C:\Users\Doan\Documents\Arduino\libraries\esp266-relay-master");
    Serial.println("Found 10. C:\Users\Doan\Documents\Arduino\libraries\MiniWebServerCommandQw\Version01_08_15\ C:\Users\Doan\Documents\Arduino\libraries\MiniWebServerCommandQw\Version01_08_15");
    Serial.println("Found 10. C:\Users\Doan\Documents\Arduino\libraries\esp266-relay-master\ C:\Users\Doan\Documents\Arduino\libraries\esp266-relay-master");
    Serial.println("Found 10. C:\Users\Doan\Documents\Arduino\libraries\MiniWebServerCommandQw\Version01_08_15\ C:\Users\Doan\Documents\Arduino\libraries\MiniWebServerCommandQw\Version01_08_15");
  }
}

```

Gambar 4.2 Tampilan *Comment* saat Program Berhasil di *Upload*

4.2 Pengujian Android

4.2.1 Tujuan Pengujian

Pengujian aplikasi Android “greenhouse.apk” bertujuan untuk mengetahui apakah aplikasi Android dapat berkomunikasi dengan ESP8266 yang kemudian dapat mengirimkan data kedalam Arduino Uno sehingga komunikasi dinyatakan berjalan dengan baik.

4.2.2 Alat yang Dibutuhkan

1. Laptop/komputer
2. Smartphone Android
3. Kabel Serial
4. Breadboard
5. *Wireless Module ESP8266*
6. Kabel Jumper

4.2.3 Prosedur Pengujian

1. Hubungkan Arduino Uno dengan *Wireless Module ESP8266* dengan bantuan menggunakan kabel jumper dan Breadboard.
2. Nyalakan komputer/laptop.

3. Hubungkan Arduino Uno dengan komputer/laptop menggunakan kabel serial.
4. Buka aplikasi Arduino IDE, kemudian buka sketch yang akan di upload.
5. Tekan upload pada aplikasi Arduino IDE dan tunggu hingga proses upload selesai.
6. Setelah itu siapkan Smartphone Android dan buka Aplikasi “greenhouse.apk” yang sudah terinstall pada Smartphone Android.
7. Kemudian pada aplikasi “greenhouse.apk” masukkan IP Address sesuai dengan IP Address yang dimiliki *Wireless Module ESP8266*.
8. Lakukan penekanan tombol yang terdapat pada aplikasi “greenhouse.apk”.

4.2.4 Hasil Pengujian

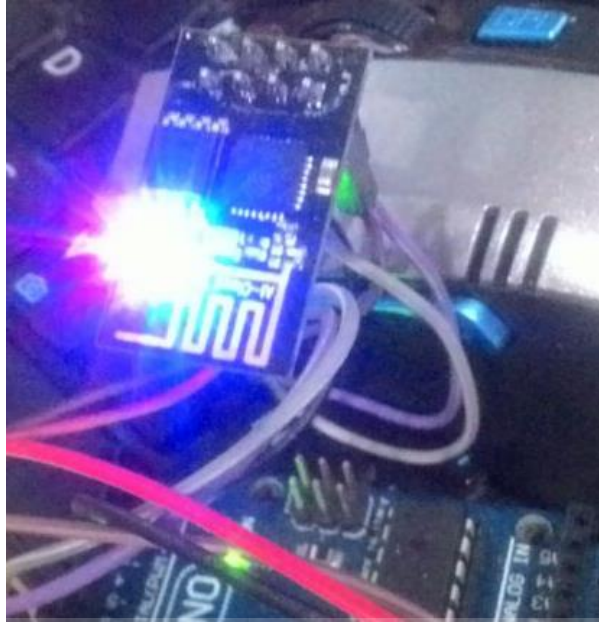
Dari percobaan di atas apabila sudah melakukan upload maka akan tampil pada serial monitor yang menyatakan bahwa *Wireless Module ESP8266* sudah siap digunakan untuk berkomunikasi dan aplikasi Android dapat berkomunikasi dengan Arduino Uno dengan melihat pada gambar 4.3. Saat *Wireless Module ESP8266* mendapatkan sinyal komunikasi dari aplikasi Android, maka ditandai dengan *Wireless Module ESP8266* akan berkedip menyala berwarna biru ketika tombol tertentu pada aplikasi Android dilakukan penekanan, dapat lihat pada gambar 4.4.

```

COM3 (Arduino/Genuino Uno)
|
| Send
|
AT+RST
OK
bb0†ÀùSbáýS-Š*) ÒnyÈÈÈ#Dìys%†
[Vendor:www.ai-thinker.com Version:0.9.2.4]
ready
AT+CRMODE=2
no change
AT+CIPFSR
192.168.4.1
OK
AT+CIPMUX=1
OK
AT+CIPSERVER=1,80
OK
ESP8266 Siap
 Autoscroll
Both NL & CR 9600 baud

```

Gambar 4.3 Tampilan Serial Monitor Saat *Wireless Module ESP8266* siap



Gambar 4.4 *Wireless Module ESP8266* Berhasil Berkomunikasi

4.3 Pengujian Penekanan Tombol

4.3.1 Tujuan Pengujian

Dalam pengujian penekanan tombol ini bertujuan untuk mengetahui bagaimana respon data yang dikirimkan dari aplikasi *Android* ke Arduino Uno dan untuk mengetahui *delay* pada saat penekanan tombol pada *Android* untuk melakukan pengiriman data yang diterima oleh Arduino Uno. Sehingga dapat diketahui proses komunikasi saat dilakukan penekanan tombol.

4.3.2 Alat yang Dibutuhkan

1. Aplikasi *Android* “greenhouse.apk”
2. Arduino Uno
3. *Wireless Module ESP8266*
4. Laptop/komputer
5. Kabel Serial
6. *Smartphone Android*

4.3.3 Prosedur Pengujian

1. Nyalakan laptop/komputer kemudian buka aplikasi Arduino IDE untuk membuka program Arduino.
2. Hubungkan Arduino Uno dengan *Wireless Module ESP8266* sesuai dengan rangkaian yang telah dibuat.
3. Hubungkan rangkaian Arduino Uno dan *Wireless Module ESP8266* ke laptop/komputer dengan menggunakan kabel serial.
4. Pada aplikasi Arduino IDE upload program ke Arduino Uno dan tunggu *upload* sampai selesai kemudian buka *serial monitor* pada.
5. Install aplikasi *Android* “greenhouse.apk” pada smartphone.
6. Hubungkan koneksi *smartphone* dengan *Wireless Module ESP8266* dengan SSID “esp-duma” dan masukkan Password “dumareza123” tunggu sampai terhubung.
7. Buka aplikasi *Android* “greenhouse.apk” kemudian masukkan IP Address “192.168.4.1” tekan tombol bergambar centang dan tekan tombol **Go** untuk masuk pada *screen 2*.
8. Lakukan penekanan tombol “Input Data Kangkung” untuk memulai pengujian.

4.3.4 Hasil Pengujian

Dari prosedur diatas dilakukan pengujian dengan melakukan penekanan tombol untuk mengirimkan data dari aplikasi Android ke Arduino Uno melalui koneksi *Wireless Module ESP8266*. Untuk melakukan pengujian ini akan diambil sampel penekanan tombol sebanyak 30 kali untuk mengetahui respon yang diterima saat melakukan pengiriman data, dapat dilihat pada tabel 4.1.

Tabel 4.1 Hasil Pengujian Respon dan *Delay* Penekanan Tombol

Penekanan Tombol ke-	Respon	<i>Delay</i>
1	Ya	3 detik
2	Tidak	-
3	Ya	0 detik
4	Ya	2 detik
5	Ya	2 detik
6	Tidak	-
7	Ya	0 detik
8	Ya	2 detik
9	Ya	3 detik
10	Tidak	-
11	Ya	1 detik
12	Ya	1 detik
13	Tidak	-
14	Ya	1 detik
15	Ya	1 detik
16	Ya	1 detik
17	Tidak	-
18	Ya	1 detik
19	Ya	0 detik
20	Ya	0 detik
21	Ya	1 detik
22	Ya	1 detik
23	Tidak	-
24	Ya	2 detik
25	Ya	0 detik
26	Ya	1 detik
27	Ya	1 detik
28	Ya	1 detik
29	Ya	2 detik
30	Ya	0 detik

Dari 30 penekanan tombol hasil pengujian di atas terjadi koneksi tidak respon sebanyak 6 penekanan tombol yaitu pada penekanan tombol ke-2, 6, 10, 13, 17, dan 23. Pengujian ini dilakukan dalam jarak sekitar 1 meter antara smartphone dan *Wireless Module ESP8266*. Rata-rata delay untuk penekanan tombol yaitu 0.9 detik yang diperoleh dari menjumlahkan seluruh delay dibagi dengan banyaknya jumlah penekanan tombol.

4.4 Pengujian Transmisi Data *Monitoring*

4.4.1 Tujuan Pengujian

Dalam pengujian transmisi data *monitoring* ini bertujuan untuk mengetahui apakah data yang dikirimkan dari Arduino Uno dapat terkirim sesuai dengan data yang telah dibuat. Data tersebut akan dikirim oleh *Wireless Module ESP8266* ke *web server* melalui koneksi internet dan data tersebut akan ditampilkan pada aplikasi Android. Kemudian data yang telah dibuat di Arduino Uno akan dibandingkan apakah data tersebut sama dengan hasil yang ditampilkan pada aplikasi Android.

4.4.2 Alat yang Dibutuhkan

1. Laptop/Komputer
2. Arduino Uno
3. *Wireless Module ESP8266*
4. Kabel serial
5. *Smartphone Android*
6. Aplikasi “greenhouse.apk”
7. Koneksi Internet
8. *Web Server* (<http://api.thingspeak.com/>)

4.4.3 Prosedur Pengujian

1. Nyalakan laptop/komputer kemudian buka aplikasi Arduino IDE untuk membuka program Arduino Uno.
2. Hubungkan Arduino Uno dengan *Wireless Module ESP8266* sesuai dengan rangkaian yang telah dibuat.

3. Hubungkan rangkaian Arduino Uno ke laptop/komputer dengan menggunakan kabel serial.
4. Pada aplikasi Arduino IDE upload program yang telah dibuat ke Arduino Uno dan tunggu *upload* sampai selesai kemudian buka *serial monitor* pada Arduino Uno untuk melihat data yang dikirimkan.
5. Setelah itu buka *web server* <http://api.thingspeak.com> kemudian buat channel baru untuk menyimpan data yang telah dibuat pada Arduino Uno agar data tersebut tersimpan pada server.
6. Setelah data terkirim ke *web server*, buka aplikasi “greenhouse.apk” dan masuk pada *screen 3* yaitu info data suhu dan kelembaban dan amati data yang telah masuk.

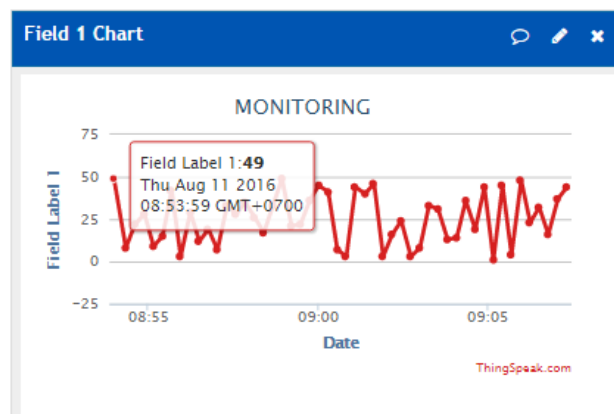
4.4.4 Hasil Pengujian

Dari prosedur pengujian diatas dilakukan dengan mengamati data yang ditampilkan di Arduino IDE pada serial monitor. Dengan menampilkan 50 data yang dikirimkan ke *web server*. Dalam pengujian ini dilakukan agar dapat mengirimkan data dari Arduino Uno melalui *Wireless Module ESP8266* untuk disimpan dalam *web server* dengan menggunakan koneksi internet. Setelah program yang telah dibuat pada Arduino Uno diupload maka akan tampil pada serial monitor pada gambar 4.5 berikut.



Gambar 4.5 Pengujian Transmisi Data Serial Monitor

Pada gambar diatas keluar sebanyak 50 data secara random yang kemudian dikirimkan oleh *Wireless Module ESP8266* ke *web server* melalui koneksi internet. Setelah data tersebut berhasil terkirim pada *web server* maka akan tampil data seperti pada gambar 4.6 berikut.



Gambar 4.6 Tampilan Data pada *web server* Transmisi Monitoring

Setelah data tersebut sudah tersimpan pada *web server* maka data tersebut akan diambil oleh aplikasi *Android* melalui koneksi internet. Maka data pada Aplikasi *Android* akan sama dengan data yang telah tersimpan pada *web server* tersebut. Kemudian dapat dilakukan pengamatan dan pengambilan 50 data pada tabel 4.2 berikut ini.

Tabel 4.2 Hasil Pengujian Transmisi Data *Monitoring*

No.	Tampilan Serial Monitor	Waktu Terima Data	Tampilan Aplikasi <i>Android</i>	Waktu Terima Data
1.	49	08:53:59	49	08:54:00
2.	8	08:54:20	8	08:54:22
3.	22	08:54:36	22	08:54:37
4.	28	08:54:53	28	08:54:54
5.	9	08:55:09	9	08:55:10
6.	15	08:55:25	15	08:55:25
7.	42	08:55:41	42	08:55:42
8.	3	08:55:57	3	08:55:58
9.	29	08:56:14	29	08:56:15
10.	12	08:56:30	12	08:56:31
11.	19	08:56:46	19	08:56:47
12.	7	08:57:02	7	08:57:03
13.	33	08:57:19	33	08:57:20
14.	28	08:57:35	28	08:57:36
15.	35	08:57:51	35	08:57:52
16.	26	08:58:08	26	08:58:09
17.	17	08:58:24	17	08:58:25
18.	33	08:58:40	33	08:58:41
19.	49	08:58:56	49	08:58:56
20.	21	08:59:13	21	08:59:14
21.	22	08:59:29	22	08:59:30
22.	36	08:59:45	36	08:59:46
23.	45	09:00:01	45	09:00:02
24.	41	09:00:18	41	09:00:19
25.	7	09:00:34	7	09:00:35
26.	3	09:00:50	3	09:00:51
27.	44	09:01:06	44	09:01:06
28.	40	09:01:23	40	09:01:24
29.	46	09:01:39	46	09:01:40
30.	3	09:01:55	3	09:01:56
31.	16	09:02:11	16	09:02:12
32.	24	09:02:27	24	09:02:28

33.	3	09:02:44	3	09:02:45
34.	8	09:03:00	8	09:03:01
35.	33	09:03:16	33	09:03:17
36.	31	09:03:32	31	09:03:33
37.	13	09:03:49	13	09:03:50
38.	14	09:04:05	14	09:04:06
39.	36	09:04:21	36	09:04:22
40.	19	09:04:38	19	09:04:39
41.	44	09:04:54	44	09:04:55
42.	1	09:05:10	1	09:05:11
43.	45	09:05:26	45	09:05:27
44.	4	09:05:42	4	09:05:43
45.	48	09:05:59	48	09:06:00
46.	23	09:06:15	23	09:06:15
47.	32	09:06:31	32	09:06:32
48.	16	09:06:47	16	09:06:48
49.	37	09:07:04	37	09:07:05
50.	44	09:07:20	44	09:07:21

Pada tabel hasil pengujian diatas dapat dilihat pengiriman 50 data 100% sukses diterima oleh aplikasi Android dengan selisih rata-rata waktu delay sebesar 0.9 detik dengan perhitungan sebagai berikut :

$$\text{Jumlah selisih waktu delay} = \text{Waktu Terima Data serial} - \text{Waktu Terima Data Aplikasi}$$

$$\text{Rata-rata delay} = \text{Jumlah selisih waktu delay} / \text{banyaknya data}$$

Sehingga,

$$\begin{aligned} \text{Rata-rata delay} &= 45 \text{ Detik} / 50 \\ &= 0.9 \text{ Detik} \end{aligned}$$

Jadi, rata-rata delay pada pengujian transmisi monitoring sebesar 0.9 detik.

4.5 Pengujian Transmisi Data *Setting*

4.5.1 Tujuan Pengujian

Dalam pengujian transmisi data *setting* ini bertujuan untuk mengetahui apakah data yang dikirimkan dari Android dapat terkirim sesuai dengan data yang

telah dibuat. Berbeda dari pengujian *monitoring*, data tersebut akan dikirim melalui *Wireless Module ESP8266* tanpa melalui *web server* dan koneksi internet. Data tersebut akan ditampilkan pada aplikasi serial monitor pada aplikasi Arduino IDE. Kemudian data yang telah diinputkan pada Android akan dibandingkan apakah data tersebut sama dengan hasil yang ditampilkan pada serial monitor.

4.5.2 Alat yang Dibutuhkan

1. Arduino Uno
2. *Wireless Module ESP8266*
3. Laptop/komputer
4. Kabel Serial
5. Aplikasi Android “greenhouse.apk”
6. Smartphone Android

4.5.3 Prosedur Pengujian

1. Nyalakan laptop/komputer kemudian buka aplikasi Arduino IDE untuk membuka program Arduino Uno.
2. Hubungkan Arduino Uno dengan *Wireless Module ESP8266* sesuai dengan rangkaian yang telah dibuat.
3. Hubungkan rangkaian Arduino Uno ke laptop/komputer dengan menggunakan kabel serial.
4. Pada aplikasi Arduino IDE upload program yang telah dibuat ke Arduino Uno dan tunggu *upload* sampai selesai kemudian buka *serial monitor* pada Arduino Uno untuk melihat data yang dikirimkan.
5. Setelah itu buka aplikasi Android “greenhouse.apk”

6. Masukkan IP *address* 192.168.4.1. IP tersebut adalah IP yang dimiliki oleh *Wireless Module ESP8266*.
7. Tekan tombol **OK** untuk menyimpan IP kemudian tekan tombol **Go**. Setelah itu tekan tombol **Input Tanaman Lain** dan isikan angka untuk melakukan pengujian.

4.5.4 Hasil Pengujian

Dari prosedur pengujian diatas dilakukan dengan mengamati data yang ditampilkan di Arduino IDE pada serial monitor. Dengan menampilkan data yang dikirimkan oleh aplikasi Android sebanyak 50 penekanan tombol. Dalam pengujian ini dilakukan agar dapat mengirimkan data dari aplikasi Android melalui *Wireless Module ESP8266* sesuai dengan apa yang telah diinputkan oleh *user* pada aplikasi Android. Dapat dilihat pada gambar 4.7 adalah hasil tampilan serial monitor saat menerima data dari aplikasi Android.

```

COM3 (Arduino Uno)
+IPD,2,463Link
+IPD,3,,Link
+IPD,2,463:GET /?pin=25 HTTP/1.1
Host: 192.168.4.1
Connection: keep-alive
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/*;q=0.8
User-Agent: Mozilla/5.0 (Linux; Android 4.4.2; ASUS_T00F Build/KVQ1.131030.0800)
Accept-Encoding: gzip,deflate
Accept-Language: id-ID,en-US;q=0.8
X-Requested-With: appinventor.ai_R324as.greenhouse_coba1
OK
+IPD,0,463:GET /?pin=0 HTTP/1.1
Host: 192.168.4.1
Connection: keep-alive
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/*;q=0.8
User-Agent: Mozilla/5.0 (Linux; Android 4.4.2; ASUS_T00F Build/KVQ1.131030.0800)
Accept-Encoding: gzip,deflate
Accept-Language: id-ID,en-US;q=0.8
X-Requested-With: appinventor.ai_R324as.greenhouse_coba1
OK

```

Gambar 4.7 Tampilan Serial Monitor Terima Data Transmisi *Setting*

Pada gambar 4.7 dapat dilihat saat pada aplikasi Android mengirimkan data “?pin=25” arduino dapat membaca ketika tombol **KIRIM** pada aplikasi ditekan. Kemudian tampilan aplikasi Android saat menginputkan data dapat dilihat pada gambar 4.8 berikut ini.



Gambar 4.8 Tampilan Aplikasi Android Pengujian Transmisi *Setting*

Pada gambar 4.8 diatas yaitu ketika user menginputkan data sebesar “25” untuk dikirimkan pada Arduino Uno. Pada tampilan serial monitor tercantum “?pin=” yang berfungsi untuk komunikasi saat penekanan tombol pada aplikasi Android. Pengujian transmisi data setting ini dilakukan sebanyak 50 pengiriman data juga dengan melakukan penekanan tombol sebanyak 50 secara manual. Hasil pengujian dapat dilihat pada tabel 4.3 berikut ini.

Tabel 4.3 Hasil Pengujian Transmisi *Setting*

No.	Input Data Aplikasi	Output pada Arduino	Keterangan
1.	34	?pin=34	Berhasil
2.	50	?pin=50	Berhasil
3.	10	Error	Gagal Menampilkan
4.	45	Error	Gagal Menampilkan
5.	33	Error	Gagal Menampilkan
6.	2	Error	Gagal Menampilkan
7.	41	Error	Gagal Menampilkan
8.	21	?pin=21	Berhasil
9.	1	Error	Gagal Menampilkan
10.	56	Error	Gagal Menampilkan
11.	32	Error	Gagal Menampilkan
12.	44	Error	Gagal Menampilkan
13.	60	?pin=60	Berhasil
14.	32	?pin=32	Berhasil
15.	16	?pin=16	Berhasil
16.	25	?pin=25	Berhasil
17.	17	Error	Gagal Menampilkan
18.	33	Error	Gagal Menampilkan
19.	51	Error	Gagal Menampilkan
20.	9	Error	Gagal Menampilkan
21.	26	Error	Gagal Menampilkan
22.	16	?pin=16	Berhasil
23.	12	Error	Gagal Menampilkan
24.	61	?pin=61	Berhasil
25.	35	?pin=35	Berhasil
26.	66	Error	Gagal Menampilkan
27.	37	Error	Gagal Menampilkan
28.	15	?pin=15	Berhasil
29.	41	Error	Gagal Menampilkan
30.	6	Error	Gagal Menampilkan
31.	51	?pin=51	Berhasil
32.	71	?pin=71	Berhasil
33.	64	Error	Gagal Menampilkan
34.	45	Error	Gagal Menampilkan
35.	2	Error	Gagal Menampilkan
36.	67	?pin=67	Berhasil
37.	26	Error	Gagal Menampilkan
38.	73	Error	Gagal Menampilkan
39.	36	?pin=36	Berhasil
40.	75	Error	Gagal Menampilkan
41.	25	?pin=25	Berhasil
42.	53	Error	Gagal Menampilkan
43.	7	Error	Gagal Menampilkan
44.	37	?pin=37	Berhasil

45.	38	Error	Gagal Menampilkan
46.	43	Error	Gagal Menampilkan
47.	23	Error	Gagal Menampilkan
48.	22	?pin=22	Berhasil
49.	52	Error	Gagal Menampilkan
50.	66	Error	Gagal Menampilkan

Pada tabel diatas dapat dilihat hasil pengujian dengan mengirimkan data sebanyak 50 data secara manual dengan penekanan tombol pada aplikasi Android. Dari 50 data tersebut yang dapat ditampilkan pada serial monitor Arduino IDE secara sempurna hanya sebanyak 18 data. Dapat dihitung presentase bahwa keberhasilan pengiriman data untuk menampilkan sesuai dengan inputan pada aplikasi dengan cara :

$$\text{Data Berhasil} / \text{Banyaknya Data} \times 100\%$$

Sehingga,

$$18 \text{ Data} / 50 \text{ Data} \times 100\% = 36\%$$

Jadi presentase keberhasilan transmisi data yang dapat ditampilkan sebesar 36%.

4.6 Pengujian Pengiriman Data Hasil dari Sensor DHT11 Sistem Greenhouse

4.6.1 Tujuan Pengujian

Dalam pengujian pengiriman data yang diperoleh dari sensor DHT11 ini bertujuan untuk mengetahui apakah data tersebut dapat sukses terkirim atau tidak terkirim melalui *Wireless Module ESP8266* yang telah diolah oleh program Arduino Uno dan juga untuk mengetahui berapa banyak data yang terkirim maupun yang tidak terkirim. Sehingga dapat diketahui berapa banyak data yang tidak

terkirim pada saat melakukan pengiriman data suhu dan kelembaban dari sensor DHT11 yang telah terpasang pada *greenhouse*.

4.6.2 Alat yang Dibutuhkan

1. Sensor DHT11
2. Arduino Uno
3. *Wireless Module ESP8266*
4. Laptop/komputer
5. Kabel Serial

4.6.3 Prosedur Pengujian

1. Nyalakan laptop/komputer kemudian buka aplikasi Arduino IDE untuk membuka program Arduino Uno.
2. Hubungkan Arduino Uno dengan *Wireless Module ESP8266* sesuai dengan rangkaian yang telah dibuat.
3. Hubungkan Arduino Uno dengan sensor DHT11 untuk pengambilan data suhu dan kelembaban.
4. Hubungkan rangkaian Arduino Uno ke laptop/komputer dengan menggunakan kabel serial.
5. Pada aplikasi Arduino IDE upload program ke Arduino Uno dan tunggu *upload* sampai selesai kemudian buka *serial monitor* pada Arduino Uno.

4.6.4 Hasil Pengujian

Dari prosedur diatas dilakukan pengujian dengan mengamati serial monitor pada Arduino IDE untuk mengetahui hasil perolehan data suhu dan kelembaban. Dari perolehan data akan diketahui data yang berhasil dikirim melalui *Wireless*

Module ESP8266 dengan melihat apabila setelah muncul data suhu dan kelembaban terdapat *command* AT+CIPCLOSE maka data tersebut tidak terkirim. AT+CIPCLOSE adalah memutuskan koneksi, maka data tersebut tidak akan terkirim pada saat muncul *command* tersebut. Dapat dilihat pada gambar 4.9 adalah hasil monitoring data yang diperoleh dari sensor DHT11.

```

COM3 (Arduino Uno)
suhu= 29.0C kelembaban= 66.0%
suhu= 29.0C kelembaban= 66.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 66.0%
suhu= 28.0C kelembaban= 66.0%
suhu= 28.0C kelembaban= 65.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 65.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 65.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 65.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 66.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 66.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 66.0%
suhu= 28.0C kelembaban= 66.0%
suhu= 28.0C kelembaban= 66.0%
suhu= 28.0C kelembaban= 66.0%
suhu= 28.0C kelembaban= 66.0%
suhu= 28.0C kelembaban= 66.0%
suhu= 27.0C kelembaban= 67.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 68.0%
suhu= 28.0C kelembaban= 68.0%
suhu= 28.0C kelembaban= 68.0%
AT+CIPCLOSE

```

Gambar 4.9 Tampilan Serial Monitor Saat Pengiriman Data

Dari data serial monitor tersebut dapat dilakukan pengujian data dari sensor DHT11 sebanyak 20 data suhu dan kelembaban, dapat dilihat pada tabel 4.4.

Tabel 4.4 Pengujian Pengiriman Data dari Sensor DHT11

No.	Data DHT11		Pengiriman Data
	Suhu (C)	Kelembaban (%)	
1.	29.0	66.0	Sukses
2.	29.0	66.0	Gagal
3.	28.0	66.0	Sukses
4.	28.0	66.0	Sukses
5.	28.0	65.0	Gagal
6.	28.0	65.0	Gagal
7.	28.0	65.0	Gagal

8.	28.0	65.0	Gagal
9.	28.0	66.0	Gagal
10.	28.0	66.0	Gagal
11.	28.0	66.0	Sukses
12.	28.0	66.0	Sukses
13.	28.0	66.0	Sukses
14.	28.0	66.0	Sukses
15.	28.0	66.0	Sukses
16.	28.0	66.0	Sukses
17.	27.0	67.0	Gagal
18.	28.0	68.0	Sukses
19.	28.0	68.0	Sukses
20.	28.0	68.0	Gagal

Dari 20 pengiriman data tersebut sukses mengirimkan data sebanyak 11 data suhu dan kelembaban. Data sukses terkirim tersebut tergantung dari koneksi internet.

4.7 Pengujian *Monitoring* pada Aplikasi *Android* Sistem *Greenhouse*

4.7.1 Tujuan Pengujian

Pengujian aplikasi *Android* ini dilakukan bertujuan untuk mengetahui apakah data suhu dan kelembaban yang ditampilkan oleh serial monitor dan LCD sama dengan data suhu dan kelembaban yang ditampilkan oleh aplikasi *Android*. Sehingga dapat melihat validasi data yang ditampilkan pada *hardware* dan *software* bisa sama sesuai dengan apa yang ditampilkan pada serial monitor Arduino IDE dengan aplikasi *Android*. Dalam pengujian ini diharapkan tampilan pada aplikasi *Android* bisa sama dengan tampilan *serial monitor* dan LCD.

4.7.2 Alat yang Dibutuhkan

1. Sensor DHT11
2. Arduino Uno
3. *Wireless Module ESP8266*
4. Laptop/komputer

5. Kabel Serial
6. *Smartphone Android*
7. Aplikasi *Android* “greenhouse.apk”
8. Koneksi *Internet*
9. *Liquid Crystal Display* (LCD)

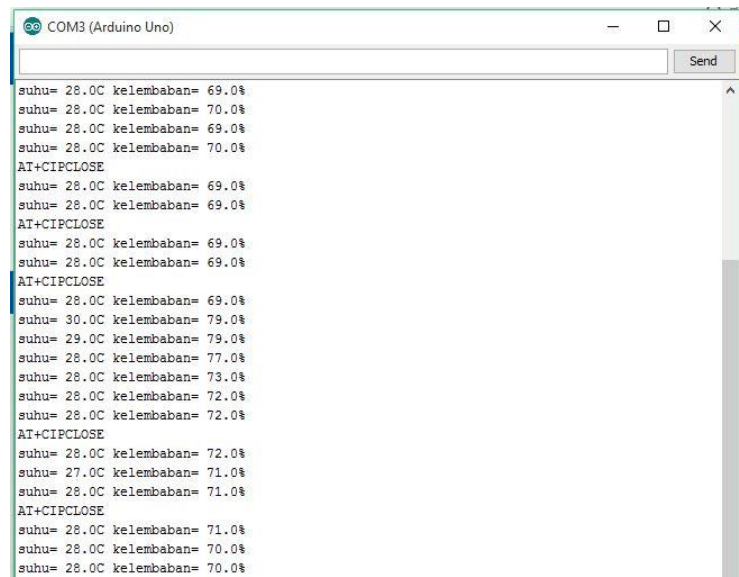
4.7.3 Prosedur Pengujian

1. Nyalakan laptop/komputer kemudian buka aplikasi Arduino IDE untuk membuka program Arduino.
2. Hubungkan Arduino Uno dengan *Wireless Module ESP8266* sesuai dengan rangkaian yang telah dibuat.
3. Hubungkan Arduino Uno dengan sensor DHT11 sesuai dengan rangkaiannya.
4. Hubungkan Arduino Uno dengan *Liquid Crystal Display* (LCD) sesuai dengan rangkaiannya.
5. Hubungkan seluruh rangkaian Arduino Uno dan *Wireless Module ESP8266* ke laptop/komputer dengan menggunakan kabel serial.
6. Hubungkan *Wireless Module ESP8266* ke koneksi *internet* dengan cara *upload* program *default* dari Arduino Uno.
7. Buka serial monitor kemudian ketikkan AT maka akan tampil OK lanjutkan dengan ketikkan AT+RST untuk mereset *Wireless Module ESP8266*. Kemudian AT+CWMODE=3 untuk mode *client* dan *host*, lanjutkan dengan AT+CWLAP untuk mengetahui daftar AP yang tersedia, kemudian AT+CWLAP=”ssid”,”password” untuk menghubungkan *Wireless Module ESP8266* ke koneksi *internet*.

8. Buka aplikasi Arduino IDE upload program ke Arduino Uno dan tunggu *upload* sampai selesai kemudian buka *serial monitor* pada Arduino Uno.
9. Install aplikasi *Android* “greenhouse.apk” pada smartphone.
10. Buka aplikasi *Android* “greenhouse.apk” kemudian masukkan *IP Address* “192.168.4.1” tekan tombol bergambar centang dan tekan tombol **Go** untuk masuk pada menu berikutnya kemudian masuk pada tombol “Info Suhu dan Kelembaban Ruangan”.
11. Lakukan pengamatan data suhu dan kelembaban pada data yang keluar pada aplikasi *Android*.

4.7.4 Hasil Pengujian

Dari prosedur diatas dilakukan pengujian dengan mengamati dan membandingkan data suhu dan kelembaban pada serial monitor, LCD, dan aplikasi *Android*. Dari perbandingan data yang diperoleh tersebut dapat dilihat apakah data pada aplikasi *Android* sama dengan data yang diperoleh oleh serial monitor dan LCD. Dapat dilihat pada gambar 4.10 yaitu tampilan serial monitor saat memperoleh data suhu dan kelembaban dari sensor DHT11 dan kemudian akan dikirimkan ke aplikasi *Android* melalui bantuan koneksi *internet* dan *website*. Dan dapat dilihat juga pada saat mengamati data antara LCD dengan aplikasi *Android* pada gambar 4.11.



```

COM3 (Arduino Uno)
Send
suhu= 28.0C kelembaban= 69.0%
suhu= 28.0C kelembaban= 70.0%
suhu= 28.0C kelembaban= 69.0%
suhu= 28.0C kelembaban= 70.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 69.0%
suhu= 28.0C kelembaban= 69.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 69.0%
suhu= 28.0C kelembaban= 69.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 69.0%
suhu= 30.0C kelembaban= 79.0%
suhu= 29.0C kelembaban= 79.0%
suhu= 28.0C kelembaban= 77.0%
suhu= 28.0C kelembaban= 73.0%
suhu= 28.0C kelembaban= 72.0%
suhu= 28.0C kelembaban= 72.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 72.0%
suhu= 27.0C kelembaban= 71.0%
suhu= 28.0C kelembaban= 71.0%
AT+CIPCLOSE
suhu= 28.0C kelembaban= 71.0%
suhu= 28.0C kelembaban= 70.0%
suhu= 28.0C kelembaban= 70.0%

```

Gambar 4.10 Tampilan Serial Monitor Pengujian Pengiriman Data ke Aplikasi

Android



Gambar 4.11 Perbandingan Data Suhu dan Kelembaban LCD dan Aplikasi

Android

Dari data serial monitor pada gambar 4.11 dapat dilakukan pengujian data dari sensor DHT11 sebanyak 20 data suhu dan kelembaban untuk membandingkan data tersebut dengan LCD dan aplikasi *Android* dapat dilihat pada tabel 4.5.

Tabel 4.5 Pengujian *Monitoring* pada Aplikasi *Android* Sistem *Greenhouse*

No.	Pengiriman Data	Serial	LCD	Aplikasi
1.	Sukses	28 C / 69 %	28 C / 69 %	28 C / 69 %
2.	Sukses	28 C / 70 %	28 C / 70 %	28 C / 70 %
3.	Sukses	28 C / 69 %	28 C / 69 %	28 C / 69 %
4.	Gagal	28 C / 70 %	28 C / 70 %	28 C / 69 %
5.	Sukses	28 C / 69 %	28 C / 69 %	28 C / 69 %
6.	Gagal	28 C / 69 %	28 C / 69 %	28 C / 69 %
7.	Sukses	28 C / 69 %	28 C / 69 %	28 C / 69 %
8.	Gagal	28 C / 69 %	28 C / 69 %	28 C / 69 %
9.	Sukses	30 C / 79 %	30 C / 79 %	30 C / 79 %
10.	Sukses	29 C / 79 %	29 C / 79 %	29 C / 79 %
11.	Sukses	28 C / 77 %	28 C / 77 %	28 C / 77 %
12.	Sukses	28 C / 73 %	28 C / 73 %	28 C / 73 %
13.	Sukses	28 C / 72 %	28 C / 72 %	28 C / 72 %
14.	Gagal	28 C / 72 %	28 C / 72 %	28 C / 69 %
15.	Sukses	28 C / 72 %	28 C / 72 %	28 C / 72 %
16.	Sukses	27 C / 71 %	27 C / 71 %	27 C / 71 %
17.	Gagal	28 C / 71 %	28 C / 71 %	27 C / 71 %
18.	Sukses	28 C / 71 %	28 C / 71 %	28 C / 71 %
19.	Sukses	28 C / 70 %	28 C / 70 %	28 C / 70 %
20.	Sukses	28 C / 70 %	28 C / 70 %	28 C / 70 %

Dari tabel 4.5 diatas dapat dilihat hasil pengujian membandingkan hasil data yang ditampilkan. Ketika pengiriman data sukses maka tampilan pada aplikasi akan mengikuti hasil yang ditampilkan oleh tampilan serial monitor dan LCD. Sedangkan pada saat pengiriman data gagal menuju ke aplikasi, maka tampilan pada aplikasi akan tetap dan tidak berubah mengikuti hasil sebelumnya.