**BAB IV**

**IMPLEMENTAS DAN PEMBAHASAN SISTEM**

1. **Implementasi**

Tahap implementasi meliputi penjelasan mengenai implementasi perangkat lunak (*software*) dan implementasi perangkat keras (*hardware*) serta implementasi antarmuka (*interface*).

1. **Ruang Lingkup Implementasi**

Ruang lingkup implementasi dimaksudkan agar implementasi menjadi lebih jelas. Ruang lingkup implementasi Sistem Keamanan Rumah Menggunakan Arduino Mega 2560 Dengan Metode *Fuzzy Logic* Berbasis Android ini adalah sebagai berikut:

1. Pengujian Sistem Keamanan Rumah Menggunakan Arduino Mega 2560 Dengan Metode *Fuzzy Logic* Berbasis Android menggunakan perangkat *smartphone android*.
2. Perangkat implementasi menggunakan perangkat *smartphone android* dan rangkaian perangkat keras.
3. Implementasi sistem keamanan rumah lebih fokus terhadap monitoring terhadap keadaan rumah.
4. **Implementasi Perangkat Keras**

Implementasi sistem keamanan rumah menggunakan Arduino Mega 2560 dengan metode *fuzzy logic* ini, perangkat keras yang digunakan adalah sebagai berikut:

**Tabel 4.1 Spesifikasi Perangkat Keras (*Hardware*)**

|  |  |
| --- | --- |
| Perangkat Kontroler | Smartphone Android |
| Arduino Mega 2560 | CPU : 1GHz |
| Arduino Ethernet Sheld | Ram : 512 MB |
| Adaptor 500A | Free Space : 20 MB |
| Sensor PIR |  |
| Sensor *Ultrasonic* |  |
| *Magnetic Switch* |  |
| *Buzzer* |  |

1. **Implementasi Perangkat Lunak**

Untuk implementasi sistem keamanan rumah menggunakan Arduino Mega 2560 dengan metode fuzzy logic ini, perangkat lunak yang digunakan untuk mendukung berjalannya sistem diantaranya sebagai berikut:

**Tabel 4.2 Spesikasi Perangkat Lunak (*Software*)**

|  |  |
| --- | --- |
| Sistem Operasi Android | Jelly KitKat 4.4 |
| Database | MySql |
| Sistem Operasi Windows | Windows 7 |
| Web Bworser | Google Chrome |

1. **Implementasi Antarmuka**

Pada tahapan ini pengujian sistem yang dibahas berdasarkan objek yang akan diuji berupa *form*-*form* sebagai berikut:

1. *Form* Login

**Gambar 4.1 *Form* Login**

Pada *form* ini pengguna dihadapkan dengan tampilan login yang dimana tampilan akan muncul ketika pengguna membuka aplikasi untuk pertama kali dan kemudian pengguna diminta untuk memasukan username dan password.

1. *Form* Daftar

**Gambar 4.2 *Form* Daftar**

Pada *form* ini pengguna dihadapkan dengan tampilan daftar yang dimana tampilan akan muncul ketika pengguna menekan tombol daftar pada halaman login kemudian pengguna diminta untuk memasukan data sesuai isian yang ditampilkan.

1. *Form* Menu Utama

**Gambar 4.3 Tampilan *Form* Menu Utama**

Pengguna akan dihadapkan pada tampilan awal yaitu *form* Menu Utama. Terdiri dari :

1. Monitoring : untuk memantau kondisi terkini rumah .
2. Kelola Profil : berfungsi untuk mengubah profil user yang sedang login
3. Kelola Password : berfungsi untuk mengubah password user yang sedang login
4. Kelola Pengguna : berfungsi untuk mengelola data pengguna (fitur ini hanya bisa diakses olehroot / koordinator keluarga)
5. *Form* Monitoring

**Gambar 4.4 Tampilan *Form* Monitoring**

Pada *form* ini menampilkan status terkini dari sensor yang terdapat pada rumah pengguna

1. *Form* Kelola Profil

**Gambar 4.5 *Form* Kelola Profil**

Pada *form* ini pengguna dihadapkan dengan tampilan ubah profil yang dimana terdapat beberapa field berisi data user yang dapat diedit.

1. *Form* Kelola Password

**Gambar 4.6 *Form* Kelola Password**

Pada *form* ini pengguna dihadapkan dengan tampilan ubah password yang dimana terdapat tiga field yang harus diisi ketika pengguna ingin merubah password miliknya.

1. *Form* Kelola Pengguna

**Gambar 4.7 *Form* Kelola Pengguna**

Pada *form* ini pengguna dihadapkan dengan submenu yaitu tambah pengguna dan daftar pengguna, jika pengguna mengklik daftar pengguna maka pengguna (root / koor kel) dapat mengedit atau menghapus pengguna tersebut.

1. *Form* Kelola Sistem

**Gambar 4.8 *Form* Kelola Sistem**

Pada *form* ini pengguna dihadapkan dengan toggle untuk mematikan atau menyalakan sistem.

1. *Form* Log

**Gambar 4.9 Tampilan *Form* Monitoring**

Pada *form* ini menampilkan data dari sensor yang statenya berubah menjadi waspada atau awas.

1. *Form* Petunjuk

**Gambar 4.10 Tampilan Petunjuk**

Pada Halaman Antarmuka ini pengguna akan diperlihatkan aturan penggunaan dan cara kerja aplikasi ini.

1. *Form* Tentang

**Gambar 4.11 Tampilan Tentang**

Pada halaman antarmuka pengguna akan diperlihatkan info pengembang diantaranya ada Dosen Pembimbing serta biodata pembuat aplikasi.

1. **Pengujian Sistem**

Pengujian sistem adalah proses mengeksekusi perangkat lunak untuk menentukan apakah sistem tersebut memenuhi spesifikasi sistem dan berjalan sesuai dengan yang diharapkan.

* + 1. **Black Box Testing**

Pada tahap ini pengujian sistem yang dibahas yaitu objek-objek pada antarmuka (*interface*) yang berupa *Button, TextView* dan koneksi antar perangkat.

**Tabel 4.3 Pengujian Black Box Testing**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Fungsi yang diuji | Cara Menguji | Hasil yang diharapkan | Hasil pengujian |
| 1. | Login | Pengguna memasukan username dan pasword kemudian menekan tombol login | Sistem akan mengarahkan halaman pada tampilan menu utama sesuai dengan tingkatan user | Sistem menampilkan halaman utama dengan alert masuk sesuai dengan tikatan user. |
| 2. | Daftar | Pengguna memasukan username, pasword, kemudian menekan tombol daftar | Jika pendaftaran berhasil sistem akan mengarahkan pengguna pada halaman login, dan jika gagal maka sistem akan mengeluarkan alert | Pendaftaran berhasil data berhasil disimpan dan redirect ke halaman login, data gagal disimpan sistem menampilkan alert |
| 3. | Monitoring | Pengguna mengirim perintah dengan menekan menu monitoring. | Sistem akan mengirim *request* data pada server untuk memunculkan data dari sensor. | Sistem menampilkan status sensor terkini. |
| 4. | Kelola Profil | Pengguna mengirim perintah dengan menekan menu kelola profil, kemudian mengedit profil dirinya sendiri. | Sistem menampilkan data pengguna yang sedang sedang login dan dapat dirubah oleh pengguna dengan menekan tombol edit. | Sistem menampilkan profil pengguna dan ketika diedit data berhasil disimpan. |
| 5. | Kelola Password | Pengguna mengirim perintah dengan menekan menu kelola password, kemudian mengedit password. | Sistem menampilkan form ubah password, jika data sesuai maka pass baru disimpan jika tidak maka sistem mengeluarkan alert. | Sistem menampilkan form ubah pass dan ketika data sesuai password berhasil disimpan ketika data tidak sesuai sistem mengeluarkan alert. |
| 6. | Kelola Pengguna | Pengguna mengirim perintah dengan menekan menu kelola pengguna, kemudian melakukan tambah, edit, atau hapus pengguna. | 1. Sistem menampilkan form tambah pengguna jika pengguna memilih submenu tersebut, jika data pengguna baru sesuai maka data akan disimpan, jika tidak maka sistem menampilkan alert  2. Jika pengguna (koor kel / root) mengedit data pengguna maka sistem akan menampilkan data pengguna yang akan diedit, jika data sesuai maka data akan disimpan, jika tidak maka sistem menampilkan alert.  3. Jika pengguna (koor kel / root) menghapus data pengguna maka sistem menampilkan alert apakah yakin akan dihapus atau tidak. | 1. Sistem menampilkan form tambah pengguna dan ketika data pengguna baru sesuai berhasil disimpan ketika data tidak sesuai sistem mengeluarkan alert.  2. Sistem menampilkan form edit pengguna dan ketika data sesuai berhasil disimpan ketika data tidak sesuai sistem mengeluarkan alert.  3. Sistem menampilkan alert ketika pengguna (koor kel / root) akan menghapus data pengguna |
| 7. | Kelola Sistem | Pengguna mengirim perintah dengan menekan tombol matikan / nyalakan sistem | Sistem mengirim perintah ke arduino untuk mematikan / menyalakan sistem, status sistem berubah menjadi nyala / mati | Sistem menampilkan status sistem menjadi nyala / mati |
| 8. | Log | Pengguna mengirim perintah dengan menekan menu Log. | Sistem akan mengirim *request* data pada server untuk memunculkan data dari sensor yang mendeteksi. | Sistem menampilkan data sensor yang mendeteksi pada waktu tertentu. |
| 9. | Notifikasi | Melakukan monitoring sampai state dari setiap sensor berubah menjadi Waspada atau Awas. | Sensor melakukan pengukran 3-5 kali untuk menampilkan notifikai waspada. Sensor melakukan pengukran > 5 kali untuk menampilkan notifikai awas. | Notifikasi muncul ketika terjadi perubahan state pada sensor. |

1. **White Box Testing**

Pengujian White Box berfokus pada structural kontrol program. Adapun metode yang digunakan dalam pengujian white box ini adalah metode basis Path untuk memastikan bahwa semua statement pada program telah dieksekusi paling tidak satu kali selama pengujian dan bahwa kondisi logis yang telah diuji. Struktur program yang akan diuji:

**Tabel 4.4 Node Flow Graph Nation**

|  |  |
| --- | --- |
| *Line*/*Node* | *Coding* |
|  | #include<LiquidCrystal.h>  #include<dht.h>  #include <Ethernet.h>  #include <SPI.h>  #define STATUS\_DISCONNECTED 0  #define STATUS\_CONNECTED 1  #define DHT\_SENSOR\_PIN 5  #define KIPAS\_PIN 3  #define LAMPU\_PIN 4    byte MAC\_eth[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED }; |
|  | void setup(){  state\_lampu = "NULL";  state\_kipas = "NULL";  Serial.begin(9600);  pinMode(KIPAS\_PIN,OUTPUT);  pinMode(LAMPU\_PIN,OUTPUT);  Serial.println("--------------------------------------------------");  Serial.println("Setting Perangkat");  Serial.println("Mohon menunggu . . . ");  Serial.println("Setting Ethernet MAC Address dan IP Address");  Serial.println("Mohon menunggu . . . ");  if (Ethernet.begin(MAC\_eth) == 0) {  Serial.println("Failed to configure Ethernet using DHCP");  Ethernet.begin(MAC\_eth,IP\_eth);  }  // Ethernet.begin(MAC\_eth,IP\_eth);  Serial.println("Setting Sensor Suhu dan Kelembaban");  Serial.println("Mohon menunggu . . . ");  Serial.print("Versi Library DHT : ");  Serial.println(DHT\_LIB\_VERSION);  delay(1000);  Serial.println("Setting Perangkat selesai!");  Serial.println("--------------------------------------------------");  }  void loop() {  Serial.print("Iterasi ke : ");  Serial.println(iterasi);  inisialisasi\_dht22();  String a = data\_dht22();  int resultBukaKoneksi = bukaKoneksi();  if(resultBukaKoneksi==1){  kirimData(a);  Serial.println();  }  delay(1000);  Serial.println("--------------------------------------------------");  iterasi++;  }  int bukaKoneksi(){  Serial.print("Mencoba sambungan ke server http://");  Serial.println(namaServer);  Serial.println("Mohon menunggu . . . ");  if(myEthernet.connect(namaServer,80)){  Serial.println("Sambungan ke server berhasil!");  return STATUS\_CONNECTED;  }else{  Serial.print("Sambungan ke server gagal!");  Serial.println();  return STATUS\_DISCONNECTED;  }  } |
|  | void kirimData(String a){  Serial.println("Menjalankan perintah kirim data");  String data = a;  int ln = data.length();  String uri\_segment;  uri\_segment = "/automation\_system\_aal/index.php/api/post\_data/" + data;  // uri\_segment = "/iot\_server/index.php/device/post\_data/" + a;  myEthernet.print("GET ");  myEthernet.print(uri\_segment);  Serial.print("Data yang dikirim di ke server : ");  Serial.println(data);  myEthernet.println(" HTTP/1.1 ");  myEthernet.print( "Host : " );    Serial.println("Host OK");  myEthernet.println( "Content-Type: application/x-www-form-urlencoded \r\n" );  Serial.println("Content type OK");  myEthernet.print( "Content-Length : " );  myEthernet.print(ln);  myEthernet.print(" \r\n");  myEthernet.println( "Connection: close" );  myEthernet.println();  String res = bacaWebText();  if(res.equals("")==false){  String response\_insert = getValue(res,'\*',0);  String response\_lampu = getValue(res,'\*',1);  String response\_kipas = getValue(res,'\*',2);  String response\_mode = getValue(res,'\*',3);  Serial.print("Response Insert : ");Serial.println(response\_insert);  Serial.print("Response Lampu : ");Serial.println(response\_lampu);  Serial.print("Response Kipas : ");Serial.println(response\_kipas);  Serial.print("Response Mode : ");Serial.println(response\_mode);  if(iterasi==1){  if(response\_lampu.equals("1")){  nyalakanLampu();  }  if(response\_kipas.equals("1")){  nyalakanKipas();  }  if(response\_lampu.equals("0")){  matikanLampu();  }  if(response\_kipas.equals("0")){  matikanKipas();  }  } |
|  | String bacaWebText(){  unsigned int time;  Serial.println("Baca respon dari server . . . ");  Serial.println("Mohon menunggu . . . ");  time = millis();  Serial.print("Timer Millis () : ");  Serial.println(time);  int stringPos = 0;  memset( &inString, 0, 32 );  int unvailable\_ctr = 0;  while(true){  if (myEthernet.available()) {  char c = myEthernet.read();  Serial.print(c);  if (c == '#' ) {  Serial.print("Menemukan start key # dengan isi : ");  startRead = true;  }else if(startRead){  if(c != '^'){  inString[stringPos] = c;  stringPos ++;  }else{  startRead = false;  Serial.println();  Serial.println("Baca respon dari server selesai!");  myEthernet.stop();  myEthernet.flush();  Serial.println("Sambungan diputuskan . . . ");  return inString;  }  }  }else{  //Serial.println("ethernet unavailable");  delay(50);  unvailable\_ctr++;  if(unvailable\_ctr == 25){  myEthernet.stop();  myEthernet.flush();  Serial.println("Koneksi mengalami time out");  Serial.println("Sambungan diputuskan . . . ");  Serial.println("Reset...");  return inString;  }  }  }  } |
|  | void inisialisasi\_dht22(){  respon\_dht22 = DHT.read22(DHT\_SENSOR\_PIN);  switch (respon\_dht22){  case DHTLIB\_OK:  Serial.println("Sensor DHT status : OK");  break;  case DHTLIB\_ERROR\_CHECKSUM:  Serial.println("Sensor DHT status : Checksum error");  break;  case DHTLIB\_ERROR\_TIMEOUT:  Serial.println("Sensor DHT status : Time out error");  break;  case DHTLIB\_ERROR\_CONNECT:  Serial.print("Sensor DHT status : Connect error");  break;  case DHTLIB\_ERROR\_ACK\_L:  Serial.print("Sensor DHT status : Ack Low error");  break;  case DHTLIB\_ERROR\_ACK\_H:  Serial.print("Sensor DHT status : Ack High error");  break;  default:  Serial.println("Sensor DHT status : Unknown error");  break;  }  delay(2000); |
|  | String data\_dht22(){  String data;  Serial.print("Temperatur : ");  Serial.print(DHT.temperature);  Serial.print(",\t");  Serial.print("Kelembaban : ");  Serial.println(DHT.humidity);  delay(2000); |
|  | String temperature = String(DHT.temperature);  String humidity = String(DHT.humidity);      counter\_dingin += 1;  counter\_normal = 0;  counter\_panas = 0;  if(counter\_dingin >= 12){  nyalakanLampu();  state\_lampu = "SET\_ON";  matikanKipas();  state\_kipas = "SET\_OFF";  }  }else    matikan lampu, matikan kipas  counter\_dingin = 0;  counter\_normal += 1;  counter\_panas = 0;  if(counter\_normal >= 12){  matikanLampu();  state\_lampu = "SET\_OFF";  matikanKipas();  state\_kipas = "SET\_OFF";  }  }else    counter\_dingin = 0;  counter\_normal = 0;  counter\_panas += 1;  if(counter\_panas >= 12){  matikanLampu();  state\_lampu = "SET\_OFF";  nyalakanKipas();  state\_kipas = "SET\_ON";  }  } (proses fuzzy) |
|  | Serial.print("Counter Dingin : ");Serial.println(counter\_dingin);  Serial.print("Counter Normal : ");Serial.println(counter\_normal);  Serial.print("Counter Panas : ");Serial.println(counter\_panas);  if(kondisi\_lampu\_terakhir.equals(state\_lampu) && kondisi\_kipas\_terakhir.equals(state\_kipas)){  Serial.println("Kirim data mode : DEFAULT");  data = temperature + "/" + humidity + "/POST\_MODE\_DEFAULT/NULL/NULL/";  }else{    }  kondisi\_lampu\_terakhir = state\_lampu;  kondisi\_kipas\_terakhir = state\_kipas;  recent\_suhu = DHT.temperature ;  recent\_kelembaban = DHT.humidity ;  return data; |
|  | void nyalakanLampu(){ (jalan perintah)  digitalWrite(LAMPU\_PIN,LOW);  Serial.println("Menyalakan Lampu");  }    digitalWrite(LAMPU\_PIN,HIGH);  Serial.println("Mematikan Lampu");  }  void nyalakanKipas(){  digitalWrite(KIPAS\_PIN,LOW);  Serial.println("Menyalakan Kipas");  }  void matikanKipas(){  digitalWrite(KIPAS\_PIN,HIGH);  Serial.println("Mematikan Kipas");  } |
|  | String getValue(String data, char separator, int index){  int found = 0;  int strIndex[] = {0, -1};  int maxIndex = data.length( ;  for(int i=0; i<=maxIndex && found<=index; i++){  if(data.charAt(i)==separator|| i==maxIndex){  found++;  strIndex[0] = strIndex[1]+1;  strIndex[1] = (i == maxIndex) ? i+1 : i;  }  } found>index data.substring [0], strIndex[1]) : "";  } |

Keterangan:

1. Node 1 merupakan Deklarasi Librasi
2. Node 2 Inisialissi set up lampu kipas dan ethernet.
3. Node 3 Set Up Iterasi dan menjalankan Respon.
4. Node 4 Untuk membaca Respon
5. Node 5 mengecek sensor DHT22
6. Node 6 merupakan kondisi membaca suhu dan kelembaban.
7. Node 7 Proses Fuzzy.
8. Node 8 merupakan kondisi membuat parameter untuk dikirim ke server
9. Node 9 merupakan proses *fuzzy rules* (*inference*) dan *defuzzification* dalam keadaan lingkungan dengan nilai suhu pada status “lampu Nyala”, “Lampu Mati”, “Lampu dan Kipas Mati”,“Kipas Nyala”, “Kipas Mati”dan nilai pada status sesuai kondisi.
10. Node 10 Membaca String Data.

Berikut ini adalah gambar *Flow Graph Notation* dalam melakukan *white box testing*:



**Gambar 4.7** *Flow Graph Notation*

1. Keterangan Gambar:

Node (N) : 10

Edge (E) : 10

1. Perhitungan
2. V(G) = E – N + 2

Dimana: E = 10

V(G) = 10 – 10 + 2

= 2

Langkah alurnya sebagai berikut :

Path 1 : 1,2,3,4,5,6,7,8,10

Path 2 : 1,2,3,4,5,6,7,9,10

Jadi setelah didapatkan path nya, berikut ini adalah cara pengujian nya:

Berikut ini pengujian dengan *graph matriks*. *Graph Matriks* merupakan matrik empat persegi yang mempunyai ukuran yang sama dengan jumlah node pada *flow graph*.

**Tabel 4.5 *Graph Matriks***

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| N | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |  |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Jumlah** | | | | | | | | | | | 1 |

V(G) = Jumlah *Graph Matriks* + 1

V(G) = 1 + 1

V(G) = 2

Sehingga, nilai pafa flowgraph metriks berjumlah 2.