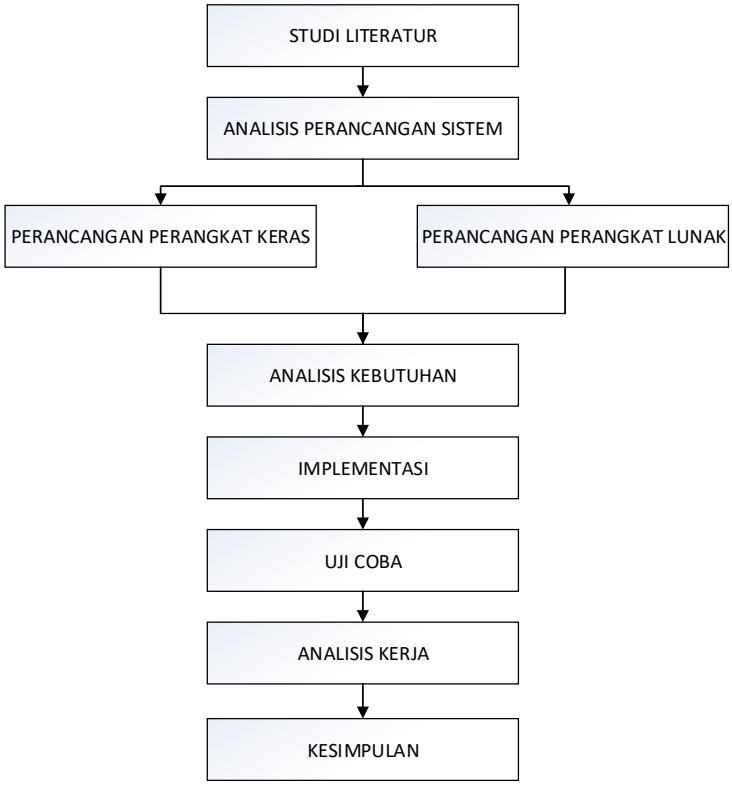
BAB III
METODE PENELITIAN

Alur penelitian yang digunakan mengacu pada model *Desain for Manufacturing and Assembly* (DFMA) (Boothroyd, Dewhurst, & Knight, 2010).



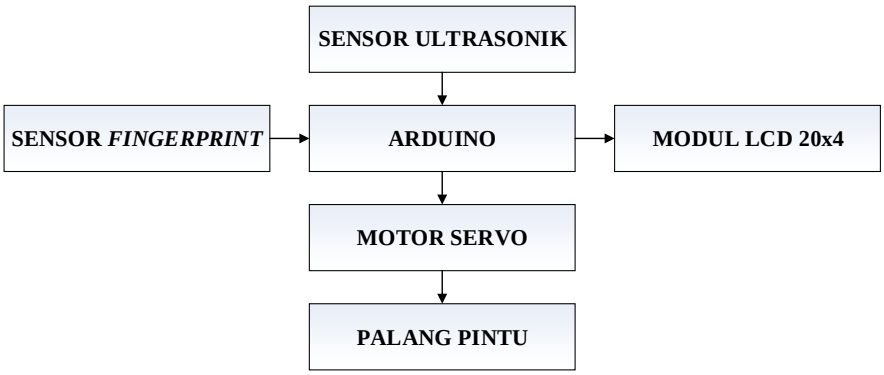
Gambar 3.1 Alur Penelitian

1.1 Studi Literatur

Pada bagian metode ini penulis mencari bahan rujukan penulisan tugas akhir dari buku, jurnal dan website yang terkait dengan pembuatan sistem Kendali Palang Parkir Motor Khusus Dosen dan Karyawan IIB Darmajaya.

1.2 Analisis Perancangan Sistem

Perancangan Kendali Palang Parkir Motor Khusus Dosen dan Karyawan IIB Darmajaya Menggunakan *Fingerprint* Berbasis Arduino Uno ini meliputi perancangan perangkat keras (*hardware*) dan perangkat lunak (*software*). Sistem yang dirancang akan membentuk suatu sistem yang dapat memilah antara mahasiswa dengan dosen dan karyawan sehingga tidak ada mahasiswa yang parkir di area parkir dosen dan karyawan serta dapat memberikan keamanan pada area parkir dosen dan karyawan IIB Darmajaya. Adapun blok diagram dari Kendali Palang Parkir Motor Khusus Dosen dan Karyawan IIB Darmajaya Menggunakan *Fingerprint* Berbasis Arduino Uno dapat dilihat pada gambar 3.2.



Gambar 3.2 Blok Diagram

Komponen – komponen pada blok diagram :

1. Modul Sensor *Fingerprint*

Berfungsi sebagai input untuk membaca rekaman sidik jari pengguna parkir motor yang ada pada *database.*

2. Sensor Ultrasonik HC-SR04

Berfungsi sebagai pendeteksi motor yang telah melewati palang parkir dan memberikan perintah ke Arduino untuk menutup kembali palang parkir.

3. Arduino Uno

Berfungsi sebagai mikrokontroler untuk memproses input dari sensor kemudian menghasilkan output ke aktuator seperti yang diinginkan.

4. Modul LCD (*Liquid Crystal Display*) 20x4

Berfungsi untuk menampilkan informasi mengenai kecocokaan data sidik jari yang telah diterima sensor *fingerprint*.

5. Motor Servo

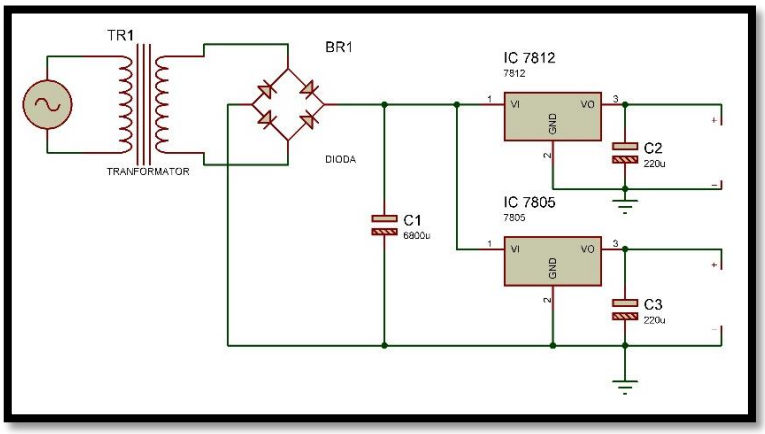
Berfungsi sebagai aktuator untuk menggerakkan palang parkir.

1.2.1 Perancangan Perangkat Keras

Perancangan perangkat keras Kendali Palang Parkir Motor Khusus Dosen dan Karyawan IIB Darmajaya Menggunakan *Fingerprint* Berbasis Arduino Uno ini menjadi bagian yang sangat penting dilakukan dalam pembuatan suatu alat karena dengan merancang terlebih dahulu dengan komponen yang tepat akan mengurangi pembelian komponen yang berlebih dan kerja alat dapat sesuai dengan yang diinginkan. Untuk menghindari kerusakan komponen perlu dipahami juga akan karakteristik dari masing - masing komponen tersebut.

1.2.1.1 Perancangan Catu Daya

Rangkaian catu daya digunakan untuk merubah tegangan AC (*Alternating Current*) 220 Volt menjadi DC (*Direct Current*) 5 Volt dan DC (*Direct Current*) 12 Volt agar rangkaian tersebut dapat bekerja sebagaimana mestinya. keluaran 12 Volt dihubungkan ke Arduino Uno dan keluaran 5 Volt dihubungkan ke Motor Servo. Perancangan catu daya dapat dilihat pada gambar 3.3

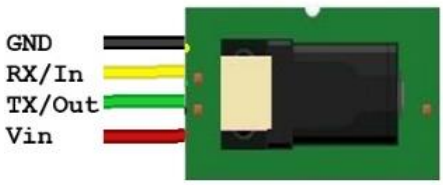


Gambar 3.3 Perancangan Catu Daya / Sumber Tegangan DC (*Direct Current*)

1.2.1.2 Perancangan Modul Sensor *Fingerprint*

Perancangan modul sensor *fingerprint* dilakukan dengan cara menghubungkan kabel dari modul sensor *fingerprint* ke modul Arduino sesuai dengan standar pengkabelan yang ada pada *datasheet* modul sensor *fingerprint*. Standar pengkabelan pada modul sensor *fingerprint* adalah sebagai berikut :

1. Vin *Fingerprint* dihubungkan ke 5 Volt Arduino.
2. GND *Fingerprint* dihubungkan ke GND Arduino.
3. Pin TX *(Data Out) Fingerprint* (Kabel Hijau) dihubungkan ke pin 2 Arduino.
4. Pin RX *(Data In) Fingerprint* (Kabel Putih) dihubungkan ke pin 3 Arduino.



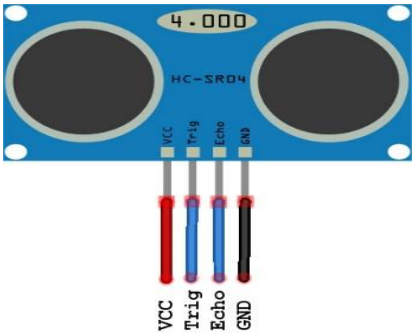
Gambar 3.4 Pengkabelan Modul Sensor *Fingerprint*

1.2.1.3 Perancangan Sensor Ultrasonik HC-SR04

Perancangan sensor ultrasonik HC-SR04 dilakukan dengan cara menghubungkan kabel dari sensor ultrasonik HC-SR04 ke modul Arduino sesuai dengan standar

pengkabelan yang ada pada *datasheet* sensor ultrasonik HC-SR04. Standar pengkabelan pada modul sensor *fingerprint* adalah sebagai berikut :

1. VCC sensor ultrasonik dihubungkan ke 5 Volt Arduino.
2. Ground sensor ultrasonik dihubungkan ke GND Arduino.
3. Pin *Trigger* dihubungkan ke pin 12 Arduino.
4. Pin *Echo* dihubungkan ke pin 13 Arduino.

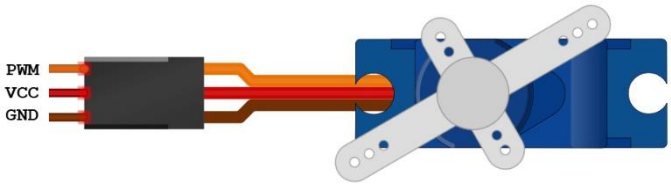


Gambar 3.5 Pengkabelan Sensor Ultrasonik

1.2.1.4 Perancangan Motor Servo

Perancangan motor servo dilakukan dengan cara menghubungkan kabel dari motor servo ke modul Arduino sesuai dengan standar pengkabelan yang ada pada *datasheet* motor servo. Standar pengkabelan pada motor servo adalah sebagai berikut :

1. VCC motor servo dihubungkan ke 5 Volt Arduino.
2. Ground motor servo dihubungkan ke GND Arduino.
3. Pin PWM motor servo dihubungkan ke pin 9 Arduino.



Gambar 3.6 Pengkabelan Motor Servo

1.2.1.5 Perancangan Modul LCD *(Liquid Crystal Display)* 20x4

Perancangan modul LCD *(Liquid Crystal Display)* 20x4 dilakukan dengan cara menghubungkan kabel dari modul LCD 20x4 ke modul Arduino sesuai dengan standar pengkabelan yang ada pada *datasheet* modul LCD *(Liquid Crystal Display)* 20x4. Standar pengkabelan pada Modul LCD *(Liquid Crystal Display)* 20x4 adalah sebagai berikut :

1. VCC LCD 20x4 dihubungkan ke 5 Volt Arduino.
2. Ground LCD 20x4 dihubungkan ke GND Arduino.
3. Pin SDA LCD 20x4 dihubungkan ke Pin A4 Arduino.
4. Pin SCL LCD 20x4 dihubungkan ke Pin A5 Arduino.

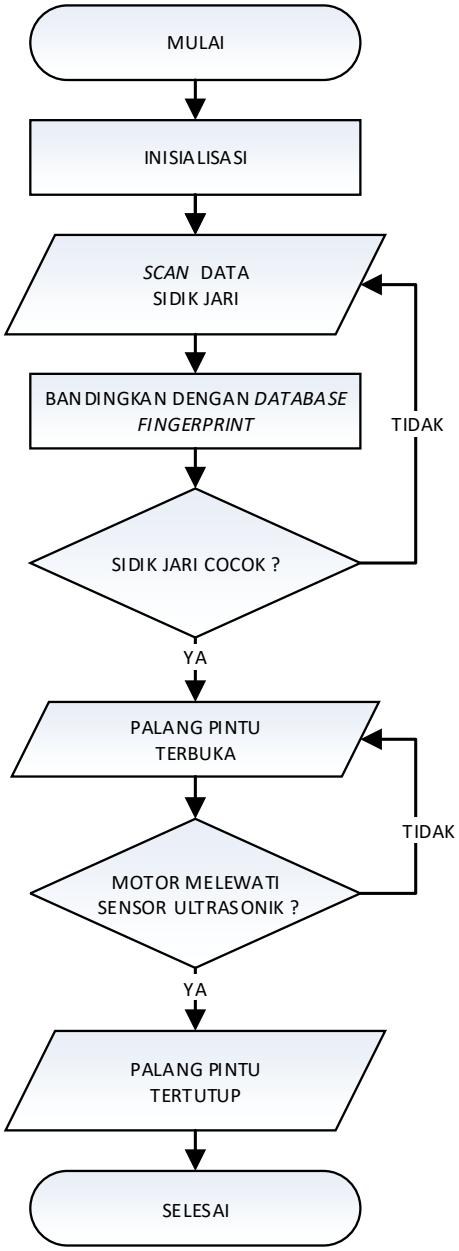
Gambar 3.7 Pengkabelan Modul LCD (*Liquid Crystal Display*) 20x4

1.2.2 Perancangan Perangkat Lunak

Perancangan perangkat lunak dimulai dari pembuatan flowchart untuk program pada arduino uno. Kemudian mendesain tampilan pada LCD (*Liquid Crystal Display*) agar dapat dapat dilihat pengguna parkir.

1.2.2.1 Perancangan *Flowchart* Sistem

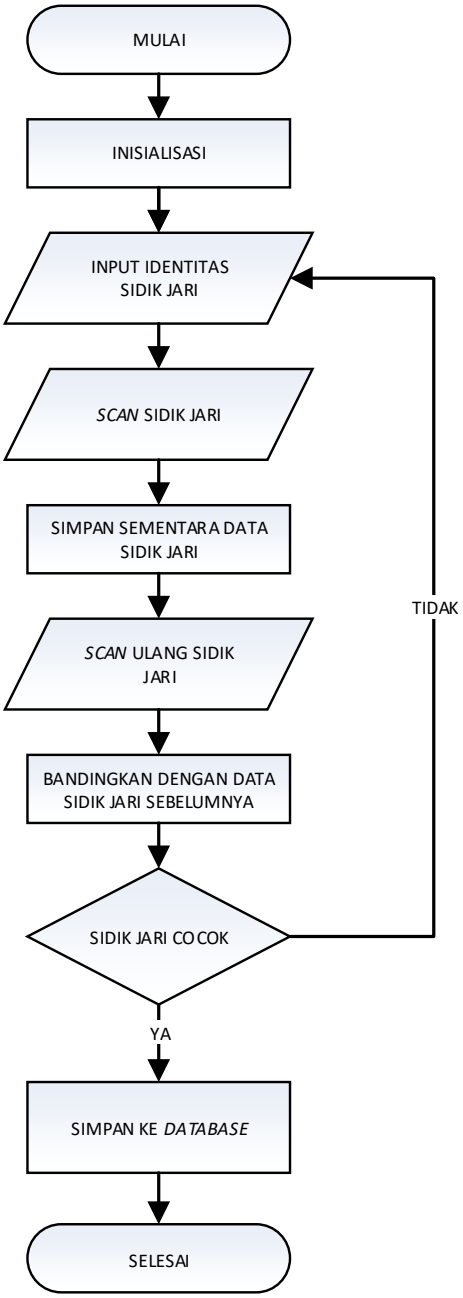
Sistem flowchart dibuat bertujuan untuk mengetahui jalannya suatu sistem yang dibuat pada *hardware* dan *software* sehingga mempermudah dalam pembuatan program. *Flowchart* sistem dapat dilihat pada gambar 3.8.



Gambar 3.8 *Flowchart* sistem

1.2.2.2 Perancangan *Flowchart enroll* pada *database* modul *fingerprint*

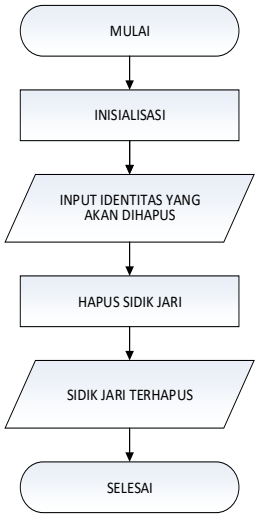
Flowchart enroll pada database modul sensor *fingerprint* dibuat bertujuan untuk mengetahui jalannya proses *enroll* pada *database* modul sensor *fingerprint*. *Flowchart Flowchart enroll* pada database modul *fingerprint* dapat dilihat pada gambar 3.9.



Gambar 3.9 *Flowchart enroll* modul *fingerprint*

1.2.2.3 Perancangan Flowchart *delete* pada *database* modul *fingerprint*

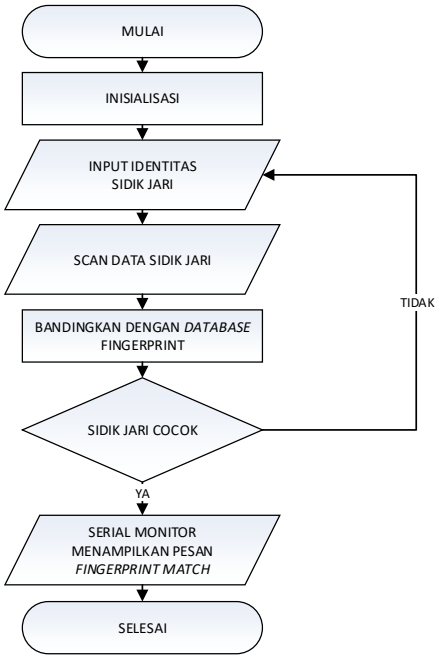
Flowchart delete pada *database* modul sensor *fingerprint* dibuat bertujuan untuk mengetahui jalannya proses *delete* pada *database* modul sensor *fingerprint*. *Flowchart delete* pada *database* modul *fingerprint* dapat dilihat pada gambar 3.10.



Gambar 3.10 *Flowchart delete database* pada modul *fingerprint*

1.2.2.4 Perancangan *Flowchart matching fingerprint* pada *database*

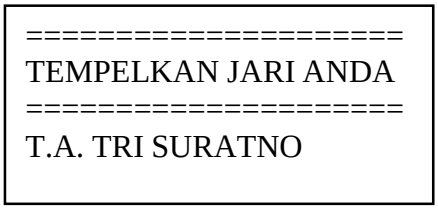
Flowchart matching fingerprint pada *database* bertujuan untuk mengetahui jalannya proses *matching fingerprint* pada *database* modul sensor *fingerprint*. *Flowchart matching fingerprint* pada *database* dapat dilihat pada gambar 3.11.



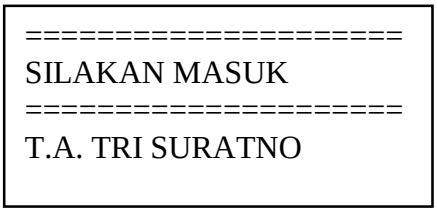
Gambar 3.11 *Flowchart matching fingerprint*

1.2.2.5 Perancangan Tampilan pada LCD (*Liquid Crystal Display*)

Perancangan desain dari tampilan LCD (*Liquid Crystal Display*) ini dibuat menggunakan program Arduino 1.6.7 kemudian diunggah pada *Board* Arduino selanjutnya Arduino memproses dan menampilkan teks ke LCD (*Liquid Crystal Display*) sesuai dengan kondisi yang berlaku. Tampilan LCD (*Liquid Crystal Display*) dapat dilihat pada gambar 3.12, gambar 3.13 dan gambar 3.14.



Gambar 3.12 Rancangan LCD Saat Modul *Fingerprint* dan Palang Pintu Siaga



Gambar 3.13 Rancangan LCD Saat Input *Fingerprint* Cocok

Gambar 3.14 Rancangan LCD Saat Input *Fingerprint* Tidak Cocok

1.3 Analisis Kebutuhan

Dari perancangan sistem dapat dianalis alat dan bahan yang diperlukan dalam pembuatan sistem Kendali Palang Parkir Motor Khusus Dosen dan Karyawan IIB Darmajaya dengan *fingerprint* ini akan ditampilkan pada tabel Tabel 3.1 dan Tabel 3.2 .

1.3.1 Alat

Sebelum pembuatan sistem Kendali Palang Parkir Motor Khusus Dosen dan Karyawan IIB Darmajaya dengan *fingerprint* ada beberapa peralatan yang harus disiapkan. Daftar alat yang digunakan dalam penelitian ini akan dituliskan pada Tabel 3.1.

Tabel 3.1 Alat yang dibutuhkan

No	Alat	Spesifikasi	Fungsi	Jumlah
1	Laptop	AMD Quad-Core A9 , Ram 4 GB, Harddisk 1TB, Windows 10 Pro 64 Bit	Sebagai media untuk membuat program Arduino, fritzing dan rangkaian proteus	1 Unit
2	Kabel USB *(Universal Serial Bus)*		Sebagai media untuk mengunggah *scetch* ke papan Arduino	1 buah
3	Multitester	Digital	Sebagai alat ukur rangkaian	1 buah
4	Obeng	Obeng + dan -	Untuk melepas atau mengencangkan baut PCB pada box alat	1 buah
5	Solder		Untuk melunakkan dan menempelkan timah ke komponen	1 buah
6	Tang Potong		Untuk memotong kabel dan kaki komponen	1 buah
7	Pemotong *Acrylic*		Untuk memotong *Acrylic*	1 buah
8	Meter Ukur		Untuk mengukur jarak pada pengujian sensor ultrasonic	1 buah
9	Busur		Sebagai alat ukur pada pengujian motor servo	1 buah

1.3.2 Komponen

Dalam pembuatan pembuatan sistem Kendali Palang Parkir Motor Khusus Dosen dan Karyawan IIB Darmajaya dengan *fingerprint* ada komponen-komponen yang harus disiapkan. Daftar komponen yang digunakan dalam penelitian ini akan dituliskan pada Tabel 3.2.

Tabel 3.2 Komponen Yang Dibutuhkan

No	Nama Komponen	Spesifikasi	Fungsi	Jumlah
1	Kit Arduino	UNO	Sebagai pemproses alat	1 Set
2	Modul *Fingerprint* ZFM-60		Sebagai masukkan untuk membaca sidik jari	1 Buah
3	Sensor Ultrasonik		Sebagai pendeteksi kendaraan yang telah melewati palang parkir	2 buah
4	Motor Servo		Sebagai aktuator untuk menggerakkan palang parkir	1 Buah
5	LCD 16x4		Sebagai penampil informasi pada sistem parkir	1 Buah
6	PCB Bolong		Tempat meletakkan komponen rangkaian	1 Lembar
7	Transformator	2A	Mnurunkan tegangan listrik	1 Buah
8	Dioda	2A	Penyearah arus listrik	4 Buah
9	Kapasitor Elektrolit Condensator (Elco) 6800 uF		Sebagai penyimpan arus dan filter tegangan	1 Buah
10	Kapasitor Elektrolit Condensator (Elco) 220 uF		Sebagai penyimpan arus dan filter tegangan	2 Buah
11	IC 7805		IC regulator tegangan	1 Buah
12	IC 7812		IC regulator tegangan	1 Buah
13	Kabel Power		Sebagai penghubung dari tegangan AC ke Transformator	1 Buah
14	Kabel Pelangi		Sebagai jumper rangkaian	1 Meter
15	Timah Solder		Untuk menyolder	1 Roll
16	Lem Bakar		Sebagai isolator antar komponen	1 Buah
17	Box Rangkaian		Sebagai tempat meletakkan catu daya dan Arduino Kit	1 Buah
18	*Acrylic*	3 mm	Sebagai tempat meletakkan seluruh sistem	50 x 50 cm

1.3.3 *Software*

Tabel 3.3 memperlihatkan daftar *software* yang digunakan untuk merancang sistem Kendali Palang Parkir Motor Khusus Dosen dan Karyawan IIB Darmajaya.

Tabel 3.3 Daftar Aplikasi

No.	Nama Aplikasi	Fungsi
1	Arduino Versi 1.6.7	Untuk membuat *Scetch* Arduino dan mengunggah ke kit Arduino
2	Proteus 7 Profesional	Untuk merancang rangkaian elektronika
3	Fritzing.0.9.0b.32.pc	Untuk merancang pengkabelan (*wiring*) sistem secara keseluruhan menggunakan papan *Breadboard* secara *software*

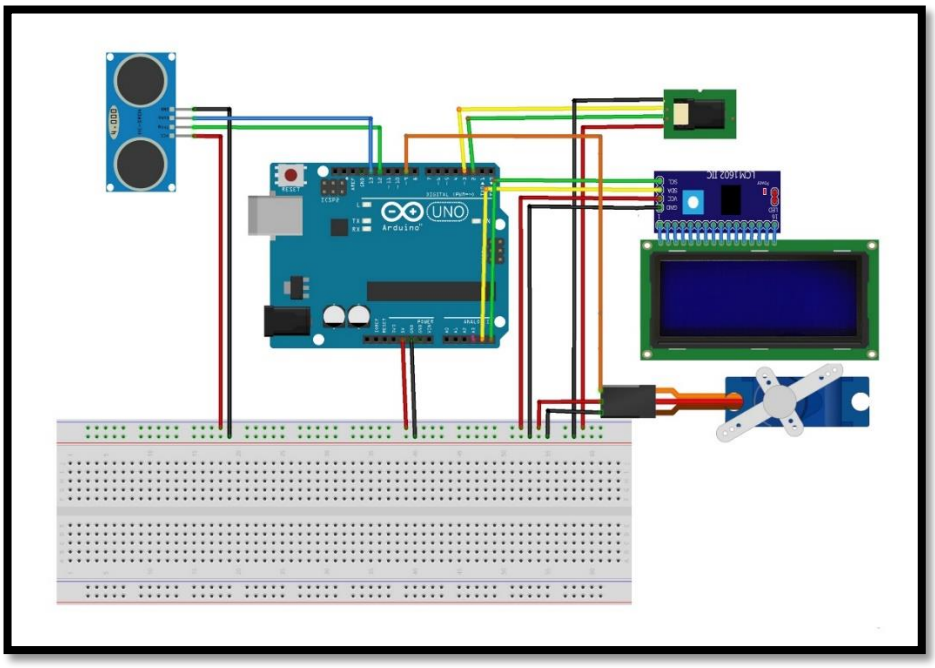
1.4 Implementasi

Setelah mengumpulkan alat dan bahan, langkah selanjutnya adalah melakukan implementasi rancangan alat yang telah dibuat. Pada tahap ini rancangan alat yang telah dibuat akan di implementasikan menjadi sistem yang sesungguhnya. Ada dua bagian dalam tahapan implementasi, diantaranya yaitu :

1. Implementasi perangkat keras
2. Implementasi perangkat lunak

1.4.1 Implementasi perangkat keras

Implementasi perangkat keras merupakan tahapan akhir dari perancangan yang telah dilakukan. Dalam tahap ini seluruh komponen dipasang sesuai dengan rancangan sistem yang telah dibuat sebelumnya. Implementasi perangkat keras dapat dilihat pada gambar 3.15



Gambar 3.15 Implementasi Perangkat Keras

1.4.2 Implementasi Perangkat Lunak

Penerapan perangkat lunak merupakan suatu tahap dimana program yang telah dirancang akan disimpan kedalam modul mikrokontroler melalui *downloader* dan menggunakan *software* tertentu sesuai dengan bahasa pemograman yang akan digunakan. Disini peneliti menggunakan bahasa C dan menggunakan *software* Arduino. Pada *software* Arduino program ditulis kemudian di*compile*, tujuanya adalah untuk mengetahui apakah program yang dibuat sudah benar atau belum. Langkah terakhir yaitu mengunggah program kedalam modul mikrokontroler. Untuk bisa mengunggah program ke Arduino Uno yang pertama harus mengatur port yang digunakan oleh Arduino. Pengaturan port Arduino menggunakan port COM1. Setelah pengaturan port langkah selanjutnya yaitu meng-*compile* program. Setelah program berhasil di *compile* selanjutnya yaitu mengunggah file ke Arduino Uno.

1.4.2.1 *Enroll fingerprint*

Implementasi *enroll* dilakukan dengan cara mengunggah *scetch* Arduino sebagai berikut:

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>

uint8_t id;
uint8_t getFingerprintEnroll();

SoftwareSerial mySerial(2, 3);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);

void setup()
{
  while (!Serial);
  delay(500);
  Serial.begin(9600);
  Serial.println("Adafruit Fingerprint sensor enrollment");
  finger.begin(57600);

  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1);
  }
}

uint8_t readnumber(void) {
  uint8_t num = 0;
  boolean validnum = false;
  while (1) {
    while (! Serial.available());
    char c = Serial.read();
    if (isdigit(c)) {
       num *= 10;
       num += c - '0';
       validnum = true;
    } else if (validnum) {
      return num;
    }
  }
}

void loop()
{
  Serial.println("Ready to enroll a fingerprint! Please Type in the ID # you want
to save this finger as...");
  id = readnumber();
  Serial.print("Enrolling ID #");
  Serial.println(id);

  while (!  getFingerprintEnroll() );
}

uint8_t getFingerprintEnroll() {

  int p = -1;
  Serial.print("Waiting for valid finger to enroll as #"); Serial.println(id);
  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image taken");
      break;
```

```
    case FINGERPRINT_NOFINGER:
      Serial.println(".");
      break;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      break;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Imaging error");
      break;
    default:
      Serial.println("Unknown error");
      break;
    }
  }

  p = finger.image2Tz(1);
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image converted");
      break;
    case FINGERPRINT_IMAGEMESS:
      Serial.println("Image too messy");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      return p;
    case FINGERPRINT_FEATUREFAIL:
      Serial.println("Could not find fingerprint features");
      return p;
    case FINGERPRINT_INVALIDIMAGE:
      Serial.println("Could not find fingerprint features");
      return p;
    default:
      Serial.println("Unknown error");
      return p;
  }

  Serial.println("Remove finger");
  delay(2000);
  p = 0;
  while (p != FINGERPRINT_NOFINGER) {
    p = finger.getImage();
  }
  Serial.print("ID "); Serial.println(id);
  p = -1;
  Serial.println("Place same finger again");
  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image taken");
      break;
    case FINGERPRINT_NOFINGER:
      Serial.print(".");
      break;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      break;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Imaging error");
      break;
    default:
      Serial.println("Unknown error");
      break;
    }
  }

  p = finger.image2Tz(2);
  switch (p) {
    case FINGERPRINT_OK:
```

```
      Serial.println("Image converted");
      break;
    case FINGERPRINT_IMAGEMESS:
      Serial.println("Image too messy");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      return p;
    case FINGERPRINT_FEATUREFAIL:
      Serial.println("Could not find fingerprint features");
      return p;
    case FINGERPRINT_INVALIDIMAGE:
      Serial.println("Could not find fingerprint features");
      return p;
    default:
      Serial.println("Unknown error");
      return p;
  }

  // OK converted!
  Serial.print("Creating model for #");  Serial.println(id);

  p = finger.createModel();
  if (p == FINGERPRINT_OK) {
    Serial.println("Prints matched!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_ENROLLMISMATCH) {
    Serial.println("Fingerprints did not match");
    return p;
  } else {
    Serial.println("Unknown error");
    return p;
  }

  Serial.print("ID "); Serial.println(id);
  p = finger.storeModel(id);
  if (p == FINGERPRINT_OK) {
    Serial.println("Stored!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_BADLOCATION) {
    Serial.println("Could not store in that location");
    return p;
  } else if (p == FINGERPRINT_FLASHERR) {
    Serial.println("Error writing to flash");
    return p;
  } else {
    Serial.println("Unknown error");
    return p;
  }
}
```

Gambar 3.16 *Scetch* Program *Scetch Enroll* Sidik Jari

1.4.2.2 *Delete* data *fingerprint*

Implementasi *delete* data *fingerprint* dilakukan dengan cara mengunggah *scetch* Arduino sebagai berikut:

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>

uint8_t getFingerprintEnroll(uint8_t id);
SoftwareSerial mySerial(2, 3);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);
void setup()
{
  Serial.begin(9600);
  Serial.println("Delete Finger");

  finger.begin(57600);

  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1);
  }
}

void loop()
{
  while (!Serial);
  delay(500);

  Serial.println("Type in the ID # you want delete...");
  uint8_t id = 0;
  while (true) {
    while (! Serial.available());
    char c = Serial.read();
    if (! isdigit(c)) break;
    id *= 10;
    id += c - '0';
  }
  Serial.print("deleting ID #");
  Serial.println(id);

  deleteFingerprint(id);
}

uint8_t deleteFingerprint(uint8_t id) {
  uint8_t p = -1;

  p = finger.deleteModel(id);

  if (p == FINGERPRINT_OK) {
    Serial.println("Deleted!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_BADLOCATION) {
    Serial.println("Could not delete in that location");
    return p;
  } else if (p == FINGERPRINT_FLASHERR) {
    Serial.println("Error writing to flash");
    return p;
  } else {
    Serial.print("Unknown error: 0x"); Serial.println(p, HEX);
    return p;
  }
}
```

Gambar 3.17 *Scetch* Program *Delete Database* Sidik Jari

1.4.2.3 *Matching Fingerprint*

Implementasi *matching fingerprint* dilakukan dengan cara mengunggah *scetch* Arduino sebagai berikut:

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>

int getFingerprintIDez();

SoftwareSerial mySerial(2, 3);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);

void setup()
{
  while (!Serial);
  Serial.begin(9600);
  Serial.println("Adafruit finger detect test");
  finger.begin(57600);

  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1);
  }
  Serial.println("Waiting for valid finger...");
}

void loop()
{
  getFingerprintIDez();
  delay(50);
}

uint8_t getFingerprintID() {
  uint8_t p = finger.getImage();
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image taken");
      break;
    case FINGERPRINT_NOFINGER:
      Serial.println("No finger detected");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      return p;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Imaging error");
      return p;
    default:
      Serial.println("Unknown error");
      return p;
  }

  p = finger.image2Tz();
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image converted");
      break;
    case FINGERPRINT_IMAGEMESS:
      Serial.println("Image too messy");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      return p;
    case FINGERPRINT_FEATUREFAIL:
```

```
      Serial.println("Could not find fingerprint features");
      return p;
    case FINGERPRINT_INVALIDIMAGE:
      Serial.println("Could not find fingerprint features");
      return p;
    default:
      Serial.println("Unknown error");
      return p;
  }

  p = finger.fingerFastSearch();
  if (p == FINGERPRINT_OK) {
    Serial.println("Found a print match!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_NOTFOUND) {
    Serial.println("Did not find a match");
    return p;
  } else {
    Serial.println("Unknown error");
    return p;
  }

  Serial.print("Found ID #"); Serial.print(finger.fingerID);
  Serial.print(" with confidence of "); Serial.println(finger.confidence);
}

int getFingerprintIDez() {
  uint8_t p = finger.getImage();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.image2Tz();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.fingerFastSearch();
  if (p != FINGERPRINT_OK)  return -1;

  Serial.print("Found ID #"); Serial.print(finger.fingerID);
  Serial.print(" with confidence of "); Serial.println(finger.confidence);
  return finger.fingerID;
}
```

Gambar 3.18 *Scetch* Program *Matching* Sidik Jari

1.5 Uji Coba

Uji coba sistem dilakukan untuk mengetahaui kinerja sistem, apakah rangkaian dan program yang telah dibuat berjalan sesuai dengan rancangan. Beberapa uji coba yang akan dilakukan untuk memastikan bahwa sistem yang dibuat telah bekerja dengan baik. Uji coba akan dilakukan mulai dari pengujian permodul sampai dengan pengujian keseluruhan sistem.

Prosedur Pengujian:

1. Mengaktifkan catu daya.
2. Membuat *scetch* arduino sesuai dengan sistem yang akan dirancang.

3. Mengunggah program Arduino.
4. Menghubungkan modul sensor *fingerprint.*
5. Menghubungkan modul LCD (*Liquid Crystal Display*) 20x4 ke arduino.
6. Menghubungkan sensor ultrasonik ke arduino.
7. Menghubungkan motor servo ke arduino.
8. Mencatat hasil pengujian.

1.5.1 Uji Coba Modul Sensor *Fingerprint*

Uji coba modul sensor *fingerprint* digunakan untuk mengetahui seberapa cepat modul *fingerprint* dalam membaca data sidik jari serta kecepatan *fingerprint* dalam mengirim sinyal ke arduino. Uji coba modul sensor *fingerprint* dilakukan dengan cara mengunggah *scetch* arduino yang telah dibuat kemudian masuk ke menu serial monitor arduino.

1.5.1.1 Uji Coba *Enroll*

Pada tahap ini modul sensor *fingerprint* akan dicoba untuk mengambil sampel sidik jari kemudian menyimpannya di *database* yang ada di modul sensor *fingerprint.* Uji coba *enroll* dilakukan untuk menguji coba apakah modul sensor *fingerprint* dapat merekam dan menyimpan data sidik jari baru.

1.5.1.2 Uji Coba *Delete*

Pada tahap ini dilakukan pengujian penghapusan rekaman sidik jari yang ada pada *database* modul *sensor fingerprint.* Uji coba *delete* dilakukan untuk menguji coba apakah modul sensor *fingerprint* dapat menghapus rekaman yang sudah ada di *database* modul sensor *fingerprint*.

1.5.1.3 Uji Coba *Matching Fingerprint*

Pada tahap ini dilakukan pengujian pencocokkan rekaman sidik jari yang ada pada *database* modul *sensor fingerprint.* Uji coba *matching fingerprint* dilakukan untuk

menguji coba apakah modul sensor *fingerprint* mampu mencocokkan sidik jari input dengan sidik jari yang ada di *database* modul sensor *fingerprint*.

1.5.2 Uji Coba Sensor Ultrasonik

Uji coba sensor ultrasonik digunakan untuk mengetahui seberapa akurat sensor dalam mendeteksi objek dan jarak yang mampu diterima sensor ultrasonik. Pengujian sensor ultrasonik HC-SR04 ini dilakukan dengan cara menghubungkan sensor ultrasonik HC-SR04 ke Modul Arduino Uno sesuai dengan *datasheet* sensor ultrasonik HC-SR04.

1.5.3 Uji Coba Motor Servo

Uji coba motor servo digunakan untuk mengetahui seberapa cepat motor servo dalam membuka dan menutup palang parkir. Pengujian motor servo dilakukan dengan menghubungkan motor servo ke catu daya 5 *Volt* dan dengan cara memberikan sinyal input pada motor servo melalui pin 9 yang terdapat pada Arduino Uno.

1.5.4 Pengujian Keseluruhan

Tujuan pengujian sistem keseluruhan adalah mengetahui apakah secara keseluruhan alat dapat bekerja dengan baik sesuai perencanaan.

1.6 Analisis Kinerja

Analisis kerja dilakukan guna mengetahui kinerja dari sistem yang telah dibuat dan akan dianalisis berdasarkan respon modul sensor *fingerprint* dalam membaca sidik jari dan respon sensor ultrasonik dalam mendeteksi objek serta respon dan kecepatan motor servo dalam membuka dan menutup palang parkir.