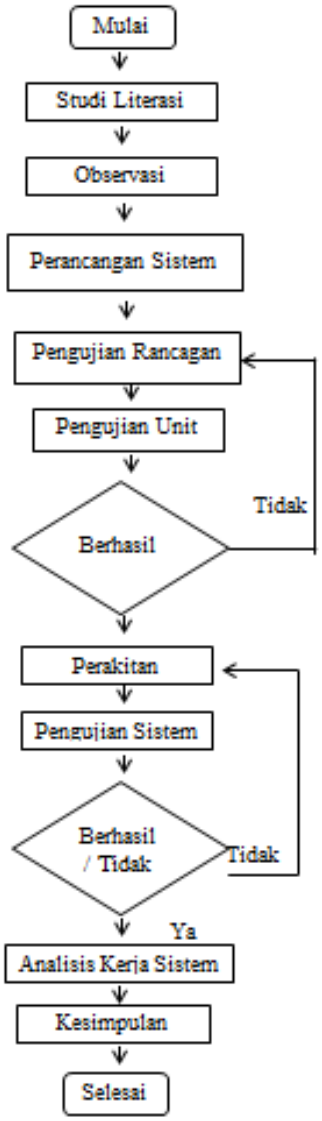
BAB III

METODOLOGI PENELITIAN

3.1. *Flowchart* Penelitian



Gambar 3. 1. *Flowchart* Penelitian

3.2. Rincian Alur Penelitian

Pada tahap ini adalah rincian alur penelitian berdasarkan *flowchart* penelitian pada Gambar 3.1 .

3.2.1. Studi Literatur

Di sini, studi literatur dilakukan untuk mencari berbagai sumber yang berkaitan dengan perancangan sistem akses gerbang yang menggunakan teknologi *RFID* berbasis *ESP32*. Sumber-sumber ini diperoleh dari buku, jurnal, *internet*, dan sumber lainnya. Beberapa materi yang dicari meliputi informasi tentang *RFID*, *ESP32*, *Firebase* dan semua literatur yang akan digunakan..

3.2.2 Alat dan Bahan

Komponen dan peralatan yang digunakan untuk penelitian sistem akses gerbang menggunakan *RFID* berbasis *ESP32* diantaranya sebagai berikut :

Tabel 3. 1. Alat dan Bahan

NO	Nama	Spesifikasi	Jumlah
1	Mikrokontroler *ESP32*	-	2 Unit
2	*RFID*-RC522	-	2 Unit
3	Kabel *Jumper*	-	-
4	*Breadboard*	-	-
5	*Power Supply*	5VDC	6 buah
6	*Micro SD*	-	2 Buah
7	Tag *RFID*	-	3 Buah
8	*Relay*	5V	4 Buah
9	Kabel *USB*	-	1 Buah
10	PC	-	1 Unit
11	*Software Arduino IDE*	-	-
12	*Akun Firebase*	-	-

13	Jaringan Internet	-	-
14	*DF mini player*		2 buah
15	*TRSS modul*		2 buah
16	*Infrared Sensor*		4 buah
17	*Loud Speaker*		2 buah
18	*Push Button*		4 buah
19	*RTC Module*		2 buah

3.2.3. Lokasi Penelitian

Kegiatan penelitian ini akan dilaksanakan di lokasi penelitian yaitu di lingkungan Universitas Silwangi, Jalan Siliwangi No.24 Kota Tasikmalaya.

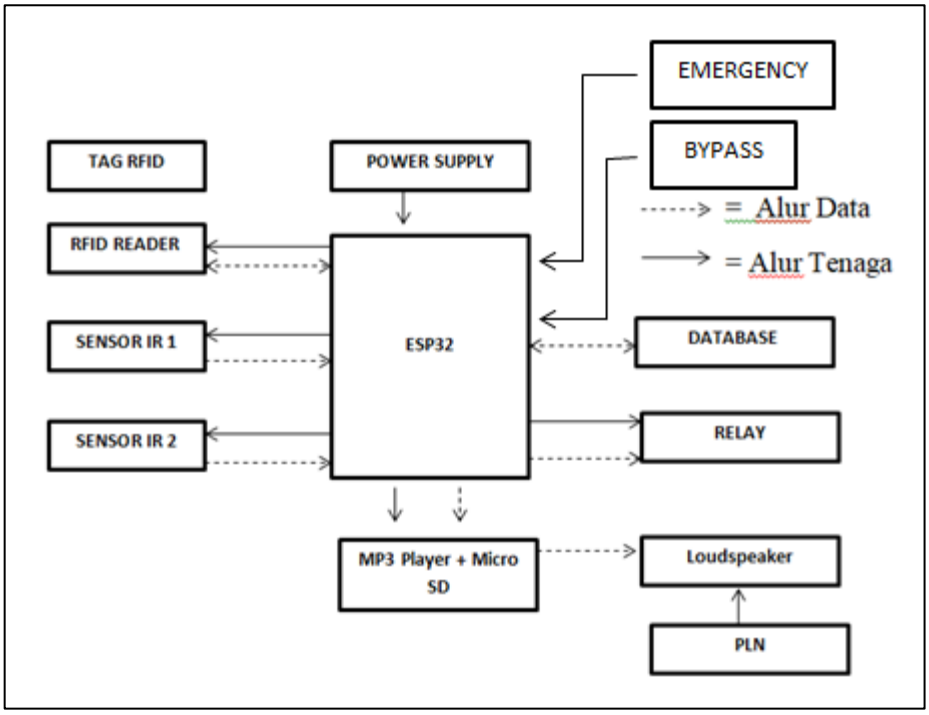
3.2.4. Perancangan Sistem

Perancangan sistem akses gerbang menggunakan teknologi *RFID* berbasis *ESP32* dilakukan dengan beberapa tahapan berikut:

1. Identifikasi kebutuhan sistem: Langkah pertama adalah mengidentifikasi kebutuhan sistem. Hal ini meliputi tujuan sistem akses gerbang, jenis pengguna yang akan mengakses gerbang, jenis tag *RFID* yang akan digunakan, dan alat apa saja yang diperlukan untuk mengimplementasikan sistem akses gerbang.
2. Perancangan skema sistem: Setelah mengidentifikasi kebutuhan sistem, langkah selanjutnya adalah merancang skema sistem akses gerbang. Skema ini harus mencakup semua komponen sistem dan bagaimana komponen tersebut saling terhubung. Skema ini juga harus mempertimbangkan ketersediaan daya listrik, konektivitas *internet*, dan sinyal *RFID*.

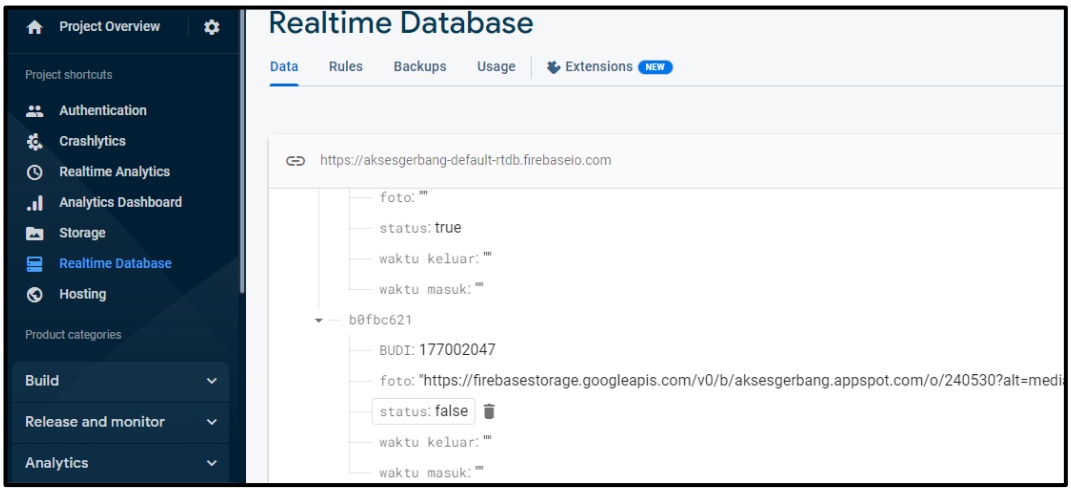
3. Pembuatan perangkat keras: Setelah merancang skema sistem, langkah selanjutnya adalah membuat perangkat keras. Perangkat keras yang diperlukan untuk sistem akses gerbang meliputi *ESP32*, modul *RFID*, *Relay, Semsor Infrared, DF mini mp3 player* dan *Loudspeaker*.
4. Konfigurasi *Firebase*: Selanjutnya, *Firebase* harus dikonfigurasi untuk sistem akses gerbang. *Firebase* akan digunakan untuk menyimpan daftar pengguna yang diizinkan dan log akses gerbang.

3.2.4.1 Skema Sistem



Gambar 3. 2. Arsitektur Sistem

3.2.4.2 Konfigurasi *Firebase*



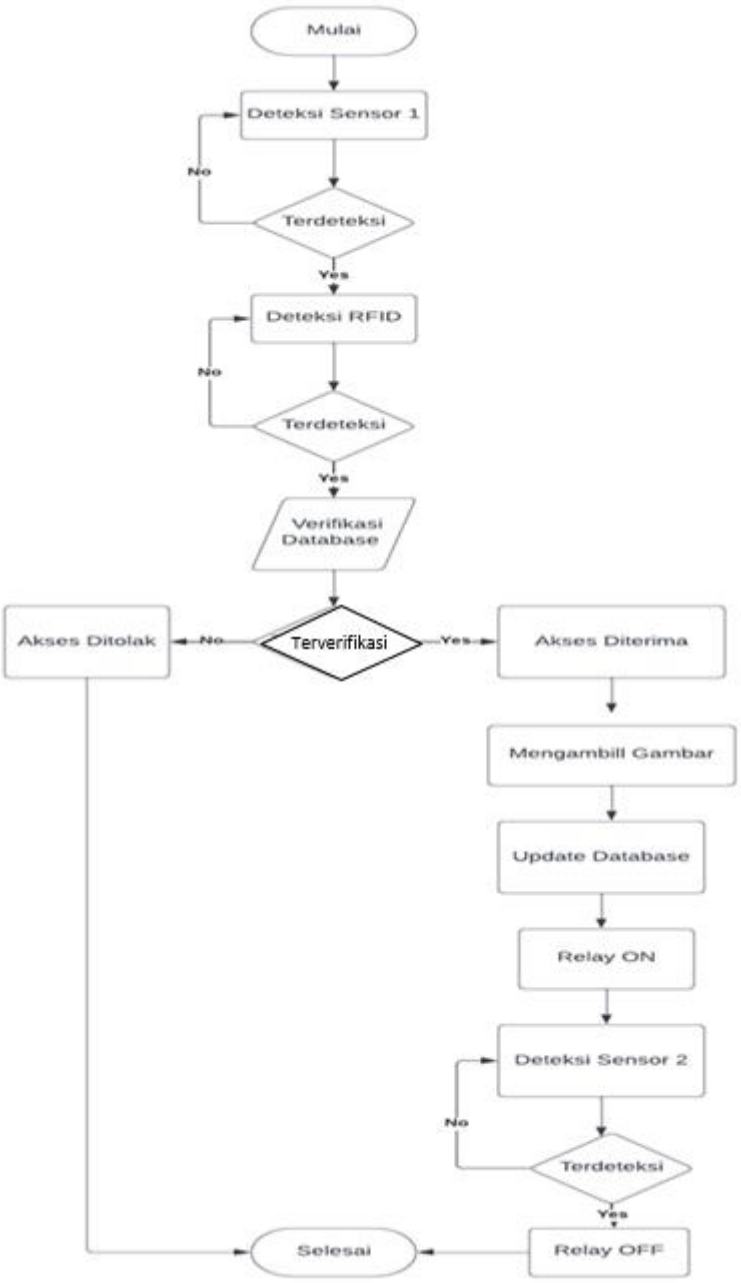
Gambar 3. 3. Struktur *Database*

Dalam konfigurasi *Database Firebase* berikut langkah-langkahnya :

1. Membuat akun proyek *Firebase*
2. Membuat *Database*
3. Mengkonfigurasi autentikasi dan perizinan akses
4. Menyusun sturuktur data.

3.2.4.3 *Flowchart* Sistem

A. Sistem *RFID*



Gambar 3. 4. *Flowchart* Sistem *RFID*

Berikut adalah penjelasan *flowchart* sistem pada Gambar 13 :

1. Deteksi Sensor 1

Jika sensor mendeteksi adanya objek maka sistem deteksi RFID data digunakan. Joika sensor tidak mendeteksi maka sistem deteksi RFID tidak dapat digunakan.

2. Deteksi Tag *RFID*

   Jika *RFID reader* mendeteksi adanya Tag *RFID* dalam jangkauan maka proses akan berlanjut , jika tidak terdapat Tag *RFID* dalam jangkauan maka *RFID reader* akan terus melakukan deteksi.

3. .Verfikasi *Database*

   Setelah Tag *RFID* terdeteksi maka *ESP32* akan mengirimkan data ke *Database* untuk melakukan dua langkah verifikasi, yaitu :

   a. Otoritasi : data tersebut memiliki otoritas jika terdaftar di *Database*
   b. Otentikasi : data tersebut terotentikasi jika status di *Database* sesuai .

4. Akses Diterima

   Data yang di *Entry* dari Tag *RFID* diterima jika terverifikasi dengan *Database*. Jika pengguna ingin mengakses gerbang masuk maka data yang di *Entry* harus memiliki otoritas terdaftar di *Database* dan terotentikasi status nya belum memasuki gerbang masuk. Jika pengguna ingin mangakses gerbang keluar maka data yang di *Entry* harus memiliki otoritas terdaftar di *Database* dan terotentikasi sudah memasuki gerbang masuk.

5. Mengambil Gambar

   Setelah akses diterima maka *ESP32* akan memerintah kan *ESP32-CAM* untuk mengambil gambar *user*.

6. Akses Ditolak

Data yang di *Entry* dari tag *RFID* ditolak jika tidak terverifikasi dengan *Database.*

7. Update *Database*

   Jika penggguna telah diizinkan masuk maka status di *Database* terupdate menjadi sudah memasuki gerbang masuk dan jika pengguna diizinkan mengakses gerbang keluar maka status di *Database* terupdate menjadi sudah keluar atau belum memasuki gerbang masuk. Selain status ,data yang di *update* juga yaitu foto, waktu masuk, dan waktu keluar.

8. *Relay ON*

   Relay menyala untuk membuka gerbang.

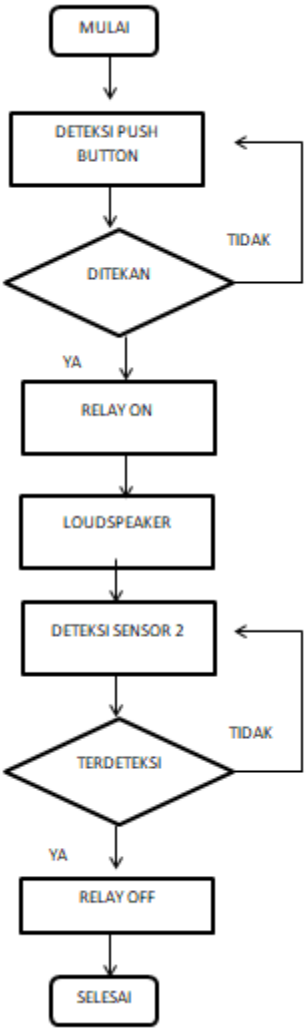
9. Deteksi Sensor 2

   Jika Sensor mendeteksi objek maka lanjut ke perintah selanjutnya yaitu *Relay OFF*.

10. *Relay OFF*

    Relay berubah ke kondisi *OFF* untuk menutup gerbang.

B. Sistem *Bypass*



Gambar 3. 5. *flowchart bypass system*

Sistem ini tidak memerlukan identifikasi tag *RFID* sebagai syarat akses sistem. Tujuan diadakan sistem ini adalah untuk memudahkan tamu yang tidak memiliki tag *RFID* agar bisa mengakses gerbang. Ketika *Push Button* ditekan maka akan mengaktifkan *relay* gerbang dan *relay* sistem tiket yang sudah ada.

C. Sistem *Emergency*

Sistem ini bertujuan untuk keadaan darurat dengan menekan *pushbutton emergency* maka relay akan menyala terus hingga sistem direset.

3.2.5. Pengujian Rancangan

Langkah pengujian rancangan adalah dengan menguji integrasi perangkat keras dengan *Firebase*. Berikut langkah-langkah, yaitu :

1. Konfigurasi Perangkat Keras: Pastikan perangkat keras, dalam hal ini *ESP32*, terhubung dengan jaringan *Wi-Fi* yang sama dengan *Firebase*. mengatur *SSID* dan kata sandi yang sesuai di *ESP32* agar dapat terhubung ke jaringan.
2. Konfigurasi *Firebase*: Buat proyek Firebase di *console Firebase* dan dapatkan informasi kredensial seperti *API Key*, ID Proyek, dan *URL Database Firebase* yang akan digunakan dalam integrasi. Pastikan telah mengaktifkan layanan *Realtime Database* .
3. Pustaka *Firebase*: Instal dan konfigurasi pustaka *Firebase* di perangkat keras *ESP32* .
4. Koneksi *Firebase* dengan *ESP32*: Dalam kode program *ESP32* , sertakan pustaka *Firebase* dan gunakan kredensial yang diperoleh dari proyek *Firebase* untuk menginisialisasi koneksi dengan *Firebase*. Pastikan telah mengatur *API Key*, ID Proyek, dan *URL Database Firebase* dengan benar.
5. Mengirim Data ke *Firebase*: Setelah koneksi berhasil, dapat mengirim data dari perangkat keras *ESP32* ke *Firebase*.
6. Pengaturan Aturan *Firebase*: *Di console Firebase,* konfigurasikan aturan untuk memproses data yang diterima dari perangkat keras. Aturan ini menentukan bagaimana data akan disimpan, diakses, dan diproses dalam *database Firebase*.

7. Pengujian Integrasi: Setelah pengaturan selesai, lakukan pengujian untuk memastikan bahwa perangkat keras *ESP32* dapat mengirim data ke *Firebase* dengan benar. Periksa apakah data yang dikirim dari *ESP32* dapat terlihat dan diproses dengan benar di *Firebase.*

```
#include <WiFi.h>
#include <FirebaseESP32.h>
// Masukkan informasi SSID dan password WiFi Anda
const char* ssid = "iPhone";
const char* password = "12345678";
// Masukkan informasi koneksi Firebase Realtime Database
#define FIREBASE_HOST "https://test-bc880-default-rtdb.firebaseio.com"
#define FIREBASE_AUTH "dKbOToZ0FYURZs78gZWYmgvoXRyLYVkuj9zgxDnH"
FirebaseData firebaseData;
String firebasePath = "/data";

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  Serial.print("Connecting to WiFi...");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();
  Serial.print("Connected to WiFi. IP address: ");
  Serial.println(WiFi.localIP());
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);
  String data="ROHMAN MAOLANA HIDAYATH";
  Firebase.pushString(firebaseData, firebasePath, data);
}
void loop() {}
```

Gambar 3. 6. Program Pengujian Integrasi Komunikasi

Gambar 3. 6 adalah program untuk pengujian integrasi antara *ESP32* dan *firebase* dengan mengirimkan data ke *database* lalu data yang dikirim ditampilkan di *realtime database* .

3.2.6. Pengujian Unit

Pada tahap ini dilaksanakan pengujian fungsionalitas masing-masing komponen perangkat keras secara terpisah.

3.2.6.1 Mikrokontroler

Tujuan dari pengujian *ESP32* sebagai mikrokontroler utama pada penelitian ini adalah seperti di bawah ini:

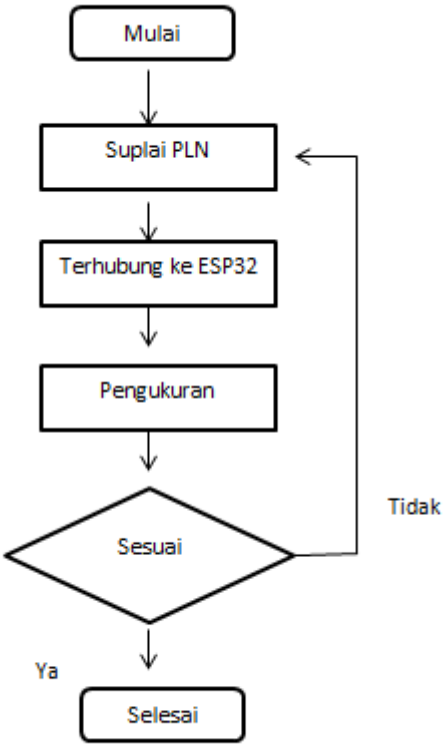
1. Mengetahui kesesuaian pin I/O pada *ESP32* dengan pin I/O dari komponen-komponen yang digunakan untuk sistem akses gerbang.
2. Mengetahui kondisi tegangan *ESP32* saat tidak ada komponen lain terpasang dan saat dalam menyuplai tegangan terhadap komponen-komponen yang digunakan untuk sistem akses gerbang.
3. Mengetahui apakah diperlukan suplai daya dari luar *ESP32* untuk mengoperasikan komponen-komponen utama dan pendukung.

Berdasarkan tujuan pengujian *ESP32* sebagai mikrokontroler maka metode pengujian yang dilakukan yaitu :

1. Memperhitungkan pin I/O pada *ESP32* dengan pin I/O dari komponen-komponen yang digunakan untuk sistem akses gerbang.
2. Mengukur tegangan *ESP32* saat tidak ada komponen lain terpasang dan saat dalam menyuplai tegangan terhadap komponen-komponen yang digunakan untuk sistem akses gerbang.
3. Memperhitungkan suplai daya dari luar *ESP32* untuk mengoperasikan komponen-komponen utama dan pendukung.

3.2.6.2 Power Supply

Power Supply yang digunakan adalah *adaptor* dengan kerja mengubah tegangan AC dari sumber (suplai PLN 220 V) menjadi tegangan DC dengan besar 5 V sebagai *output*. Nilai tersebut disesuaikan dengan kebutuhan suplai listrik menuju *ESP32*.



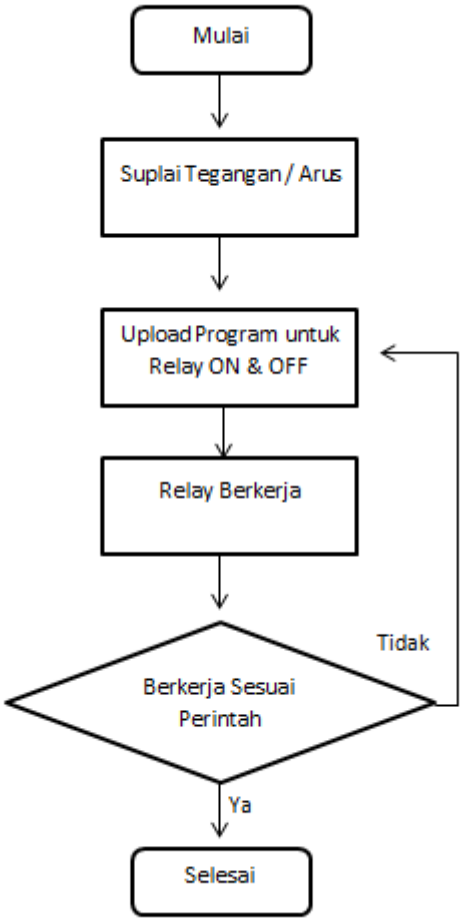
Gambar 3. 7. *flowchart* pengujian *power supply*

Gambar 3.7 menunjukkan flowchart dari pengujian *power supply*. Pengujian *power supply* adalah untuk mengetahui *power supply* dapat menyuplai dengan sumber suplai listrik dari PLN dan untuk mengetahui besar tegangan dan besar arus yang mengalir saat suplai dari suplai PLN menuju *ESP32*. Tegangan suplai PLN 220 V diturunkan oleh adaptor sehingga menghasilkan kesesuaian tegangan keluaran

dengan besar yang diinginkan. Keluaran tersebut akan diukur oleh *multitester*. Jika tegangan tidak terukur, maka dilakukan pengecekan, perbaikan, pergantian *adaptor* yang digunakan, atau dilakukan penyesuaian pengukuran sampai didapatkan hasil pengukuran dan pengujian selesai.

Metode pengujian berdasarkan *flowchart* dari pengujian *power supply* yaitu mengukur tegangan output dari catu daya sebanyak 5 kali dengan interval waktu 5 detik.

3.2.6.3 Relay



Gambar 3. 8. *flowchart* pengujian *power supply*

Pada Gambar 3.8 menunjukan *flowchart* pengujian *relay* yang bertujuan untuk mengetahui ketepatan respon *relay* terhadap perintah dari mikrokontroler yang akan memutuskan atau menyambungkan aliran arus *OUTPUT* dari *relay* tersebut.

```
#define RELAY_PIN 21

void setup(){

  pinMode(RELAY_PIN, OUTPUT);

}


void loop(){
  digitalWrite(21 ,LOW);
  delay(3000);
  digitalWrite(21,HIGH);
  }
```

Gambar 3. 9. Program pengujian relay

Pada Gambar 3.9 adalah program pengujian *relay* dengan *ESP32* untuk mengendalikan *relay* dengan perintah *relay* hidup, lalu *delay* selama 3 detik, kemadian *relay* mati.

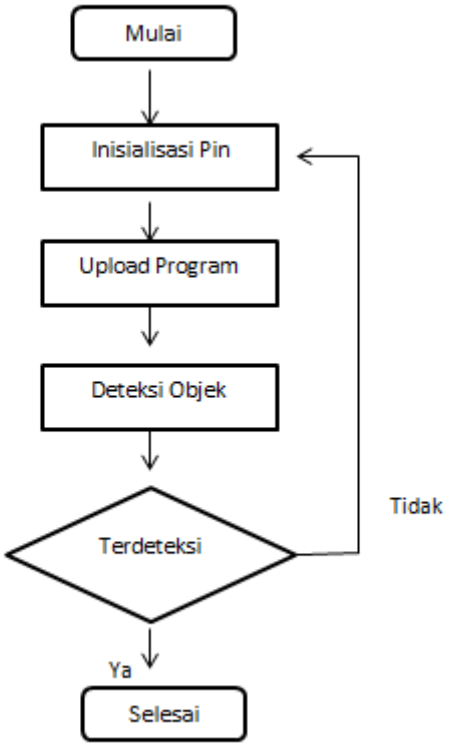
Tabel 3. 2. Konektivitas Pin *Relay* Dan *ESP32*

RELAY	**ESP32**
VCC	5V
GND	GND
OUT	12

Tabel 3. 2 adalah *wiring* antara *relay* dan *ESP32* dengan pin 12 sebagai *output* untuk mengatur *relay*.

Metode pengujian yang digunakan yaitu memberi *input LOW* dan *HIGH* secara bergantian dengan interval waktu 3 detik.

3.2.6.4 Sensor IR Proximity



Gambar 3. 10. *Flowchart* Pengujian Sensor *IR Proximity*

Pada Gambar 3. 10 adalah *flowchart* pengujian sensor *IR proximity* (kedekatan) melibatkan verifikasi kemampuan sensor untuk mendeteksi objek atau hambatan yang berada dalam jarak dekat dengan sensor tersebut. Langkah pengujian jarak deteksi yaitu letakkan objek pada jarak menggunakan alat ukur penggaris lalu hasilnya ditampilkan di *serial monitor*.

Pengujian ini membantu memverifikasi kinerja sensor *IR proximity* dan memastikan bahwa sensor dapat diandalkan dalam sistem yang membutuhkan deteksi objek.

```
const int IR_PIN = 2; // Pin sensor IR proximity dihubungkan ke GPIO 2 pada ESP32

void setup() {
  Serial.begin(115200);
  pinMode(IR_PIN, INPUT);
}

void loop() {
  int proximityValue = digitalRead(IR_PIN);

  if (proximityValue == HIGH) {
    Serial.println("Tidak ada objek.");
  } else {
    Serial.println("Objek terdeteksi!");
  }

  delay(3000);
}
```

Gambar 3. 11. Program Pengujian Sensor IR Proximity

Pada Gambar 3. 11 adalah program sensor *IR proximity* untuk mendeteksi objek menggunakan mikrokontroler *ESP32*. Ketika tidak ada objek terdeteksi maka serial monitor menampilkan “Tidak ada objek”. Apabila sensor mendeteksi ada objek maka serial monitor menampilkan “Objek Terdeteksi”.

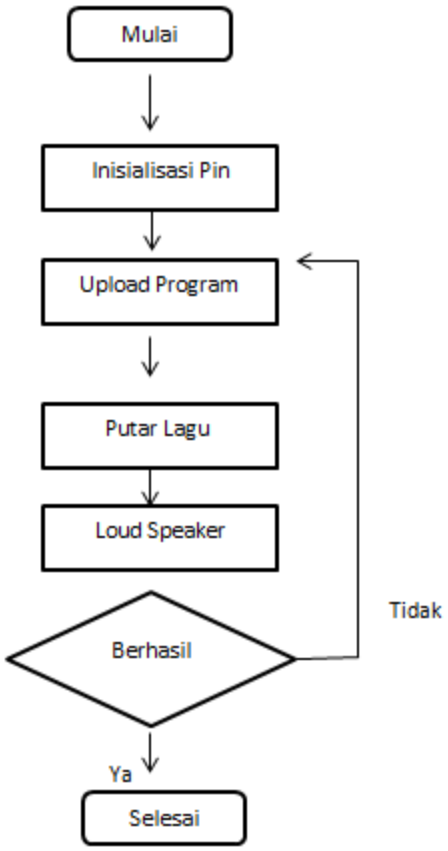
Tabel 3. 3. Konektivitas Pin *ESP32* dan *IR Proximity*

IR PROXIMITY	**ESP32**
VCC	5V
GND	GND
OUT	2

Pada Tabel 3. 3 adalah konfigurasi pin yang digunakan antara *ESP32* dan *IR Proximity* dengan pin 2 sebagai *input* nilai *proximity*.

Metode pengujian *IR Proximity* yaitu melaksanakan 12 kali pengujian pendeteksian objek untuk mencatat jarak maksimal jangkauan.

3.2.6.5 *DF Mini MP3 Player*



Gambar 3. 12. *Flowchart* Pengujian *DF Mini MP3 Player*

Pada Gambar 3. 12 adalah *flowchart* pengujian *DF Mini MP3 Player* melibatkan serangkaian langkah untuk memverifikasi fungsi dan kinerja pemutar *MP3*. Berikut adalah beberapa langkah yang dapat dilakukan dalam pengujian *DF Mini MP3 Player*:

1. Pengujian *Format Audio* yang Didukung: *Transfer* beberapa *file audio* dengan *format* yang didukung oleh *DF Mini MP3 Player* ke kartu SD atau memori *internal*nya. Pastikan pemutar *MP3* dapat membaca dan memutar file-file tersebut. *Format audio* yang umum didukung meliputi *MP3, WAV, WMA*, dan *AAC*.
2. Pengujian *Output Audio*: Hubungkan *DF Mini MP3* Player ke *speaker* atau *headphone*, dan putar beberapa *file audio*. Lalu catat apakah *DF Mini MP3 Player* berkerja atau tidak.

```
#include <DFRobotDFPlayerMini.h>
#include <SoftwareSerial.h>
#define MP3_RX_PIN 16 // RX MP3 terhubung ke TX2 ESP32
#define MP3_TX_PIN 17 // TX MP3 terhubung ke RX2 ESP32

SoftwareSerial mp3Serial(MP3_RX_PIN, MP3_TX_PIN);
DFRobotDFPlayerMini mp3Player;
void setup(){
 mp3Serial.begin(9600);
  delay(100);
  mp3Player.begin(mp3Serial);

  mp3Player.volume(20);
}
void loop(){
  playAudio(1);
  delay(3000);
  }
  void playAudio(uint16_t fileNumber) {
  mp3Player.play(fileNumber);
}
```

Gambar 3. 13. Program *Pengujian DF Mini MP3 Player*

Pada Gambar 3. 13 adalah program untuk memutar *mp3* pada urutan pertama lalu *looping* dengan jeda 3 detik.

Tabel 3. 4. Konektivitas Pin antara *DF Mini MP3 Player, ESP32,* dan *TRRS*

DF MINI MP3 PLAYER	**ESP32**	**TRRS**
VCC	5V	
GND	GND	GND
RX	TX2	
TX	RX2	
DAC_R		RING 1
DAC_1		TIP

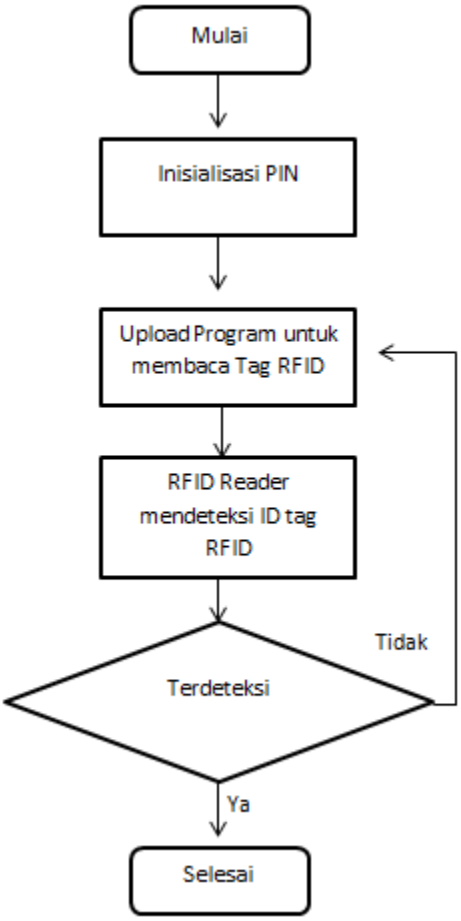
Tabel 3. 4 adalah konfigurasi pin antara *DF Mini MP3 Player, ESP32,* dan *TRRS* dalam pengujian *MP3 Player* menggunakan program komunikasi serial.

3.2.7. Pengujian Deteksi *RFID*

Tujuan pengujian deteksi *RFID* dengan cara mendeteksi tag *RFID* adalah untuk mengetahui pengaruh dari beberapa variabel terhadap *RFID reader*.

3.2.7.1 Identifikasi Tag

Menguji kemampuan unit RFID untuk mengenali dan membaca tag RFID yang berbeda. Ini melibatkan meletakkan tag RFID di dekat pembaca atau antena RFID dan memastikan bahwa tag tersebut terdeteksi dan dibaca dengan benar.



Gambar 3. 14. *Flowchart* pengujian identifikasi tag *RFID*

Gambar 3.14 adalah gambar untuk *Flowchart* yang digunakan dalam pengujian identifikasi tag *RFID* . Berdasarkan tujuan pengujian identifikasi tag maka metode pengujian yang dilakukan yaitu tag didekatkan terhadap *Reader* lalu ID dari tag yang didekatkan akan ditampilkan di *Serial Monitor*.

Tabel 3. 5. Konektivitas Pin *RFID* dan *ESP32*

RFID	**ESP32**
3.3V	3.3V
GND	GND

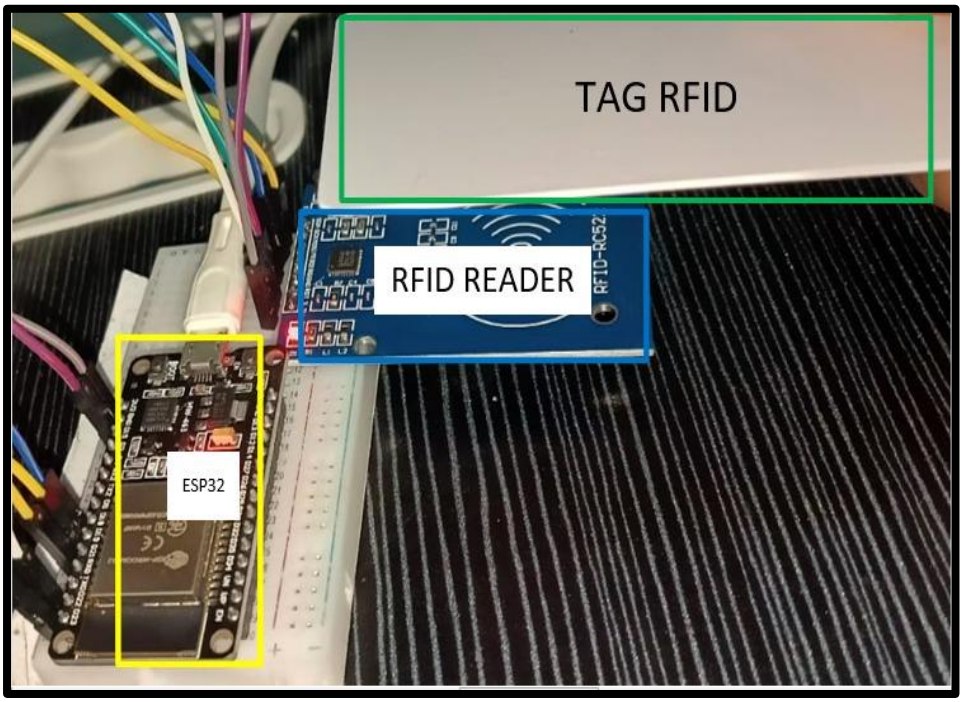
SS	21
SCK	18
MOSI	23
MISO	19
RST	22

Tabel 3. 5 menunjukan konfigurasi pin antara *RFID RC522* dengan *ESP32* yang digunakan untuk pegujian kapasitas penanganan *multiple* tag yang dapat dibaca oleh *reader*.

```
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN         22          // Pin RST pada modul RC522
#define SS_PIN          21          // Pin SDA (SS) pada modul RC522
MFRC522 mfrc522(SS_PIN, RST_PIN);   // Inisialisasi obyek MFRC522

void setup() {
  Serial.begin(115200);             // Inisialisasi Serial Monitor
  SPI.begin();                      // Inisialisasi SPI bus
  mfrc522.PCD_Init();               // Inisialisasi modul RC522
  Serial.println("Waiting for RFID card...");
}
void loop() {
  // Periksa apakah ada tag RFID yang terdeteksi
  if (mfrc522.PICC_IsNewCardPresent() && mfrc522.PICC_ReadCardSerial()) {
    // Baca ID tag RFID
    String rfidID = "";
    for (byte i = 0; i < mfrc522.uid.size; i++) {
      rfidID.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? "0" : ""));
      rfidID.concat(String(mfrc522.uid.uidByte[i], HEX));
    }
    Serial.print("RFID ID: ");
    Serial.println(rfidID);
    mfrc522.PICC_HaltA();           // Hentikan komunikasi dengan tag RFID
    mfrc522.PCD_StopCrypto1();      // Hentikan enkripsi pada tag RFID
  }
}
```

Gambar 3. 15. Program Pengujian Identifikasi *RFID*

Pada Gambar 3. 15 adalah program untuk mengidentifikasi tag *RFID* lalu tag *RFID* yang teridentifikasi akan ditampilkannya di *serial monitor*.

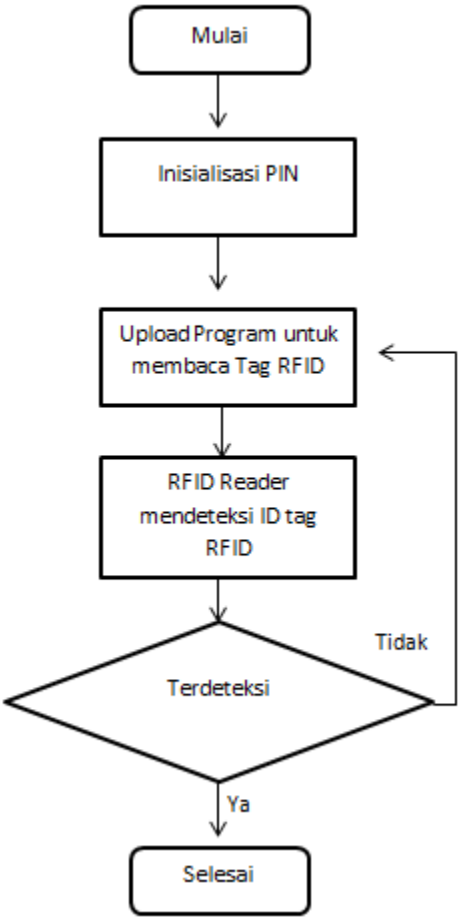


Gambar 3. 16. Pengujian Identifikasi pengujian *RFID*

Pada Gambar 3. 16 dilakukan Pengujian identifikasi ID Tag *RFID* dengan cara mendekatkan tag ke *reader*.

3.2.7.2 Kapasitas Penanganan

Menguji sejauh mana unit *RFID* dapat menangani jumlah tag *RFID* yang ada dalam jangkauannya secara bersamaan. Ini penting untuk memastikan bahwa sistem *RFID* tidak mengalami kegagalan atau penurunan kinerja saat ada banyak tag dalam area baca.



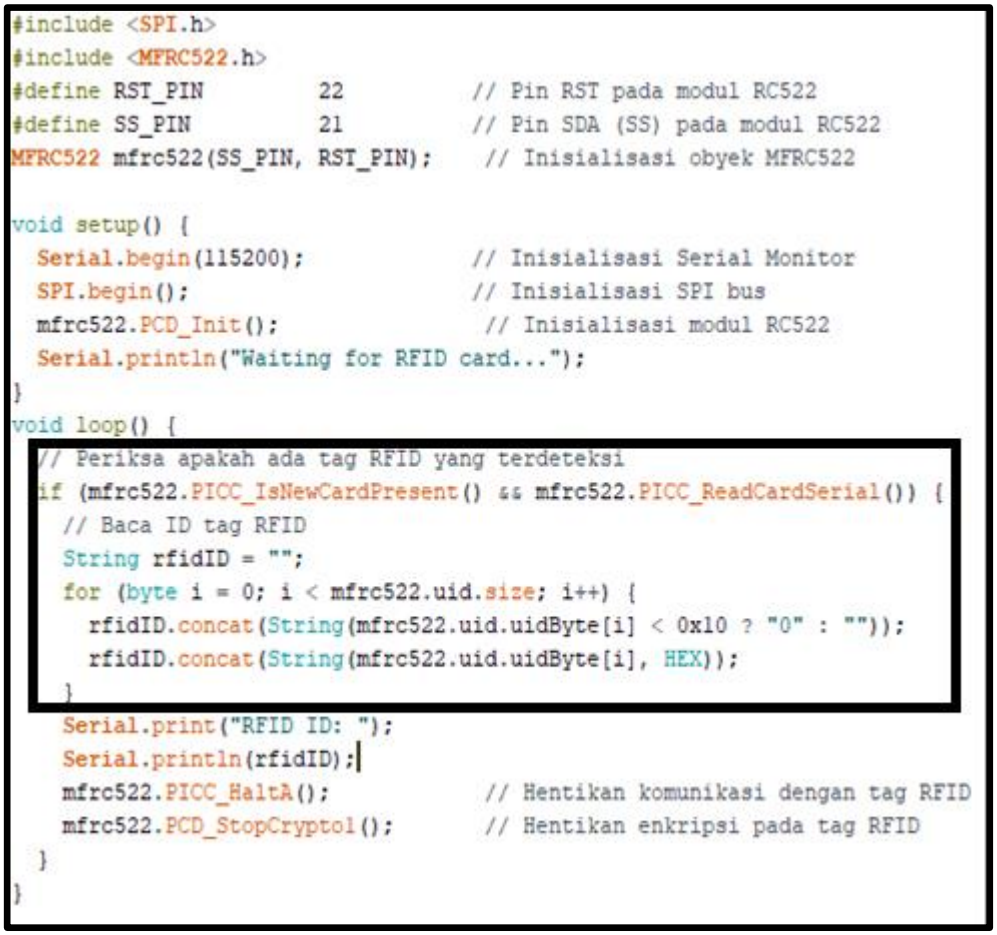
Gambar 3. 17. Flowchart pengujian kapasitas penangan tag *RFID*

Pada Gambar 3. 17 adalah *flowchart* yang digunakan untuk pengujian kapasitas penanganan *multiple* tag yang dapat dibaca oleh *reader*. Tag yang terbaca lalu akan ditampilkan di *serial monitor.*

Berdasarkan tujuan pengujian kapasistas penanganan *multiple* tag maka metode pengujian yang dilakukan yaitu menguji penanganan *multiple* tag secara bersamaan mulai dari penggunaan 2 tag, 3 tag, hingga *RFID* tidak dapat menangani banyaknya jumlah tag yang dideteksi secara bersamaan.

Gambar 3. 18. pengujian Kapasitas Penanganan *Multiple* Tag terhadap *RFID reader*

Gambar 3. 18 adalah rangkaian pengujian Kapasitas Penanganan *Multiple* Tag terhadap *RFID reader*. Tujuan pengujian ini untuk mengetahui berapa kapasitas jumlah tag yang dapat dideteksi dan di identifikasi dalamwaktu yang bersamaan.



```
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN         22          // Pin RST pada modul RC522
#define SS_PIN          21          // Pin SDA (SS) pada modul RC522
MFRC522 mfrc522(SS_PIN, RST_PIN);    // Inisialisasi obyek MFRC522

void setup() {
  Serial.begin(115200);             // Inisialisasi Serial Monitor
  SPI.begin();                      // Inisialisasi SPI bus
  mfrc522.PCD_Init();                // Inisialisasi modul RC522
  Serial.println("Waiting for RFID card...");
}
void loop() {
  // Periksa apakah ada tag RFID yang terdeteksi
  if (mfrc522.PICC_IsNewCardPresent() && mfrc522.PICC_ReadCardSerial()) {
    // Baca ID tag RFID
    String rfidID = "";
    for (byte i = 0; i < mfrc522.uid.size; i++) {
      rfidID.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? "0" : ""));
      rfidID.concat(String(mfrc522.uid.uidByte[i], HEX));
    }
    Serial.print("RFID ID: ");
    Serial.println(rfidID);
    mfrc522.PICC_HaltA();           // Hentikan komunikasi dengan tag RFID
    mfrc522.PCD_StopCrypto1();      // Hentikan enkripsi pada tag RFID
  }
}
```

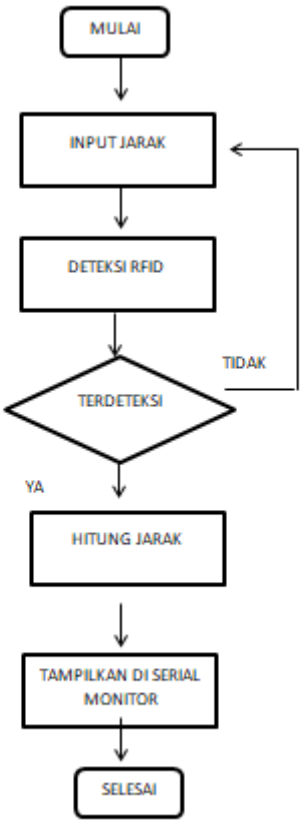
Gambar 3. 19. Program Pengujian Kapasitas penanganan *multiple* tag *RFID*

Pada Gambar 3. 19 adalah program untuk mengidentifikasi kapastitas penanganan tag *RFID* dalam waktu yang bersamaan.

3.2.7.3 Pengaruh Jarak Dalam Mendeteksi

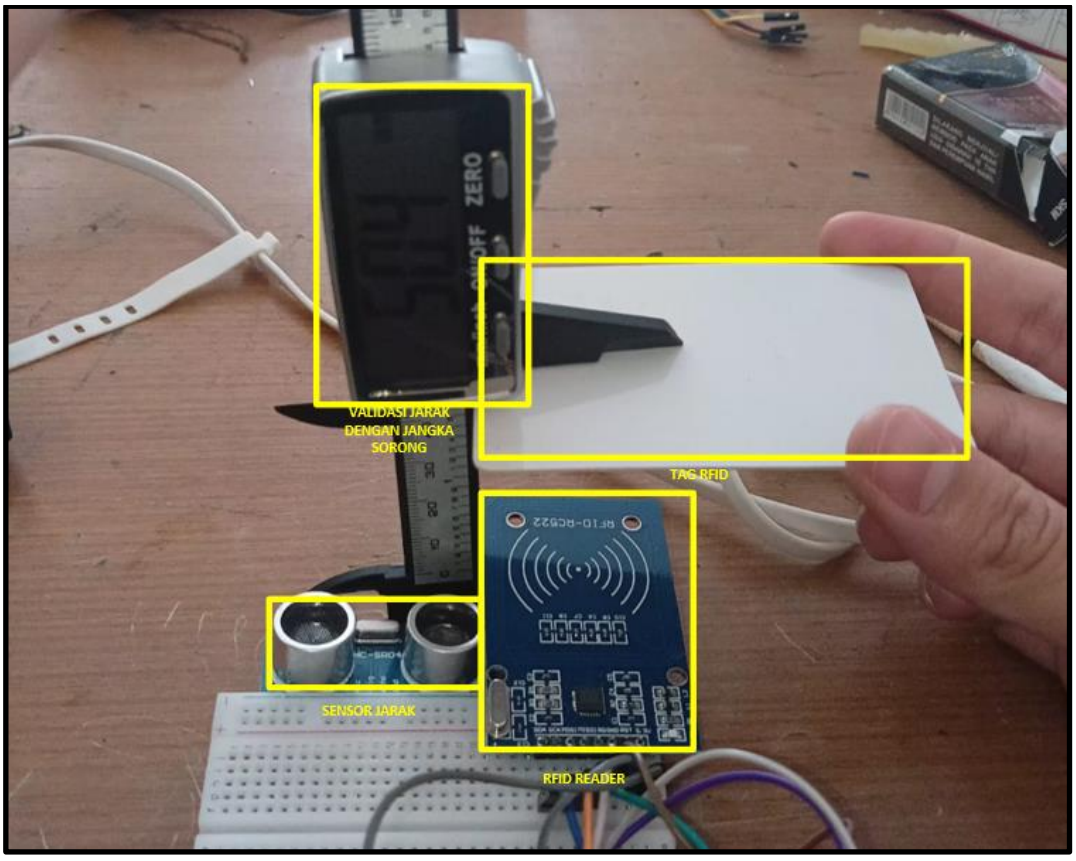
Menguji kemampuan *RFID* untuk mendeteksi dalam berbagai jarak. Ini bertujuan untuk mengetahui jarak jangkauan kemampuan *RFID* yang digunakan. Berdasarkan tujuan pengujian pengaruh jarak *RFID* dalam mendeteksi maka metode pengujian yang digunakan yaitu dilakukannya 2 kali pengujian , pengujian pertama untuk mengambil data jarak jangkauan *RFID* dengan bantuan sensor ultrasonik, setiap pengujian dilakukan 10 kali iterasi dengan cara mendekatkan tag

dari jarak 10cm mendekati *reader* hingga terdeteksi di *serial monitor*. Pengujian kedua dilakukan validasi data untuk pengujian pertama.



Gambar 3. 20. *Flowchart* Pengujian Pengrauh Jarak Terhadap Deteksi *RFID*

Pada Gambar 3. 20 adalah *Flowchart* Pengujian Pengrauh Jarak Terhadap Deteksi *RFID*. Dalam pengujian ini menggunakan komponen bantuan *sensor ultrasonik* yang telah di validasi menggunakan jangka sorong, cara pengujian ini adalah ketika tag *RFID* terdeteksi maka *sensor ultrasonik* mulai mengitung jarak antara tag *RFID* terhadap *RFID reader*. Jarak deteksi RFID yang terbaca adalah jarak terjauh dari kinerja *RFID* tersebut . berikut langkah pengujian nya yaitu letakan tag *RFID* secara mendekat dari jarak *outrange* nya. Pengujian dilaksanakan hingga tag *RFID* terdeteksi dan jaraknya akan ditampilkan di *serial monitor* lalu selesai.



Gambar 3. 21. Pengujian Pengaruh Jarak Terhadap Deteksi *RFID*

Gambar 3. 21 adalah rangkaian pengujian pengaruh jarak terhadap deteksi *RFID* menggunakan bantuan sensor *ultrasonik* untuk mengukur jarak yang telah di kalibrasi dan validasi oleh jangka sorong *digital*.

Tabel 3. 6. Konektivitas Pin Antara *RFID,ESP32*, dan *Sensor Ultrasonik*

RFID	**ESP32**
3.3V	3.3V
GND	GND
SS	21
SCK	18
MOSI	23
MISO	19

RST	22
SENSOR ULTRASONIK	
TRIG	12
ECHO	14
VCC	5V
GND	GND

Tabel 3. 6 menunjukan konfigurasi pin antara *ESP32, sensor ultrasonic,* dan *RFID.* Dalam rangkaian ini digunakan satu buah catu daya untuk menyuplai semua komponen yang digunakan .

```
#include <SPI.h>
#include <MFRC522.h>

#define SS_PIN 5
#define RST_PIN 25

#define TRIGGER_PIN 12
#define ECHO_PIN 14

MFRC522 mfrc522(SS_PIN, RST_PIN);

void setup() {
  Serial.begin(115200);
  SPI.begin();
  mfrc522.PCD_Init();
  pinMode(TRIGGER_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  Serial.println("Scan RFID Tag...");
}

void loop() {
  if (mfrc522.PICC_IsNewCardPresent() && mfrc522.PICC_ReadCardSerial()) {
    String tagID = "";
    for (byte i = 0; i < mfrc522.uid.size; i++) {
      tagID += String(mfrc522.uid.uidByte[i], HEX);
    }

    Serial.print("Tag ID: ");
```

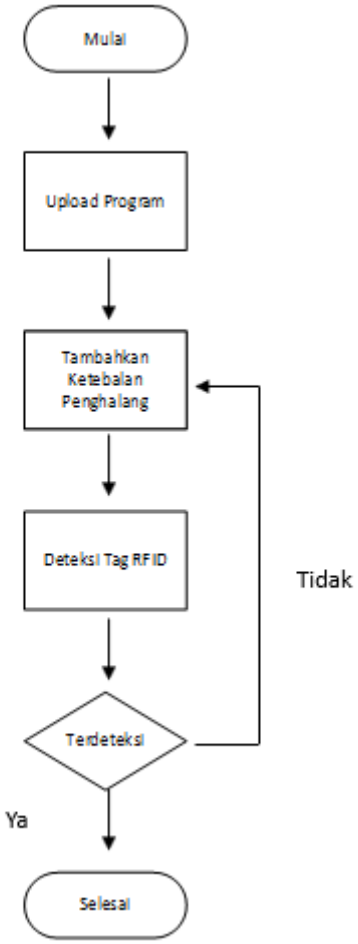
Gambar 3. 22. Pogram Pengujian Pengaruh Jarak Terhadap Deteksi *RFID* (1)

```
    Serial.println(tagID);
    Serial.print("Jarak: ");
    Serial.print(distance);
    Serial.println(" cm");

    mfrc522.PICC_HaltA();
    delay(1000);
  }
}

float calculateDistance() {
  digitalWrite(TRIGGER_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIGGER_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIGGER_PIN, LOW);

  long duration = pulseIn(ECHO_PIN, HIGH);

  // Calculate distance in centimeters
  float distance = duration * 0.034 / 2.0;

  return distance;
}
```

Gambar 3. 23. Pogram Pengujian Pengaruh Jarak Terhadap Deteksi *RFID* (2)

Pada Gambar 3. 22 dan Gambar 3. 23 adalah program untuk memerintahkan *RFID reader* membaca ID tag *RFID* yang terdeteksi oleh modul *MFRC522* dan menghitung jarak antara tag dan pembaca *RFID* waktu pertama kali terdeteksi. Hasil analisis, yaitu ID tag dan jarak dalam sentimeter akan ditampilkan melalui *serial monitor*.

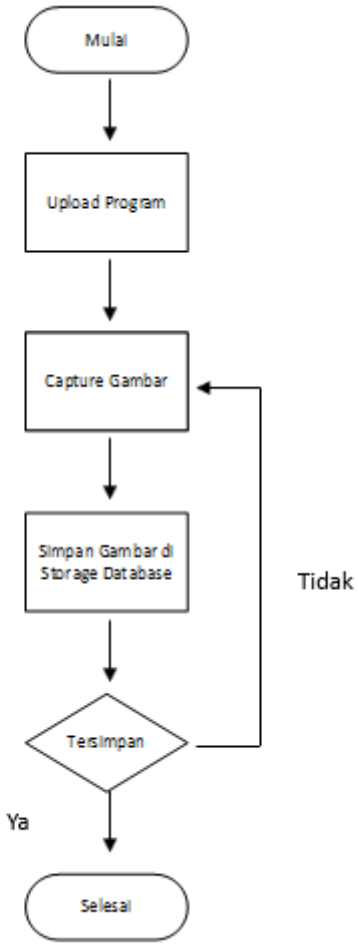
3.2.7.4 Pengaruh Penghalang Mika/Akrilik Dalam Mendeteksi



Gambar 3. 24. *Flowchart* pengujian pengaruh penghalang dalam mendeteksi

Berdasarkan *Flowchart* Gambar 3. 24 maka metode pengujian yang dilakukan yaitu menguji pendeteksian RFID menggunakan penghalang 1mm, 6mm, 7mm, dan 10mm. Tujuan dari pengujian pengaruh penghalang mika/akrilik dalam mendeteksi adalah sebagai refrensi jika alat yang diteliti akan menggunakan *cover* yang berbahan akrilik.

3.2.8 Pengujian Pengambilan Gambar



Gambar 3. 25. *Flowchart* pengujian pengambilan gambar

Berdasarkan *Flowchart* pengujian pengambilan gambar pada Gambar 3. 25 dilakukan pengujian dengan metode pengambilan gambar dan penyimpanan gambar di *storage database* lalu catat waktu yang dibutuhkan untuk proses keseluruhan mulai dari pengambilan gambar hingga penyimpanan gambar.

Gambar 3. 26. pengujian unit kamera menggunakan modul *ESP32-CAM*

Gambar 3. 26 adalah pengujian unit kamera menggunakan modul *ESP32-CAM* yang terhubung dengan *firebase storage* untuk tempat penyimpanan foto yang berhasil di ambil. Tujuan pengujian ini untuk mengetahui kinerja dari mdoul kamera yang digunakan.

```
void loop() {
  if (takeNewPhoto) {
    capturePhotoSaveSpiffs();
    takeNewPhoto = false;
  }
  delay(1);
  if (Firebase.ready() && !taskCompleted){
    taskCompleted = true;
    Serial.print("Uploading picture... ");

    //MIME type should be valid to avoid the download problem.
    //The file systems for flash and SD/SDMMC can be changed in FirebaseFS.h.
    if (Firebase.Storage.upload(&fbdo, STORAGE_BUCKET_ID /* Firebase Storage bucket id */, FILE_PHOTO /*
    }
    else{
      Serial.println(fbdo.errorReason());
    }
  }
}
```

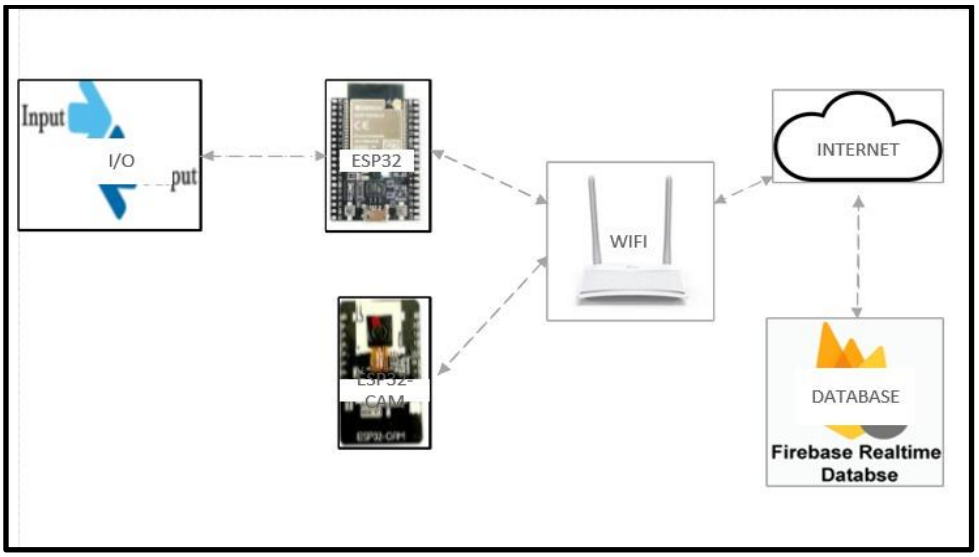
Gambar 3. 27. Program Pengujian *ESP32-CAM*

Gambar 3. 27 adalah algoritma program *ESP32-CAM* yang memberi perintah mengambil gambar lalu gambar tersebut di upload ke *firebase storage.*

3.2.9 Perakitan

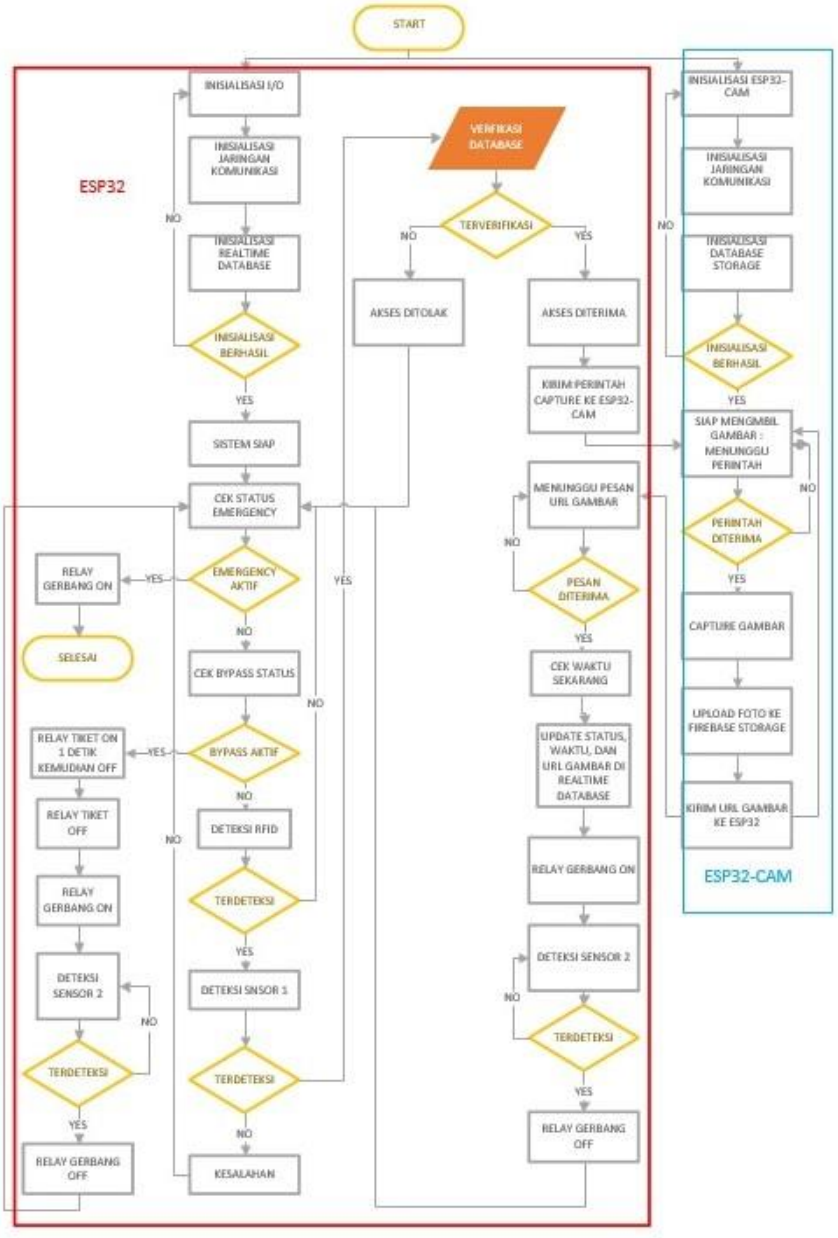
Pada tahap ini merakit perangkat keras sesuai dengan rancangan sistem. Ini melibatkan langkah-langkah berikut:

1. Memeriksa dan mempersiapkan semua komponen yang diperlukan dan komponen pendukung lainnya.
2. Mengupload program ke *ESP32*
3. Mengupload program ke *ESP32-Cam*
4. Mengikuti skema yang telah dirancang sebelumnya untuk menentukan posisi dan koneksi setiap komponen.
5. Memasang *ESP32* pada breadboard
6. Menghubungkan modul *RFID* dengan *ESP32* menggunakan kabel jumper dan memperhatikan penempatan pin yang benar.
7. Memasang sensor *IR Proximity.*
8. Memasang *Relay* yang akan digunakan, memastikan kabelnya terhubung dengan benar ke *ESP32*.
9. Memasang *MP3* dan Loads*peaker*
10. Memasang modul *RTC*
11. Konfigurasi *Database.*
12. Memeriksa kembali setiap koneksi dan memastikan bahwa semua komponen terpasang dengan aman dan sesuai dengan rancangan sistem.



Gambar 3. 28. Arsitektur Jaringan Komunikasi Data

Gambar 3. 28 adalah alur komunikasi pengiriman dan penerimaan data, dimulai dari *I/O* yaitu sensor-sensor yang terhubung ke *ESP32*, *ESP32-CAM*, *WiFi* , jaringan *internet* hingga *firebase*. Komunikasi antara I/O dan *ESP32* menggunakan komunikasi serial, dimana data dikirim melauli kabel. Sedangkan komunikasi antara *ESP32*, *ESP32-CAM*, dan *firebase* menggunakan protokol *esp.now* yang terhubung secara *wireless* di satu jaringan *internet* yang sama dengan pengalamatan menggunakan *mac adress* dari masing-masing perangkat.



Gambar 3. 29. *Flowchart* Sistem Akses Gerbang

Gambar 3. 29 adalah *flowchart* sistem akses gerbang yang mencakup sistem *emergency*, sistem *bypass*, dan sistem *RFID*. Pada kotak bagian berwarna merah menunjukan bahwa proses dilaksanakan oleh *ESP32* dan kotak bagian yang berwarna biru proses dilaksanakan oleh *ESP32-CAM* . *Flowchart / Algoritma* tersebut digunakan sebagai acuan pembuatan program.

3.2.10. Prosedur Penggunaan Alat

Sistem gerbang dapat di akses menggunakan 3 cara yaitu dengan sistem *RFID*, sistem *bypass*, dan sistem *emergency* . berikut adalah penjelasan cara mengakses menggunakan masing-masing sistem :

1. Sistem *RFID*

   Akses dengan menggunakan sistem *RFID* akan dimulai secara otomatis apabila sensor 1 mendeteksi objek yang berada di depan gerbang dan *RFID reader* mendeteksi tag dalam jangkauannya. Akses ini digunakan oleh mahasiswa , dosen, dan pegawai untuk masuk dan keluar .

2. Sistem *Bypass*

   Akses dengan menggunaan sistem *bypass* akan dimulai ketika *push button* yang berada di ruang control ditekan. Pengguna yang ingin mengakses dengan menggunakan sistem *bypass* meminta izin ke petugas yang berada di ruang kontrol gerbang masuk. Setelah petugas menekan tombol *bypass* maka tiket akan diterbitkan. Tiket tersebut digunakan sebagai syarat untuk akses gerbang keluar dengan menyerahkan tiket tersebut ke petugas di ruang kontrol gerbang keluar. Akses ini digunakan oleh tamu udangan, kurir, orang tua wali, dan sebagainya.

3. Sistem *Emergency*

   Akses dengan menggunakan sistem *emergency* akan di mulai ketika *push button* di ruang control ditekan, setelah sistem *emergency* aktif maka gerbang tidak akan pernah tertutup hingga sistem gerbang di *restart*. Tombol *emergency* terletak di ruang kontrol petugas. Akses ini digunakan

ketika dalam kondisi darurat seperti kebakaran , bencana alam, dan acara-acara tertentu.

3.2.11. Pengujian Sistem

Pada tahap ini dilaksanakan pengujian sistem gerbang masuk dan keluar menggunakan *RFID*, *bypass*, dan *emergency* lalu pengujian kinerja sistem dengan menggunakan kamera dan tidak menggunakan kamera.

3.2.12. Analisis Sistem Kerja

Analisis sistem kerja terdiri dari proses pengumpulan data dari hasil pengujian sehingga dapat diketahui kelebihan dan kekurangan dari sistem tersebut.

3.2.13. Kesimpulan

Setelah data diambil kemudian dilakukan analisa terhadap hasil pengujian, maka selanjutnya dapat menentukan kesimpulan dari rumusan dan tujuan penelitian.berdasarkan hasil pengujian.