BAB 3
METODE PENELITIAN

Pada BAB 3 metode penelitian ini membahas tentang alur penelitian, perancangan prototipe, dan pengujian pada brankas yang menggunakan sensor RFID, *One Time Password* (OTP) dan Bot Telegram. Sistem analog dipadukan dengan sistem digital sehingga perancangan prototipe dibagi menjadi 2 bagian yaitu perancangan *hardware*, dan perancangan *software*. Perancangan *hardware* meliputi sebuah sensor, *microcontroller*, dan sumber tegangan. Perancangan *software* meliputi pengkodean program dan logika brankas.

3.1 ALAT YANG DIGUNAKAN

3.1.1 Perangkat Keras (*Hardware*)

Dalam penelitian ini digunakan perangkat keras sebagai berikut:

Tabel 3.1 Perangkat keras yang digunakan

NO.	Alat	Jumlah
1	Laptop	1
2	*Smartphone*	1
3	ESP8266 NodeMCU	1
4	*Base Plate Board* NodeMCU	1
4	RFID	1
5	RFID Tag	20
6	Keypad 4X4	1
7	LED	1
8	Kabel Jumper	~
9	*Breadboard*	1
10	*Solenoid lock* 12v	1
11	Catu Daya 12v	1

3.1.2 Perangkat Lunak (*Software*)

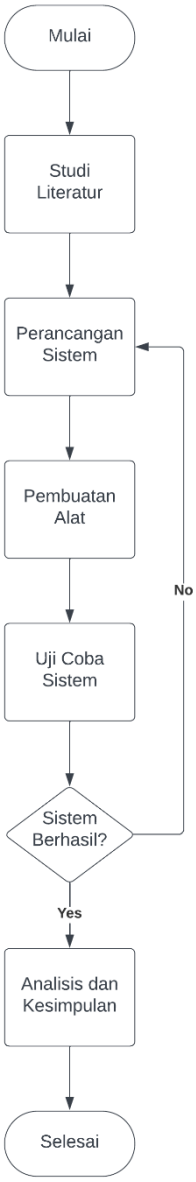
Perangkat lunak yang digunakan dalam penelitian ini adalah:

1. Arduino IDE
2. Bot Telegram

3.2 ALUR PENELITIAN

Penelitian ini melalui serangkaian alur penelitian yang terstruktur, dimulai dengan studi literatur yang mendalam untuk memahami landasan teori dan penelitian terdahulu yang berkaitan dengan topik penelitian. Studi literatur ini mencakup penelusuran jurnal, buku, artikel, dan sumber-sumber terpercaya lainnya yang memberikan wawasan tentang teknologi yang digunakan, seperti ESP8266 NodeMCU, RFID, *One Time Password* (OTP), dan bot Telegram. Proses ini juga melibatkan analisis penelitian-penelitian sebelumnya untuk mengidentifikasi metode, pendekatan, dan hasil yang relevan, serta mengevaluasi kekuatan dan kelemahan dari pendekatan-pendekatan tersebut. Dengan memahami konteks dan perkembangan terkini dalam bidang ini, peneliti dapat mengidentifikasi celah pengetahuan yang ada dan menentukan arah penelitian yang akan memberikan kontribusi baru dan signifikan.

Tahap berikutnya adalah perancangan sistem, di mana konsep dan spesifikasi alat dirumuskan dengan cermat berdasarkan temuan dari studi literatur dan kebutuhan penelitian. Perancangan ini melibatkan pemilihan komponen *hardware* yang tepat, seperti sensor RFID, keypad, dan modul relay, serta perangkat lunak yang diperlukan untuk mengendalikan dan mengintegrasikan semua komponen tersebut. Dalam tahap ini, dibuat pula diagram alir dan sketsa desain untuk memetakan fungsi dan alur kerja sistem secara keseluruhan. Konsep perancangan yang matang memastikan bahwa setiap komponen berfungsi dengan baik dan dapat saling berintegrasi untuk mencapai tujuan sistem keamanan brankas yang diinginkan. Setelah sistem teruji, dilakukan pengambilan data yang relevan yang akan dianalisis untuk menarik kesimpulan yang akurat. Data ini dianalisis untuk mengevaluasi keberhasilan sistem dalam memenuhi tujuan penelitian. Keseluruhan alur penelitian ini disusun secara sistematis untuk memastikan kelancaran dan validitas hasil penelitian yang diperoleh.



Gambar 3.1 *Flowchart* alur penelitian

3.2.1 Studi Literatur

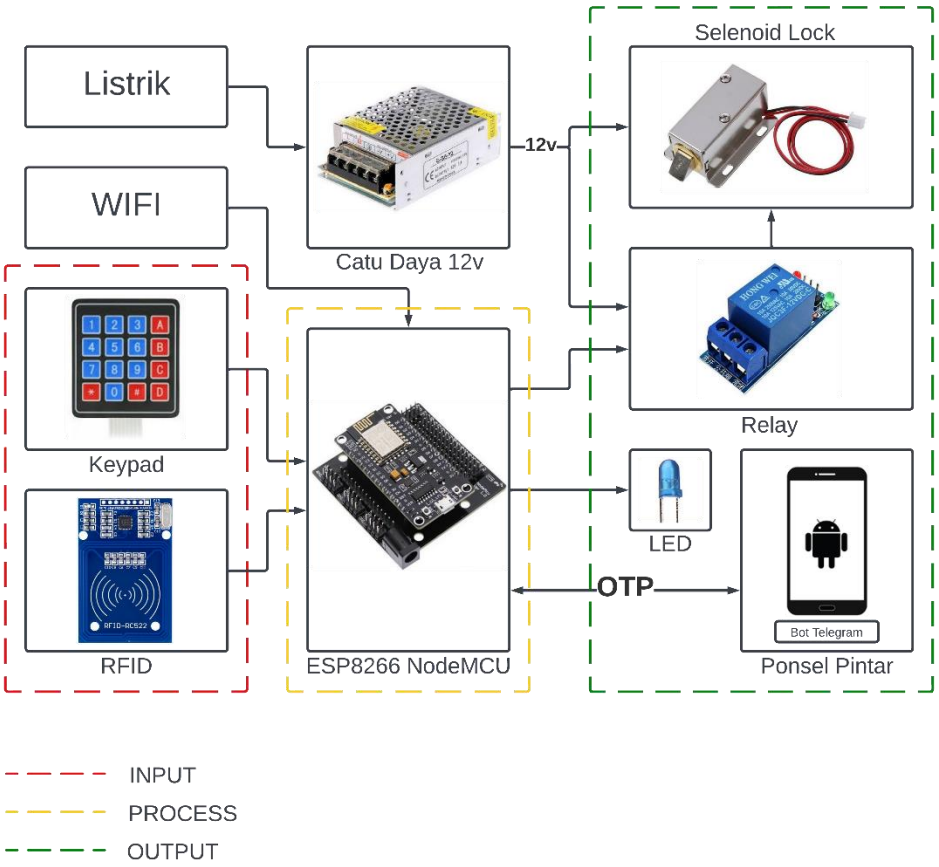
Studi literatur menjadi tahap awal dalam penelitian ini, di mana dilakukan eksplorasi mendalam terhadap berbagai referensi dan penelitian terkait untuk memperoleh pemahaman yang komperhensif. Melalui studi literatur, diperoleh pemahaman tentang teori-teori yang relevan serta temuan-temuan penting dari penelitian terdahulu yang dapat menjadi landasan bagi perancangan sistem dan penelitian lebih lanjut. Tahap ini mengarahkan penelitian ke arah yang tepat dan memastikan keberhasilannya melalui penguatan dasar ilmiah yang kuat.

3.2.2 Perancangan Sistem

Perancangan sistem merupakan tahapan penting dalam pengembangan sebuah alat atau sistem. Tahap ini melibatkan dua aspek utama yang saling terkait, yaitu perancangan perangkat fisik dan perancangan perangkat lunak. Perancangan perangkat fisik mencakup pemilihan komponen-komponen fisik yang sesuai dengan kebutuhan penelitian seperti yang sudah disebutkan pada alat dan bahan. Di sisi lain, perancangan perangkat lunak adalah pembuatan kode program yang akan mengendalikan operasi sistem secara keseluruhan. Dalam perancangan ini, ditetapkan algoritma-algoritma, logika kontrol, dan interaksi antar modul sistem untuk memastikan fungsi sistem sesuai dengan tujuan penelitian. Perancangan sistem ini melibatkan tiga tahapan utama, yaitu *input*, proses, dan *output*. *Input* sistem ini terdiri dari RFID dan *One Time Password* (OTP) melalui keypad, yang kemudian diproses oleh ESP8266 NodeMCU. *Output* dari sistem mencakup bot Telegram yang menampilkan OTP dan notifikasi, serta relay yang mengontrol kunci *solenoid lock* untuk membuka brankas.

3.2.2.1 Perancangan Perangkat Fisik

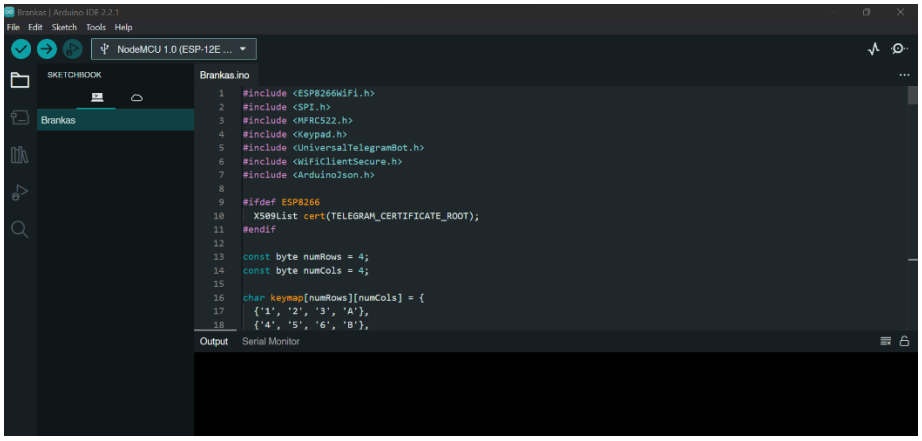
Dalam proses pembuatan prototipe keamanan brankas digunakan beberapa perangkat fisik yang memiliki peran penting dalam operasional sistem. Perangkat-perangkat tersebut antara lain ESP8266 NodeMCU sebagai basis kontrol utama, modul RFID untuk deteksi akses, RFID Tag untuk akses, Keypad 4X4 untuk input kode OTP, LED sebagai indikator, kabel jumper untuk menghubungkan komponen-komponen, breadboard sebagai tempat penyusunan sementara, *solenoid lock* 12V sebagai mekanisme pengunci brankas, relay untuk mengendalikan daya yang diberikan pada *solenoid lock*, dan catu daya 12V untuk memberikan daya pada *solenoid lock*. Integrasi berbagai perangkat fisik ini memungkinkan prototipe brankas memiliki fungsi keamanan yang efektif. Berikut ini adalah Blok diagram brankas secara keseluruhan yang terintegrasi mulai dari *Input, Process* dan yang terakhir adalah *Output*:



Gambar 3.2 Blok diagram brankas

3.2.2.1 Perancangan Perangkat Lunak

Perancangan perangkat lunak memegang peranan penting dalam menentukan kinerja dan fungsionalitas sistem secara keseluruhan. Dalam konteks pengembangan sistem keamanan brankas, perangkat lunak bertujuan untuk mengatur alur kerja sistem secara efisien dan efektif. Dalam pelaksanaannya, perangkat lunak dikembangkan menggunakan lingkungan pengembangan Arduino IDE, yang menyediakan platform yang dapat diprogram untuk mengontrol perangkat keras mikrokontroler. Arduino IDE berfungsi untuk melakukan pengkodean dan kompilasi program-program yang diperlukan untuk mengontrol berbagai aspek dalam sistem keamanan brankas, termasuk interaksi dengan komponen perangkat keras seperti RFID, keypad, LED, dan relay. Proses penulisan kode melibatkan pemrograman dalam bahasa pemrograman C/C++, yang memungkinkan untuk mengatur logika operasional sistem keamanan brankas.



Gambar 3.3 Arduino IDE

Perangkat lunak juga melibatkan integrasi dengan UniversalTelegramBot Library, yang memungkinkan sistem untuk berkomunikasi dengan bot Telegram dan mengirimkan OTP dan notifikasi ke pengguna secara real-time. Bot Telegram berperan sebagai antarmuka untuk pengguna, menyediakan layanan untuk menerima OTP serta notifikasi terkait keamanan brankas. Selain itu, kode yang dibangun dalam perangkat lunak bertanggung jawab untuk mengelola *input* dari berbagai sumber, seperti pemindaian RFID dan input *One Time Password* (OTP), serta memicu *output* yang sesuai berupa OTP dan notifikasi kepada pengguna melalui bot Telegram dan mengendalikan kunci *solenoid lock* melalui relay. Berikut ini adalah penjelasan kode yang yang digunakan dalam penelitian ini.

```
#include <ESP8266WiFi.h>
#include <SPI.h>
#include <MFRC522.h>
#include <Keypad.h>
#include <UniversalTelegramBot.h>
#include <WiFiClientSecure.h>
#include <ArduinoJson.h>
```

Kode tersebut merupakan *library* yang diimpor untuk mendukung berbagai fungsi yang diperlukan dalam proyek ini. *Library* <*ESP8266WiFi.h*> digunakan untuk menghubungkan modul ESP8266 ke jaringan WiFi, <*SPI.h*> untuk komunikasi dengan modul RFID melalui *serial peripheral interface* (SPI), <*MFRC522.h*> untuk berinteraksi dengan modul RFID MFRC522, <*Keypad.h*> untuk mengelola input dari keypad, <*UniversalTelegramBot.h*> dan

<*WiFiClientSecure.h*> untuk berkomunikasi dengan bot Telegram secara aman, serta <*ArduinoJson.h*> untuk memproses data JSON.

```
const byte numRows = 4;
const byte numCols = 4;

char keymap[numRows][numCols] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};

byte rowPins[numRows] = {D1, D2, D3, D4};
byte colPins[numCols] = {D5, D6, D7, D8};

Keypad keypad =
Keypad(makeKeymap(keymap), rowPins,
colPins, numRows, numCols);
```

Kode menginisialisasi keypad yang digunakan dalam proyek dengan mendefinisikan matriks tombol 4x4 yang terdiri dari angka dan huruf, serta menghubungkan baris dan kolom keypad ke pin digital ESP8266. Hal ini memungkinkan sistem untuk membaca input dari pengguna saat mereka memasukkan OTP melalui keypad.

```
const byte RST_PIN = D3;
const byte SS_PIN = D4;

MFRC522 mfrc522(SS_PIN, RST_PIN);
```

Pada bagian ini, modul RFID MFRC522 diinisialisasi dengan menentukan pin reset *(RST_PIN)* dan *slave select (SS_PIN)* yang terhubung ke ESP8266. Inisialisasi ini memungkinkan modul untuk membaca dan memproses data dari kartu RFID yang digunakan untuk otentikasi.

```
const int internalLedPin = 2;

const int ledBrankas = D1;

char generatedOTP[7] = "";
char inputOTP[7] = "";
int inputIndex = 0;
```

```
const char* ssid = "BrankasWifi";
const char* password = "123456778";

#define BotToken
"7040455604:AAGcmr5WiFJsrGr__nc4NP0H
TPsj-XuCRRM"
#define CHAT_ID "1837546129"

const int relayPin = D0;

WiFiClientSecure client;
UniversalTelegramBot bot(BotToken, client);

unsigned long lastOTPTimestamp = 0;
const unsigned long otpInterval = 60000;

enum OperatingMode {
  RFID_MODE,
  OTP_MODE,
  LOCKED_MODE
};

OperatingMode currentMode = RFID_MODE;
```

Pin dan variabel didefinisikan untuk mengontrol komponen seperti LED internal, LED brankas, relay, serta menyimpan informasi jaringan WiFi dan token bot Telegram. Selain itu, variabel untuk menyimpan OTP yang dihasilkan dan diinput oleh pengguna juga diinisialisasi, serta mode operasi awal ditetapkan ke *RFID_MODE*.

```
void setup() {
  Serial.begin(9600);

  pinMode(ledBrankas, OUTPUT);

  #ifdef ESP8266
    configTime(0, 0, "pool.ntp.org");
    client.setTrustAnchors(&cert);
  #endif

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
```

```
  Serial.println("Connected to WiFi");

  pinMode(relayPin, OUTPUT);

 digitalWrite(relayPin, LOW);

  SPI.begin();
  mfrc522.PCD_Init();

  bot.sendMessage(CHAT_ID, "SCAN RFID
ANDA", "");
}
```

Fungsi *setup()* mengatur konfigurasi awal sistem termasuk menginisialisasi komunikasi serial, menghubungkan ESP8266 ke jaringan WiFi, mengatur pin untuk relay dan LED, serta menginisialisasi modul RFID dan mengirim pesan awal melalui bot Telegram untuk menginstruksikan pengguna untuk memindai kartu RFID.

```
void loop() {
  if (currentMode == RFID_MODE) {
    if (mfrc522.PICC_IsNewCardPresent()) {
      if (mfrc522.PICC_ReadCardSerial()) {
        for(int i = 0; i < 5; i++) {
          digitalWrite(ledBrankas, HIGH);
          delay(100);
          digitalWrite(ledBrankas, LOW);
          delay(100);
        }
        if (isAuthorizedRFID()) {
          // RFID terotorisasi
          generateNewOTP();
          currentMode = OTP_MODE;
        }
      }
    } else {
      digitalWrite(ledBrankas, LOW);
    }
  } else if (currentMode == OTP_MODE) {
    char key = keypad.getKey();

    analogWrite(ledBrankas, 100); // Misalnya,
nilai kecerahan 50 (dari 0 hingga 255)

    if (key == '#') {
      if (inputIndex == 6) {
        checkOTP();
      }
```

```
    } else if (isdigit(key) && inputIndex < 6) {
      inputOTP[inputIndex] = key;
      inputIndex++;
      Serial.print(key);
    }
  } else if (currentMode == LOCKED_MODE) {
    digitalWrite(ledBrankas, LOW);

    delay(5000);

    SPI.begin();
    mfrc522.PCD_Init();

    currentMode = RFID_MODE;
  }
}
```

Fungsi *loop()* mengontrol alur kerja utama sistem berdasarkan mode operasi saat ini. Dalam *RFID_MODE*, sistem mendeteksi kartu RFID dan memvalidasi otorisasi. Jika otorisasi berhasil, sistem beralih ke *OTP_MODE*, di mana pengguna harus memasukkan OTP yang dihasilkan. Setelah OTP valid atau tidak valid, sistem beralih ke *LOCKED_MODE*, memastikan brankas tetap terkunci dan siap untuk siklus berikutnya.

```
bool isAuthorizedRFID() {
  byte authorizedUID[][4] = {
    {0xA3, 0xDB, 0x7C, 0xF5}, {0x3F, 0x2F,
0x63, 0x26}, {0x2F, 0xF4, 0x06, 0x26}, {0x1F,
0xD2, 0x8A, 0x26}, {0x3F, 0x25, 0x4F, 0x26},
{0x3F, 0x2D, 0xA6, 0x26}, {0xFE, 0xA4, 0x39,
0x1D}, {0x3F, 0x03, 0xA7, 0x26}, {0x3F, 0x71,
0x1B, 0x26}, {0x3F, 0x68, 0x8B, 0x26}, {0x3F,
0x70, 0x6B, 0x26}
  };

  for (byte j = 0; j < sizeof(authorizedUID) /
sizeof(authorizedUID[0]); j++) {
    bool isUIDMatch = true;
    for (byte i = 0; i < mfrc522.uid.size; i++) {
      if (mfrc522.uid.uidByte[i] !=
authorizedUID[j][i]) {
        isUIDMatch = false;
        break;
      }
    }
    if (isUIDMatch) {;
      return true;
    }
```

```
 }
  bot.sendMessage(CHAT_ID, "AKSES
DITOLAK", "");
  delay(5000);
  return false;
}
```

Fungsi *isAuthorizedRFID()* memeriksa apakah UID kartu RFID yang dibaca cocok dengan daftar UID yang telah diotorisasi. Jika cocok, fungsi ini mengembalikan nilai *true*, menunjukkan bahwa kartu RFID sah untuk mengakses brankas. Jika tidak, fungsi ini mengirim pesan akses ditolak melalui bot Telegram dan mengembalikan nilai *false*.

```
void generateNewOTP() {
  randomSeed(millis());
  int newOTP = random(100000, 999999);
  snprintf(generatedOTP,
sizeof(generatedOTP), "%06d", newOTP);

  String message = "RFID
BERHASIL!!!\n\nOTP Anda: " +
String(generatedOTP);
  bot.sendMessage(CHAT_ID, message, "");
}
```

Fungsi *generateNewOTP()* menghasilkan OTP baru secara acak setiap kali kartu RFID yang diotorisasi terdeteksi. OTP yang dihasilkan dikirim ke pengguna melalui bot Telegram, memastikan bahwa hanya pengguna yang memiliki akses ke bot Telegram yang dapat menerima OTP untuk membuka brankas.

```
void checkOTP() {
  if (strcmp(inputOTP, generatedOTP) == 0) {

    for(int i = 0; i < 5; i++) {
      digitalWrite(ledBrankas, HIGH);
      delay(100);
      digitalWrite(ledBrankas, LOW);
      delay(100);
    }

    bot.sendMessage(CHAT_ID, "OTP
BENAR!!!\n\nBRANKAS TERBUKA!!!", "");
```

```
    digitalWrite(relayPin, HIGH);
    digitalWrite(ledBrankas, HIGH);
    delay(10000);

    bot.sendMessage(CHAT_ID, "BRANKAS
TERKUNCI!!!", "");

    digitalWrite(relayPin, LOW);

    resetInput();
    currentMode = LOCKED_MODE;
  } else {
      for(int i = 0; i < 5; i++) {
        digitalWrite(ledBrankas, HIGH);
        delay(100);
        digitalWrite(ledBrankas, LOW);
        delay(100);
      }
    bot.sendMessage(CHAT_ID, "OTP
SALAH!!!\n\nSilahkan scan RFID untuk
membuat OTP baru.", "");

  (default: terkunci)
    digitalWrite(relayPin, LOW);

    resetInput();
    currentMode = LOCKED_MODE;
  }
```

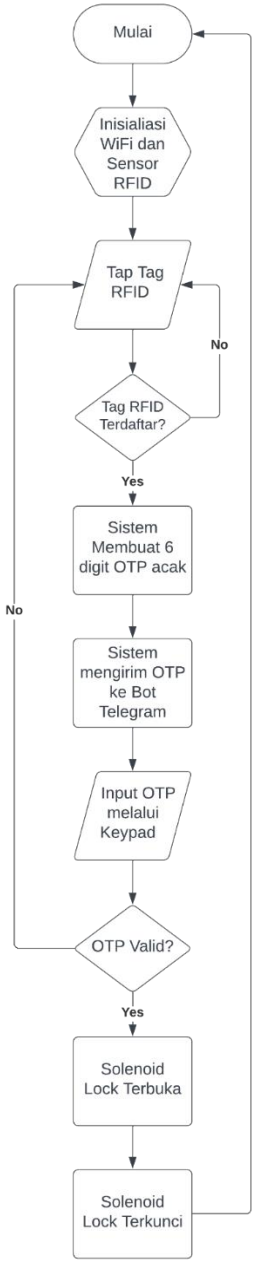
Fungsi *checkOTP()* membandingkan OTP yang diinput oleh pengguna dengan OTP yang dihasilkan. Jika OTP cocok, brankas dibuka dan pesan konfirmasi dikirim melalui bot Telegram, kemudian brankas dikunci kembali setelah beberapa saat. Jika OTP tidak cocok, sistem mengirim pesan kesalahan dan kembali ke mode terkunci, meminta pengguna untuk memindai kartu RFID lagi untuk menghasilkan OTP baru.

```
void resetInput() {
  memset(inputOTP, 0, sizeof(inputOTP));
  inputIndex = 0;
}
```

Fungsi resetInput() mereset input OTP yang dimasukkan oleh pengguna, membersihkan buffer input dan mengatur ulang indeks input, memastikan sistem siap untuk menerima OTP baru pada siklus berikutnya.

3.2.3 Simulasi Alat

Setelah merancang sistem, dilanjutkan melakukan simulasi alat menggunakan Arduino IDE untuk menguji fungsi-fungsi dasar. Simulasi mencakup verifikasi RFID, pembuatan OTP, pengambilan input dari keypad, pengiriman OTP ke Bot Telegram, serta pengoperasian *solenoid lock*. Simulasi ini menjadi langkah krusial sebelum implementasi ke perangkat fisik untuk memastikan keandalan dan keefektifan algoritma yang diusung oleh sistem berjalan dengan benar.



Gambar 3.4 Flowchart simulasi brankas

Simulasi sistem keamanan brankas terintegrasi dengan perangkat keras yaitu ESP8266 NodeMCU yang berfungsi sebagai modul WiFi yang mengatur koneksi ke Bot Telegram serta mengendalikan perangkat lainnya. Penggunaan RFID sebagai metode identifikasi pemilik brankas memberikan dasar untuk melanjutkan proses pembuatan OTP oleh sistem. Keypad digunakan sebagai input untuk memasukkan kode OTP, sementara *solenoid lock* bertugas mengendalikan mekanisme kunci brankas secara elektronik. OTP yang dihasilkan oleh sistem melalui Arduino IDE berfungsi sebagai lapisan kedua verifikasi keamanan, dengan Bot Telegram sebagai media penerima OTP yang akan diteruskan kepada pengguna.

Simulasi alat dimulai dengan inisialisasi koneksi WiFi dan sensor RFID sesuai langkah-langkah dalam *flowchart*. Pengguna diminta mendekatkan tag RFID ke sensor, dan jika tag yang terdeteksi tidak terdaftar, proses diulang hingga tag yang valid terdeteksi. Sistem memberikan notifikasi ke Bot Telegram jika RFID salah dan meminta pengguna untuk mengulang dari proses pemindaian tag RFID. Setelah tag yang terdeteksi valid dengan yang terdaftar pada sistem, secara otomatis sistem akan menghasilkan OTP acak berupa enam digit angka. Pengguna harus memasukkan OTP tersebut melalui keypad, dan jika OTP salah, notifikasi akan dikirim ke Bot Telegram dan pengguna diminta untuk mengulang proses buka brankas dari pemindaian RFID. Namun, jika OTP benar, sistem akan mengirimkan notifikasi ke Bot Telegram dan mengaktifkan relay untuk membuka kunci *solenoid lock*, sehingga brankas dapat terbuka sesuai dengan tujuan penelitian ini.

3.2.4 Pengujian alat

Setelah simulasi sistem, dilanjutkan dengan pengujian alat untuk memastikan kinerja sistem apakah sesuai dengan tujuan penelitian ini. Tahap pengujian ini merupakan langkah penting dalam memvalidasi keandalan dan kestabilan alat. Pada tahap pengujian, dilakukan serangkaian uji coba sebanyak 10 kali untuk masing-masing komponen utama, termasuk RFID, OTP, *Delay*, dan *Solenoid lock*. Hasil dari pengujian ini akan memberikan informasi yang diperlukan untuk memperbaiki atau mengoptimalkan performa alat sebelum digunakan secara penuh.

3.2.3.1 Pengujian RFID

Pengujian RFID melibatkan serangkaian langkah untuk memastikan bahwa sistem dapat secara efektif mengenali tag ID dari pemilik yang terdaftar, serta memblokir tag acak atau tidak dikenal agar tidak diizinkan untuk melanjutkan proses verifikasi. Langkah-langkah pengujian ini mencakup validasi kemampuan RFID dalam membaca tag ID yang terdaftar dalam basis data, pengujian respons sistem terhadap tag yang tidak dikenal atau tidak sah, serta evaluasi terhadap tingkat akurasi dan keandalan sistem dalam mengidentifikasi pemilik yang sah.

3.2.3.2 Pengujian *One Time Password* (OTP)

Pengujian OTP melibatkan serangkaian prosedur untuk memverifikasi bahwa setiap kode OTP yang dihasilkan terdiri dari 6 angka acak dan tidak menghasilkan OTP yang sama selama pengujian alat berlangsung. Langkah-langkah pengujian ini mencakup pengecekan keberagaman serta keunikan setiap kode OTP yang dihasilkan selama periode pengujian. Melalui pengujian ini, keamanan sistem OTP dapat dijamin dengan memastikan bahwa setiap kode yang dihasilkan tidak dapat diprediksi dan dapat diandalkan untuk mengamankan akses ke brankas.

3.2.3.3 Pengujian *Delay* Pengiriman OTP

Pengujian *Quality of Service* (QoS) pada sistem ini difokuskan pada evaluasi *delay* pengiriman OTP dari sistem ke Bot Telegram. *Delay* pengiriman ini menjadi indikator penting dalam mengevaluasi sistem dalam menyampaikan OTP kepada pengguna melalui Bot Telegram. Pengujian dilakukan dengan mengirim sejumlah OTP dan mencatat waktu yang diperlukan untuk setiap pengiriman. Hasilnya akan memberikan gambaran mengenai performa sistem dalam pengiriman OTP dan memastikan pengguna menerima OTP dengan tepat waktu. Melalui pengujian ini, diharapkan dapat diperoleh data akurat mengenai konsistensi dan keandalan sistem dalam berbagai kondisi jaringan. Dengan demikian, langkah-langkah perbaikan dapat dirancang untuk meminimalkan *delay* dan meningkatkan pengalaman pengguna secara keseluruhan.

3.2.3.4 Pengujian *Solenoid lock*

Pengujian *Solenoid lock* melibatkan serangkaian langkah untuk memverifikasi fungsi dan keandalan perangkat. Langkah-langkah pengujian ini mencakup pengujian terhadap respons *Solenoid lock* terhadap input OTP yang benar, untuk memastikan apakah *Solenoid lock* akan membuka pintu brankas setelah OTP yang benar dimasukkan. Selain itu, dilakukan juga pengujian terhadap respons *Solenoid lock* terhadap input angka acak, untuk memastikan bahwa perangkat tidak akan membuka pintu brankas ketika input tidak valid dimasukkan. Melalui pengujian ini, kehandalan dan keamanan *Solenoid lock* dapat dipastikan, sehingga memberikan perlindungan yang efektif terhadap akses yang tidak sah terhadap brankas.