LAMPIRAN A
PROGRAM

A. PROGRAM SISTEM KEAMANAN

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
#include <EEPROM.h>
#include <Wire.h>
#include <RtcDS3231.h>
#include <LiquidCrystal_I2C.h>

RtcDS3231<TwoWire> Rtc(Wire);
LiquidCrystal_I2C lcd(0x27, 16, 2);
SoftwareSerial mySerial(5, 4);
SoftwareSerial ss(6, 7);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);

char daysOfTheWeek[7][12] = {"minggu", "senin", "selasa", "rabu",
"kamis", "jumat", "sabtu"};
int detik, menit, jam, tgl, bln, thn;
int tombol1 = 13;
int relay1 = 11;
int eadd = 0;
int getFingerprintIDez();
uint8_t getFingerprintEnroll(int id);

String last_h, lastTime;
unsigned long timeBack = 0;
String dataSS;
int dataControl;
int dataRelay;
unsigned long timeShow;
byte flagRun;
int logicVoice = 0;

void setup() {
  Serial.begin(115200);
  lcd.begin();
  ss.begin(9600);
  for (int i = 0 ; i < EEPROM.length() ; i++) {
    EEPROM.write(i, 0);
    Serial.println(i);
  }
  pinMode(relay1, OUTPUT);
  pinMode(tombol1, INPUT_PULLUP);
  digitalWrite(relay1, HIGH);
  pinMode(8, INPUT);
  finger.begin(57600);

  delay(10);
  eadd = EEPROM.read(0);
  if (eadd > 200) {
    EEPROM.write(0, 0);
  }
  if (finger.verifyPassword()) {
    Serial.println("sensor terdetect");
  }
  else {
    Serial.println("sensor gak detect");
//    while (1);
  }
  eadd = EEPROM.read(0);

  rtcSet();
  //  voiceSet();
}

void loop() {
  RtcDateTime now = Rtc.GetDateTime();
  printDateTime(now);
  jam = now.Hour();//jam saat ini
  menit = now.Minute();//menit saat ini
  detik = now.Second();//detik saat ini
  tgl = now.Day();
```

```
bln = now.Month();
thn = now.Year();
Serial.println("run");

//manual dari blynk

if ((millis() - timeBack) > 100) {
  ss.print("{");
  ss.print(jam);
  ss.print(",");
  ss.print(menit);
  ss.print(",");
  ss.print(detik);
  ss.print(",");
  ss.print(tgl);
  ss.print(",");
  ss.print(bln);
  ss.print(",");
  ss.print(thn);
  ss.print(",");
  ss.print(last_h);
  ss.println("}");
  timeBack = millis();
}

//  Serial.println("logicRelay=" + String(logicRelay));
//  Serial.println("logicVoice=" + String(logicVoice));
//  Serial.println("pin8=" + String(digitalRead(8)));
//  Serial.println(String(last_h));

if (!digitalRead(tombol1)) {
  delay(1000);
  if (!digitalRead(tombol1)) {
    finger.emptyDatabase();
    eadd = 0;
    EEPROM.write(0, eadd);
    lcd.clear();
    delay(15);
    lcd.setCursor(3, 0);
```

```
      lcd.print("Sidik Jari");
      lcd.setCursor(1, 1);
      lcd.print("Telah Dihapus");
      delay(2500);
      lcd.clear();
      goto awal;
    }
    eadd += 1;
    if (eadd > 50)eadd = 0;
    EEPROM.write(0, eadd);
    getFingerprintEnroll(eadd);
    eadd = EEPROM.read(0);
  }
awal:
  getFingerprintIDez();
//  delay(10);
  show(true);
  if (digitalRead(8) == HIGH) {
    digitalWrite(relay1, LOW);
    last_h =  lastTime;
  }
  else {
    digitalWrite(relay1, HIGH);
  }
}

void show(bool x) {
  if (x == true) {
    if (millis() - timeShow > 3000) {

      flagRun++;
      timeShow = millis();
    }

    switch (flagRun) {
      case 0:
        flagRun = 1;
        break;
      case 1:
        lcd.setCursor (0, 0);
        lcd.print(F(" -System Ready- "));        lcd.setCursor(0, 1);
        lcd.print(" Tap Your Finger");
        break;
      case 2:
        lcd.setCursor(0, 0);
        lcd.print("   -Time NOW-   ");
        lcd.setCursor(0, 1);
        lcd.print(String(tgl) + "/" + String(bln) + "/23 " + String(jam) + ":" +
String(menit) + ":" + String(detik) + "   ");
        break;
      case 3:
        lcd.setCursor(0, 0);
        lcd.print("   -Last Time-  ");
        lcd.setCursor (0, 1);
        lcd.print(last_h);
        break;
      case 4:
        flagRun = 0;
        break;
    }
  }
}
#define countof(a) (sizeof(a) / sizeof(a[0]))
void printDateTime(const RtcDateTime & dt)
{
  char datestring[20];

  snprintf_P(datestring,
             countof(datestring),
             PSTR("%02u/%02u/%04u-%02u:%02u:%02u"),
             dt.Month(),
             dt.Day(),
             dt.Year(),
             dt.Hour(),
             dt.Minute(),
             dt.Second() );
  //  updateTime = datestring;
  lastTime = datestring;
  //  Serial.println(datestring);
}
void rtcSet() {
  Rtc.Begin();
  RtcDateTime compiled = RtcDateTime(__DATE__, __TIME__);
  printDateTime(compiled);
  Serial.println();
  if (!Rtc.IsDateTimeValid())
  {
    if (Rtc.LastError() != 0)
    {
      Serial.print("RTC communications error = ");
      Serial.println(Rtc.LastError());
    }
    else
    {
      Serial.println("RTC lost confidence in the DateTime!");
      Rtc.SetDateTime(compiled);
    }
  }
  if (!Rtc.GetIsRunning())
  {
    Serial.println("RTC was not actively running, starting now");
    Rtc.SetIsRunning(true);
  }

  RtcDateTime now = Rtc.GetDateTime();
  if (now < compiled)
  {
    Serial.println("RTC is older than compile time!  (Updating DateTime)");
    Rtc.SetDateTime(compiled);
  }

  Rtc.Enable32kHzPin(false);
  Rtc.SetSquareWavePin(DS3231SquareWavePin_ModeNone);
}
```

B. PROGRAM ENROLL

```
uint8_t getFingerprintEnroll(int id) {
  int p = -1;
  lcd.clear();
  lcd.setCursor(1, 0);
  lcd.print("Silahkan Scan");
  lcd.setCursor(3, 1);
  lcd.print("Sidik Jari");
  Serial.println( "daftarkan");
  //  delay(1000);
  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
      case FINGERPRINT_OK:
        break;
      case FINGERPRINT_NOFINGER:
        break;
      case FINGERPRINT_PACKETRECIEVEERR:
        break;
      case FINGERPRINT_IMAGEFAIL:
        break;
      default:
        break;
    }
  }

  p = finger.image2Tz(1);
  switch (p) {
    case FINGERPRINT_OK:

      break;
    case FINGERPRINT_IMAGEMESS:
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:

      return p;
    case FINGERPRINT_FEATUREFAIL:
      delay(200);
      return p;
    case FINGERPRINT_INVALIDIMAGE:
      delay(200);
      return p;
    default:

      return p;
  }
  lcd.clear();
  lcd.setCursor(1, 0);
  lcd.print("Lepaskan Jari");
  Serial.println("Lepaskan jari");
  delay(1000);


  p = 0;
  while (p != FINGERPRINT_NOFINGER) {
    p = finger.getImage();
  }

  p = -1;
  lcd.clear();
  delay(15);
  lcd.setCursor(1, 0);
  lcd.print("Tempelkan Jari");
  Serial.println("Tempelkan jari");
  delay(1000);
  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
      case FINGERPRINT_OK:
        break;
      case FINGERPRINT_NOFINGER:
        break;
      case FINGERPRINT_PACKETRECIEVEERR:
        break;
      case FINGERPRINT_IMAGEFAIL
        break;
      default:
      break;
    }
  }

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

      break;
    case FINGERPRINT_IMAGEMESS:

      return p;
    case FINGERPRINT_PACKETRECIEVEERR:

      return p;
    case FINGERPRINT_FEATUREFAIL:

      return p;
    case FINGERPRINT_INVALIDIMAGE:

      return p;
    default:

      return p;
  }

  p = finger.createModel();
  if (p == FINGERPRINT_OK) {


  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {

    return p;
  } else if (p == FINGERPRINT_ENROLLMISMATCH) {
    Serial.println("gagal tersimpan");
    lcd.clear();
    delay(15);
    lcd.setCursor(1, 0);
    lcd.print("Gagal Disimpan");
    delay(1000);
    delay(15);
    lcd.clear();

    return p;
  } else {

    return p;
  }

  p = finger.storeModel(id);
  if (p == FINGERPRINT_OK) {
    lcd.clear();
    lcd.setCursor(4, 0);
    lcd.print("Berhasil");
    lcd.setCursor(4, 1);
    lcd.print("Disimpan");
    Serial.println("Disimpan");
    delay(2500);
    delay(15);
    lcd.clear();

  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {

    return p;
  } else if (p == FINGERPRINT_BADLOCATION) {

    return p;
  } else if (p == FINGERPRINT_FLASHERR) {

    return p;
  } else {

    return p;
  }
}
```

C. PROGRAM FINGERPRINT

```
uint8_t getFingerprintID() {
  uint8_t p = finger.getImage();
  switch (p) {
    case FINGERPRINT_OK:
      break;
    case FINGERPRINT_NOFINGER:

      return p;
    case FINGERPRINT_PACKETRECIEVEERR:

      return p;
    case FINGERPRINT_IMAGEFAIL:

      return p;
    default:

      return p;
  }

  p = finger.image2Tz();
  switch (p) {
    case FINGERPRINT_OK:
      break;
    case FINGERPRINT_IMAGEMESS:

      return p;
    case FINGERPRINT_PACKETRECIEVEERR:

      return p;
    case FINGERPRINT_FEATUREFAIL:

      return p;
    case FINGERPRINT_INVALIDIMAGE:

      return p;
    default:
```

```
    return p;
  }

  p = finger.fingerFastSearch();
  if (p == FINGERPRINT_OK) {

  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {

    return p;
  } else if (p == FINGERPRINT_NOTFOUND) {

    return p;
  } else {
    return p;
  }
}

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)
  {

    lcd.clear();
    lcd.setCursor(1, 0);
    lcd.print("Akses Ditolak");

    Serial.println("Salah");

    while (p != FINGERPRINT_NOFINGER) {
      p = finger.getImage();
```

```
      Serial.println("Lepaskan");
    }
    lcd.clear();
    delay(15);
    return -1;
  }

  last_h =  lastTime;
  //  last_m = menit;
  lcd.setCursor(1, 0);
  lcd.print("Akses Diterima");
  lcd.setCursor (0, 1);
  lcd.print(F("AutoLock after "));
  digitalWrite(relay1, LOW);
  for (int i = 5; i > 0; i--) {
    lcd.setCursor (15, 1);
    lcd.print(i);
    Serial.println(i);
    delay (1000);
  }

  digitalWrite(relay1, HIGH);
  //  logicRelay = 2;
  dataRelay = dataRelay + 1;

  if (dataRelay >= 2) {
    dataRelay = 0;
  }
  lcd.clear();

  return finger.fingerID;
}
```

D. PROGRAM KEAMANAAN MENGGUNAKAN ESP 8266

```
// Keamananbrankas03@gmail.com
//brankas2023@
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
//#include <Wire.h>
//#include <LiquidCrystal_I2C.h>
#include <SoftwareSerial.h>
#include "VoiceRecognitionV3.h"
#define Open    (1)
#define Close   (2)
#define Buka    (3)
#define Tutup   (4)

char auth[] = "S_wOAq-cQYX6hi7bjQmtC-TxV6Wvdey4";
char ssid[] = "Ta diki";
char pass[] = "00000000";
String data;
String jamStr, menitStr, detikStr, tglStr, blnStr, thnStr, lastTime;
unsigned long timeBack = 0;
int flagRun;
unsigned long timeShow = 0;
int pinValue ;
int pinValue1 ;
String sendSS;
uint8_t records[7]; // save record
uint8_t buf[64];
int onRelay;
BlynkTimer timer;
WidgetLCD lcd(V0);
SoftwareSerial ss(12, 14); //rx tx
VR myVR(4, 5);//2 1
BLYNK_WRITE(V1)
{
  pinValue = param.asInt(); // assigning incoming value from pin V1 to
a variable
}
void myTimerEvent()
```

```
{
  show();
}
void setup() {
  Serial.begin(115200);
  ss.begin(9600);
  voiceSet();
    Serial.print(blnStr);
    Serial.print("/");
    Serial.println(thnStr);
    Serial.println("lastTime" + String(lastTime));
    timeBack = millis();
  }

  if (pinValue == 1) {
    digitalWrite(2,  HIGH);
  }
  else {
    //   digitalWrite(2, onRelay);
    int ret;
    ret = myVR.recognize(buf, 50);
    if (ret > 0) {
      switch (buf[1]) {
        case Open://1
          Serial.println("open");
          digitalWrite(2,  HIGH);
          onRelay = 1;
          //       logicVoice = 1;
          //       digitalWrite(relay1, LOW);
          break;
        case Close://3
          Serial.println("close");
          digitalWrite(2,  LOW);
          onRelay = 0;
          //       logicVoice = 3;
          //       digitalWrite(relay1, HIGH);
          break;
        case Buka://2
          Serial.println("buka");
          digitalWrite(2,  HIGH);
          onRelay = 1;
          //        logicVoice = 2;
          //        digitalWrite(relay1, LOW);
          break;
        case Tutup://4
          Serial.println("tutup");
          digitalWrite(2,  LOW);
          onRelay = 0;
          //        logicVoice = 4;
          //        digitalWrite(relay1, HIGH);
          break;
        default:
          Serial.println("Record function undefined");
          break;
      }
      printVR(buf);
    }
  }

  Blynk.run();
  timer.run();
}

void show() {
  if (millis() - timeShow > 3000) {

    flagRun++;
    timeShow = millis();
  }

  switch (flagRun) {

    case 0:
      flagRun = 1;
      break;
    case 1:
      lcd.print(0, 0, " -System Ready- ");
      lcd.print(0, 1, " Tap Your Finger");
      break;
    case 2:
      lcd.print(0, 0, "   -Time NOW-   ");
      lcd.print(0, 1, String(tglStr) + "/" + String(blnStr) + "/" +
String(thnStr) + " " + String(jamStr) + ":" + String(menitStr) + "  ");
      break;
    case 3:      lcd.print(0, 0, "   -Last Time-  ");
      lcd.print(0, 1, lastTime);
      break;
    case 4:
      flagRun = 0;
      break;
  }
}
```

E. PROGRAM VOICE SET

```
void voiceSet() {
  /** initialize */
  myVR.begin(9600);
  Serial.println("Elechouse Voice Recognition V3 Module\r\nControl
LED sample");

//    pinMode(13, OUTPUT);

  if (myVR.clear() == 0) {
    Serial.println("Recognizer cleared.");
  } else {
    Serial.println("Not find VoiceRecognitionModule.");
    Serial.println("Please check connection and restart Arduino.");
    while (1);
  }

  if (myVR.load((uint8_t)Open) >= 0) {
    Serial.println("open loaded");
  }

  if (myVR.load((uint8_t)Close) >= 0) {
    Serial.println("close loaded");
  }
  if (myVR.load((uint8_t)Buka) >= 0) {
    Serial.println("buka loaded");
  }

  if (myVR.load((uint8_t)Tutup) >= 0) {
    Serial.println("tutup loaded");
  }}

void printVR(uint8_t *buf) {
  Serial.println("VR Index\tGroup\tRecordNum\tSignature");
  Serial.print(buf[2], DEC);
  Serial.print("\t\t");
  if (buf[0] == 0xFF) {
    Serial.print("NONE");
  }
  else if (buf[0] & 0x80) {
    Serial.print("UG ");
    Serial.print(buf[0] & (~0x80), DEC);
  }
  else {
    Serial.print("SG ");
    Serial.print(buf[0], DEC);
  }
  Serial.print("\t");
  Serial.print(buf[1], DEC);
  Serial.print("\t\t");
  if (buf[3] > 0) {
    printSignature(buf + 4, buf[3]);
  }
  else {
    Serial.print("NONE");
  }
  Serial.println("\r\n");
}
void printSignature(uint8_t *buf, int len) {
  int i;
  for (i = 0; i < len; i++) {
    if (buf[i] > 0x19 && buf[i] < 0x7F) {
      Serial.write(buf[i]);
    }
    else {
      Serial.print("[");
      Serial.print(buf[i], HEX);
      Serial.print("]");
    }
  }
}
```