**LISTING PROGRAM ARDUINO**

**Serial**

void serialEvent() {

while (Serial.available()) {

char inChar = (char)Serial.read();

inputString += inChar;

if (inChar == '\n') {

if(inputString=="up\n"){

Serial.println("ali");

}

else if(inputString.substring(0,1)=="i"){

num =inputString.substring(1).toInt();

id = num;

getFingerprintEnroll();

// Serial.println(id);

}

else if(inputString.substring(0,1)=="o"){

buka();

}

inputString="";

}

}

}

void buka(){ //akses doorlock

lcd.clear();

lcd.print("Akses Diterima");

delay(1000);

lcd.setCursor(0,1);

lcd.print("Silahkan Masuk");

digitalWrite(7,LOW); // pin relay doorlock

delay(5000);

digitalWrite(7,HIGH);

}

**finger\_prog**

#include <Adafruit\_Fingerprint.h> //library

#include <SoftwareSerial.h>

#include <SPI.h>

#include <MFRC522.h>

#include <LiquidCrystal\_I2C.h>

#include <Wire.h>

#define SS\_PIN 10 //inisialisasi

#define RST\_PIN 9

MFRC522 rfid(SS\_PIN,RST\_PIN);

#define I2C\_ADDR 0x27 // <<- Add your address here.

#define Rs\_pin 0

#define Rw\_pin 1

#define En\_pin 2

#define BACKLIGHT\_PIN 3

#define D4\_pin 4

#define D5\_pin 5

#define D6\_pin 6

#define D7\_pin 7

LiquidCrystal\_I2C lcd(I2C\_ADDR,En\_pin,Rw\_pin,Rs\_pin,D4\_pin,D5\_pin,D6\_pin,D7\_pin); //konfigurasi lcd

SoftwareSerial mySerial(2, 3); //serial data sensor finger

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&mySerial);

int nilai[1][4] = {{202,249,167,233}};

String inputString = "";

int id;

int num = 0;

void setup() {

// put your setup code here, to run once:

Serial.begin(9600); //baudrate atau kecepatan data transfer

SPI.begin();

rfid.PCD\_Init();

Wire.begin();

pinMode(7,OUTPUT); //pin relay doorlock

pinMode(6,INPUT\_PULLUP); //push button

// set the data rate for the sensor serial port

finger.begin(57600); //baudrate serial (bit per second)

if (finger.verifyPassword()) {

Serial.println("Found fingerprint sensor!");

} else {

Serial.println("Did not find fingerprint sensor :(");

while (1) { delay(1); }

}

finger.getTemplateCount();

Serial.print("Sensor contains "); Serial.print(finger.templateCount); Serial.println(" templates");

Serial.println("Waiting for valid finger...");

lcd.begin (16,2);

// LCD Backlight ON

lcd.setBacklightPin(BACKLIGHT\_PIN,POSITIVE);

lcd.setBacklight(HIGH);

lcd.home (); // go home on LCD

lcd.setCursor (0,0);

lcd.print("Door Lock System");

lcd.setCursor (0,1);

lcd.print("Pintu R.Teknisi");

delay(2000);

lcd.clear();

digitalWrite(7,HIGH);

}

//program utama

void loop() {

// put your main code here, to run repeatedly:

lcd.setCursor (0,0);

lcd.print("Silahkan Tempel ");

lcd.setCursor (0,1);

lcd.print("Jari atau Kartu ");

getFingerprintIDez(); //fungsi finger

//RFID Code

if(rfid.PICC\_IsNewCardPresent()) //perintak pengecekan kartu

{

if(rfid.PICC\_ReadCardSerial()) //perintah baca kartu rfid

{

Serial.print(rfid.uid.uidByte[0]); //data 3 digit awal kartu dan di kirim ke aplikasi

Serial.print(rfid.uid.uidByte[1]);

Serial.println(",");

}

}

rfid.PICC\_HaltA();

rfid.PCD\_StopCrypto1();

//push button doorlock

if(digitalRead(6) == LOW){

digitalWrite(7,LOW); //buka doorlock

delay(5000);

digitalWrite(7,HIGH); //tutup doorlock

}

}

**Finger**

int getFingerprintIDez() {

uint8\_t p = finger.getImage();

if (p != FINGERPRINT\_OK) return -1;

p = finger.image2Tz(); //pada saat tidak ada id finger yg terdaftar

if (p!=2){

Serial.println("no finger");

lcd.setCursor (0,0);

lcd.print(" Akses Ditolak ");

lcd.setCursor (0,1);

lcd.print(" ");

delay(2000);

}

if (p != FINGERPRINT\_OK) return -1;

p = finger.fingerFastSearch(); //id finger terdaftar

if (p != FINGERPRINT\_OK) return -1;

// pass = true;

// sidik = false;

// found a match!

// Serial.print("Found ID #");

Serial.print(finger.fingerID); //kirim data ID ke aplikasi

Serial.println(",");

// Serial.print(" with confidence of "); Serial.println(finger.confidence);

buka(); //akses doorlock

return finger.fingerID;

}

uint8\_t readnumber(void) {

while (num == 0) {

while (! Serial.available());

num = Serial.parseInt();

}

return num;

}

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;

}

}

// OK success!

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;

}

}

// OK success!

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;

}

delay(2000);

id = 0;

}