**A Project on**

FINGERPRINT BASED BIOMETRIC ATTENDANCE SYSTEM USING ARDUINO

**Prepared By:**

Rahul N Suhagiya(17EC091)

**Under the guidance of**

**Prof. Vishal Tank**

****

DEPARTMENT OF ELECTRONICS & COMMUNICATION

Faculty of Technology & Engineering, CHARUSAT

Chandubhai S. Patel Institute of Technology

At: Changa, Dist: Anand– 388421

**Abstract :**

Fingerprints are a form of biometric identification which is unique and does not change in one’s entire lifetime. This paper presents the attendance management system using fingerprint technology in a university environment. It consists of two processes namely; enrolment and authentication. During enrolment, the fingerprint of the user is captured and its unique features extracted and stored in a database along with the users identity as a template for the subject.

So, after getting introduced to the Arduino Uno and other peripherals and its programming and interfacing, I was since then very fascinated and was wanting to try something new out of it. So, I learnt how to interface a RTC Module and 16\*2 LCD and a fingerprint module and then created a fingerprint attendance system.

This project is mainly divided into two parts namely; Hardware & Software (Code). To be frank the hardware part is pretty easy and the main brain of this project is the programming and coding. I have created a program for the attendance system such that it could get fit inside the 16\*2 lcd display of the screen and then very accurately have interfaced the fingerprint module as well, just to make it more user friendly.

**Application :**

* Fingerprint based attendance management system” project can be used by the :
* Employees
* Staff
* Workers
* Schools
* Universities

**CODE:**

#include<EEPROM.h>

#include<LiquidCrystal.h> // for lcd library

LiquidCrystal lcd(13,12,11,10,9,8); // for defining Pin Number in Arduino for interfacing Lcd

#include <SoftwareSerial.h> //For serial communication

SoftwareSerial fingerPrint(2, 3);

#include <Wire.h>

#include "RTClib.h" // For defining Library for RTC module

RTC\_DS1307 rtc;

#include "Adafruit\_Fingerprint.h"

uint8\_t id;

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

// For Defining Digital Sensors Pin Number in Arduino for interfacing push button

#define enroll A0

#define del A1

#define up A2

#define down A3

#define match 5

#define indFinger 7

#define buzzer 5

#define records 4 // 5 for 5 user

int user1,user2,user3,user4,user5;

DateTime now;

// Setups all Datatypes

void setup()

{

delay(1000);

lcd.begin(16,2);

Serial.begin(9600);

pinMode(enroll, INPUT\_PULLUP);

pinMode(up, INPUT\_PULLUP);

pinMode(down, INPUT\_PULLUP);

pinMode(del, INPUT\_PULLUP);

pinMode(match, INPUT\_PULLUP);

pinMode(buzzer, OUTPUT);

pinMode(indFinger, OUTPUT);

digitalWrite(buzzer, LOW);

if(digitalRead(enroll) == 0)

{

digitalWrite(buzzer, HIGH);

delay(500);

digitalWrite(buzzer, LOW);

lcd.clear();

lcd.print("Please wait");

lcd.setCursor(0,1);

lcd.print("Downloding Data");

Serial.println("Please wait");

Serial.println("Downloding Data..");

Serial.println();

Serial.print("S.No. ");

for(int i=0;i<records;i++)

{

digitalWrite(buzzer, HIGH);

delay(500);

digitalWrite(buzzer, LOW);

Serial.print(" User ID");

Serial.print(i+1);

Serial.print(" ");

}

Serial.println();

int eepIndex=0;

for(int i=0;i<30;i++)

{

if(i+1<10)

Serial.print('0');

Serial.print(i+1);

Serial.print(" ");

eepIndex=(i\*7);

download(eepIndex);

eepIndex=(i\*7)+210;

download(eepIndex);

eepIndex=(i\*7)+420;

download(eepIndex);

eepIndex=(i\*7)+630;

download(eepIndex);

// eepIndex=(i\*7)+840; // 5th user

// download(eepIndex);

Serial.println();

}

}

if(digitalRead(del) == 0)

{

lcd.clear();

lcd.print("Please Wait");

lcd.setCursor(0,1);

lcd.print("Reseting.....");

for(int i=1000;i<1005;i++)

EEPROM.write(i,0);

for(int i=0;i<841;i++)

EEPROM.write(i, 0xff);

lcd.clear();

lcd.print("System Reset");

delay(1000);

}

lcd.clear();

lcd.print(" Attendance ");

lcd.setCursor(0,1);

lcd.print(" System ");

delay(2000);

lcd.clear();

lcd.print(" ");

lcd.setCursor(0,1);

lcd.print("RAHUL SUHAGIYA");

delay(2000);

digitalWrite(buzzer, HIGH);

delay(500);

digitalWrite(buzzer, LOW);

for(int i=1000;i<1000+records;i++)

{

if(EEPROM.read(i) == 0xff)

EEPROM.write(i,0);

}

finger.begin(57600);

Serial.begin(9600);

lcd.clear();

lcd.print("Finding Module");

lcd.setCursor(0,1);

delay(1000);

if (finger.verifyPassword())

{

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

lcd.clear();

lcd.print("Found Module ");

delay(1000);

}

else

{

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

lcd.clear();

lcd.print("module not Found");

lcd.setCursor(0,1);

lcd.print("Check Connections");

while (1);

}

if (! rtc.begin())

Serial.println("Couldn't find RTC");

// rtc.adjust(DateTime(F(\_\_DATE\_\_), F(\_\_TIME\_\_)));

if (! rtc.isrunning())

{

Serial.println("RTC is NOT running!");

// following line sets the RTC to the date & time this sketch was compiled

rtc.adjust(DateTime(F(\_\_DATE\_\_), F(\_\_TIME\_\_)));

// This line sets the RTC with an explicit date & time, for example to set

// January 21, 2014 at 3am you would call:

// rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));

}

lcd.setCursor(0,0);

lcd.print("Press Match to ");

lcd.setCursor(0,1);

lcd.print("Start System");

delay(2000);

user1=EEPROM.read(1000);

user2=EEPROM.read(1001);

user3=EEPROM.read(1002);

user4=EEPROM.read(1003);

user5=EEPROM.read(1004);

lcd.clear();

digitalWrite(indFinger, HIGH);

}

void loop()

{

now = rtc.now();

lcd.setCursor(0,0);

lcd.print("Time->");

lcd.print(now.hour(), DEC);

lcd.print(':');

lcd.print(now.minute(), DEC);

lcd.print(':');

lcd.print(now.second(), DEC);

lcd.print(" ");

lcd.setCursor(0,1);

lcd.print("Date->");

lcd.print(now.day(), DEC);

lcd.print('/');

lcd.print(now.month(), DEC);

lcd.print('/');

lcd.print(now.year(), DEC);

lcd.print(" ");

delay(500);

int result=getFingerprintIDez();

if(result>0)

{

digitalWrite(indFinger, LOW);

digitalWrite(buzzer, HIGH);

delay(100);

digitalWrite(buzzer, LOW);

lcd.clear();

lcd.print("ID:");

lcd.print(result);

lcd.setCursor(0,1);

lcd.print("Please Wait....");

delay(1000);

attendance(result);

lcd.clear();

lcd.print("Attendance ");

lcd.setCursor(0,1);

lcd.print("Registed");

delay(1000);

digitalWrite(indFinger, HIGH);

return;

}

checkKeys();

delay(300);

}

// dmyyhms - 7 bytes

void attendance(int id)

{

int user=0,eepLoc=0;

if(id == 1)

{

eepLoc=0;

user=user1++;

}

else if(id == 2)

{

eepLoc=210;

user=user2++;

}

else if(id == 3)

{

eepLoc=420;

user=user3++;

}

else if(id == 4)

{

eepLoc=630;

user=user4++;

}

/\*else if(id == 5) // fifth user

{

eepLoc=840;

user=user5++;

}\*/

else

return;

int eepIndex=(user\*7)+eepLoc;

EEPROM.write(eepIndex++, now.hour());

EEPROM.write(eepIndex++, now.minute());

EEPROM.write(eepIndex++, now.second());

EEPROM.write(eepIndex++, now.day());

EEPROM.write(eepIndex++, now.month());

EEPROM.write(eepIndex++, now.year()>>8 );

EEPROM.write(eepIndex++, now.year());

EEPROM.write(1000,user1);

EEPROM.write(1001,user2);

EEPROM.write(1002,user3);

EEPROM.write(1003,user4);

// EEPROM.write(4,user5); // figth user

}

void checkKeys()

{

if(digitalRead(enroll) == 0)

{

lcd.clear();

lcd.print("Please Wait");

delay(1000);

while(digitalRead(enroll) == 0);

Enroll();

}

else if(digitalRead(del) == 0)

{

lcd.clear();

lcd.print("Please Wait");

delay(1000);

delet();

}

}

void Enroll()

{

int count=1;

lcd.clear();

lcd.print("Enter Finger ID:");

while(1)

{

lcd.setCursor(0,1);

lcd.print(count);

if(digitalRead(up) == 0)

{

count++;

if(count>records)

count=1;

delay(500);

}

else if(digitalRead(down) == 0)

{

count--;

if(count<1)

count=records;

delay(500);

}

else if(digitalRead(del) == 0)

{

id=count;

getFingerprintEnroll();

for(int i=0;i<records;i++)

{

if(EEPROM.read(i) != 0xff)

{

EEPROM.write(i, id);

break;

}

}

return;

}

else if(digitalRead(enroll) == 0)

{

return;

}

}

}

void delet()

{

int count=1;

lcd.clear();

lcd.print("Enter Finger ID");

while(1)

{

lcd.setCursor(0,1);

lcd.print(count);

if(digitalRead(up) == 0)

{

count++;

if(count>records)

count=1;

delay(500);

}

else if(digitalRead(down) == 0)

{

count--;

if(count<1)

count=records;

delay(500);

}

else if(digitalRead(del) == 0)

{

id=count;

deleteFingerprint(id);

for(int i=0;i<records;i++)

{

if(EEPROM.read(i) == id)

{

EEPROM.write(i, 0xff);

break;

}

}

return;

}

else if(digitalRead(enroll) == 0)

{

return;

}

}

}

uint8\_t getFingerprintEnroll()

{

int p = -1;

lcd.clear();

lcd.print("finger ID:");

lcd.print(id);

lcd.setCursor(0,1);

lcd.print("Place Finger");

delay(2000);

while (p != FINGERPRINT\_OK)

{

p = finger.getImage();

switch (p)

{

case FINGERPRINT\_OK:

Serial.println("Image taken");

lcd.clear();

lcd.print("Image taken");

break;

case FINGERPRINT\_NOFINGER:

Serial.println("No Finger");

lcd.clear();

lcd.print("No Finger");

break;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

lcd.clear();

lcd.print("Comm Error");

break;

case FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

lcd.clear();

lcd.print("Imaging Error");

break;

default:

Serial.println("Unknown error");

lcd.clear();

lcd.print("Unknown Error");

break;

}

}

// OK success!

p = finger.image2Tz(1);

switch (p) {

case FINGERPRINT\_OK:

Serial.println("Image converted");

lcd.clear();

lcd.print("Image converted");

break;

case FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

lcd.clear();

lcd.print("Image too messy");

return p;

case FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

lcd.clear();

lcd.print("Comm Error");

return p;

case FINGERPRINT\_FEATUREFAIL:

Serial.println("Could not find fingerprint features");

lcd.clear();

lcd.print("Feature Not Found");

return p;

case FINGERPRINT\_INVALIDIMAGE:

Serial.println("Could not find fingerprint features");

lcd.clear();

lcd.print("Feature Not Found");

return p;

default:

Serial.println("Unknown error");

lcd.clear();

lcd.print("Unknown Error");

return p;

}

Serial.println("Remove finger");

lcd.clear();

lcd.print("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");

lcd.clear();

lcd.print("Place Finger");

lcd.setCursor(0,1);

lcd.print(" 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");

return;

}

}

// 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!");

lcd.clear();

lcd.print("Stored!");

delay(2000);

} 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;

}

}

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.print("Finger Not Found");

lcd.setCursor(0,1);

lcd.print("Try Later");

delay(2000);

return -1;

}

// found a match!

Serial.print("Found ID #");

Serial.print(finger.fingerID);

return finger.fingerID;

}

uint8\_t deleteFingerprint(uint8\_t id)

{

uint8\_t p = -1;

lcd.clear();

lcd.print("Please wait");

p = finger.deleteModel(id);

if (p == FINGERPRINT\_OK)

{

Serial.println("Deleted!");

lcd.clear();

lcd.print("Figer Deleted");

lcd.setCursor(0,1);

lcd.print("Successfully");

delay(1000);

}

else

{

Serial.print("Something Wrong");

lcd.clear();

lcd.print("Something Wrong");

lcd.setCursor(0,1);

lcd.print("Try Again Later");

delay(2000);

return p;

}

}

void download(int eepIndex)

{

if(EEPROM.read(eepIndex) != 0xff)

{

Serial.print("T->");

if(EEPROM.read(eepIndex)<10)

Serial.print('0');

Serial.print(EEPROM.read(eepIndex++));

Serial.print(':');

if(EEPROM.read(eepIndex)<10)

Serial.print('0');

Serial.print(EEPROM.read(eepIndex++));

Serial.print(':');

if(EEPROM.read(eepIndex)<10)

Serial.print('0');

Serial.print(EEPROM.read(eepIndex++));

Serial.print(" D->");

if(EEPROM.read(eepIndex)<10)

Serial.print('0');

Serial.print(EEPROM.read(eepIndex++));

Serial.print('/');

if(EEPROM.read(eepIndex)<10)

Serial.print('0');

Serial.print(EEPROM.read(eepIndex++));

Serial.print('/');

Serial.print(EEPROM.read(eepIndex++)<<8 | EEPROM.read(eepIndex++));

}

else

{

Serial.print("---------------------------");

}

Serial.print(" ");