

D.K.T.E Society’s

Textile and Engineering Institute, Ichalkaranji

(An Autonomous Institute Affiliated to Shivaji University Kolhapur)

**Department of Electronics and Telecommunication A Mini-Project Report on**

***FINGERPRINT ATTENDANCE SYSTEM.***

**By**

|  |  |  |  |
| --- | --- | --- | --- |
| **SR.NO** | **STUDENT NAME** | **PRN** | **SIGN** |
| **1** | **Bhushan Amit Gandhi.** | **21UET028** |  |
| **2** | **Tejaswini Mahadev Chavan.** | **21UET013** |  |
| **3** | **Pooja Muttappa Baswanbagewadi.** | **21UET009** |  |

**Under Guidance of PRO. Y.V.SAWANT**

CERTIFICATE

This is to certify that of Second Year B. Tech [ETC] has satisfactorily completed mini-project entitled ***FINGERPRINT ATTENDANCE SYSTEM*** for partial fulfilment of Under Graduation in Electronics and Telecommunication Engineering at D.K.T.E’s Textile and Engineering Institute, Ichalkaranji for Academic Year 2022-2023

**By**

|  |  |  |
| --- | --- | --- |
| **SR.NO** | **STUDENT NAME** | **PRN** |
| **1** | **Bhushan Amit Gandhi.** | **21UET028** |
| **2** | **Tejaswini Mahadev Chavan.** | **21UET013** |
| **3** | **Pooja Muttappa Baswanbagewadi.** | **21UET009** |

Prof.Y.V.Sawant Prof. (Dr.)S.A.Patil Prof. (Dr.) P. V. Kadole

**Guide Head of ETC Department Director**

***Contents:***

# Abstract

1. **Aim**

# Block Diagram

1. **Hardware Requirement**

# Components Descriptions

1. **Schematic Diagram**
2. **Working principle**

# Code/program

1. **cost**

# Advantages

1. **Bibligraphy**

**ABSTRACT**

Attendance System is used to detect the user identification and time management. Attendance system is required in many different places such as offices, companies, schools, organization and institution etc. Attendance system enables the user to track the working hours and late arrivals, early departures, time taken on breaks and absentees. This paper describes one of the attendance system. The main aim of the paper is to construct the attendance system using Fingerprint module and RTC module. In this system Arduino Uno and Parallax Data acquisition tool (PLX-DAQ) are the main components to display record in Excel. In this project, we are going to design a Fingerprint Sensor Based Biometric Attendance System using Arduino. Simply we will be **interfacing fingerprint sensor with Arduino**, **LCD Display** & **RTC Module** to design the desired project. In this project, we used the fingerprint Module and Arduino to take and keep attendance data and records.

Biometric Attendance systems are commonly used systems to mark the presence in offices and schools. This project has a wide application in school, college, business organization, offices where marking of attendance is required accurately with time. By using the fingerprint sensor, the system will become more secure for the users.

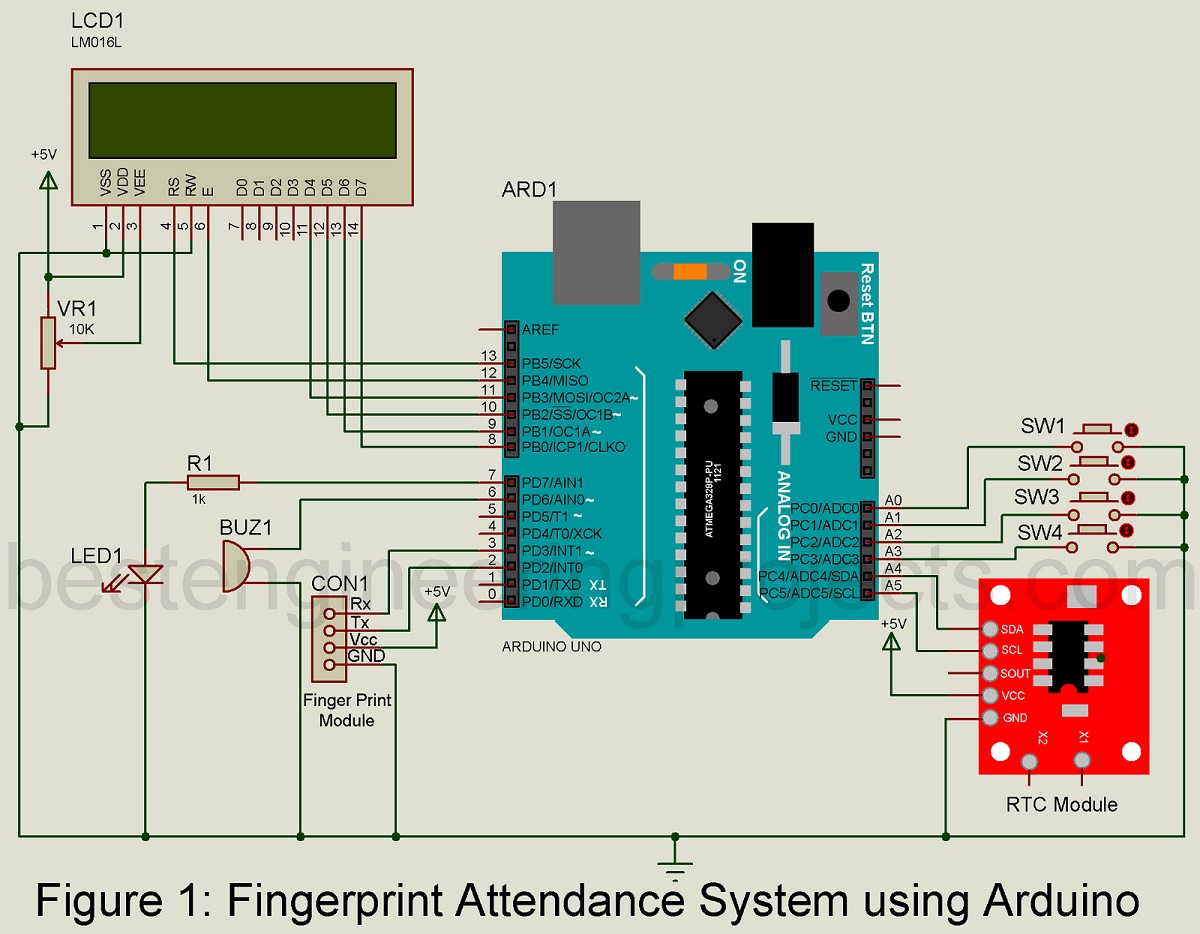
You can check the advanced version of this project here: [IoT Biometric Fingerprint Attendance System](https://how2electronics.com/iot-biometric-fingerprint-attendance-system-nodemcu/" \t "_blank). In case if you are not satisfied with R305/R307 Fingerprint sensors, you can refer to [GT511C3](https://how2electronics.com/interfacing-gt511c3-fingerprint-sensor-fps-with-arduino/) fingerprint sensor which is a faster and better fingerprint sensor.

**Aim**

The main aim that this system would test whether attendance by fingerprint is enough for identification. It is expected that the work in this system will reach the stage of being able to fully test hypothesis. Fingerprints are the oldest form of biometric identification.

Also to construct the attendance system using Fingerprint module and RTC module. In this system Arduino Uno and Parallax Data acquisition tool (PLX-DAQ) are the main components to display record in Excel.

# Block Diagram



**Hardware Requirement**

|  |  |  |  |
| --- | --- | --- | --- |
| SR.NO | COMPONENT | QUANTITY | |
| 1 | Arduino UNO Board | 1 | |
| 2 | R305 Fingerprint Sensor | 1 | |
| 3 | DS3231 RTC Module | 1 | |
| 4 | 16x2 LCD Display | 1 | |
| 5 | Push Buttons | 4 | |
| 6 | Resistor 2.7k ohm | 1 | |
| 7 | Buzzer 5V | 1 | |
| 8 | 3 volt battery | | 1 |
| 9 | 4volt battery | | 2 |
| 10 | SPST Switch | | 1 |
| 11 | Red LED | | 1 |
| 12 | Connector wires | |  |

# 

# Components Descriptions

# 1)Arduino UNO Board :

# 

Arduino/Genuino boards senses the environment by receiving inputs from many sensors, and affects their surroundings by controlling lights, motors, and other actuators. Arduino/Genuino boards are the microcontroller development platform that will be at the heart of your projects. When making something you will be building the circuits and interfaces for interaction, and telling the microcontroller how to interface with other components. Here the anatomy of Arduino/Genuino Uno.

* 1. **Digital pins** Use these pins with digital Read(), digital Write(), and analog Write(). Analog Write () works only on the pins with the PWM symbol.
* 2. **Pin 13 LED** The only actuator built-in to your board. Besides being a handy target for your first blink sketch, this LED is very useful for debugging.
* 3. **Power LED** Indicates that your Genuino is receiving power. Useful for debugging.
* 4. ATmega **microcontroller** The heart of your board.
* 5. **Analog in** Use these pins with analog Read().
* 6. **GND and 5V pins** Use these pins to provide +5V power and ground to your circuits.
* 7. **Power connector** This is how you power your Genuino when it’s not plugged into a USB port for power. Can accept voltages between 7-12V.
* 8. **TX and RX** LEDs These LEDs indicate communication between your Genuino and your computer. Expect them to flicker rapidly during sketch upload as well as during serial communication. Useful for debugging.
* 9. **USB port** Used for powering your Genuino Uno, uploading your sketches to your Genuino, and for communicating with your Genuino sketch (via Serial. println() etc.).
* 10. **Reset button** Resets the ATmega microcontroller.

### **2.R307 Fingerprint Scanner Sensor Module:**



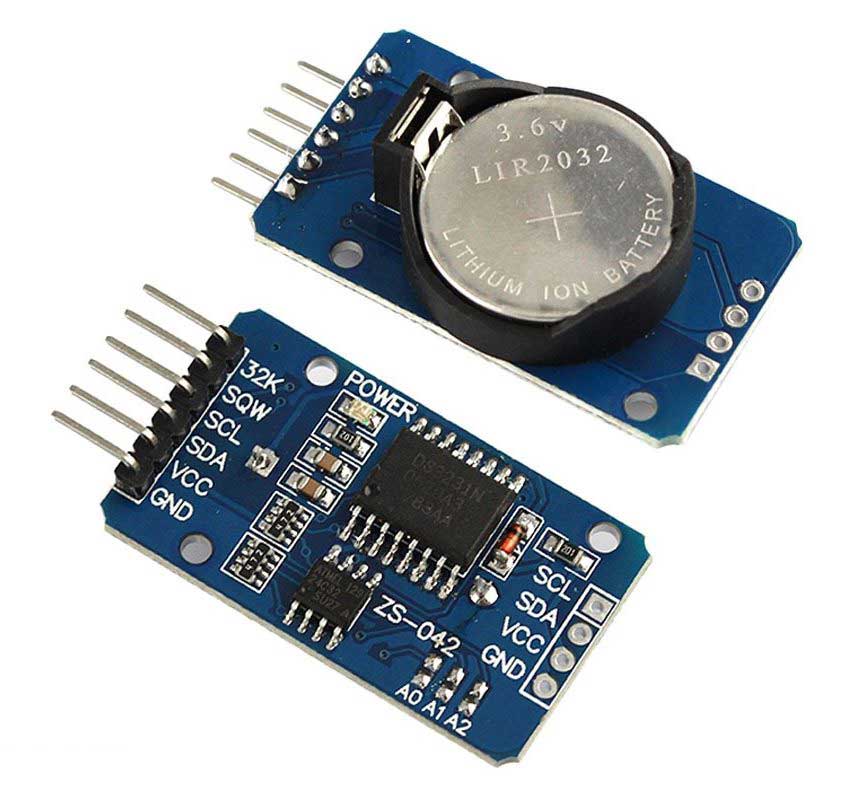
This is a fingerprint sensor module with TTL UART interface for direct connections to microcontroller UART or to PC through MAX232 / USB-Serial adapter. The user can store the fingerprint data in the module and can configure it in 1:1 or 1: N mode for identifying the person. The Fingerprint module can be directly interfaced with any microcontroller as well as Arduino Board. This optical biometric fingerprint reader with great features and can be embedded into a variety of end products like access control system, attendance system, safety deposit box, car door locking system.

#### ****Features****

1. Integrated image collecting and algorithm chip together, ALL-in-One
2. Fingerprint can conduct secondary development & embedded into a variety of end products
3. Low power consumption, low cost, small size, excellent performance
4. Professional optical technology, precise module manufacturing techniques
5. Good image processing capabilities can successfully capture image up to resolution 500 dpi
6. Fingerprint sensor type: Optical
7. Sensor Life: 100 million times
8. Static indicators: 15KVBacklight: bright green
9. Interface: USB1.1/UART(TTL logical level)
10. RS232 communication baud rate: 4800BPS~115200BPS changeable
11. Dimension: 55*32*21.5mm
12. Image Capture Surface 15—18(mm)
13. Verification Speed: 0.3 sec
14. Scanning Speed: 0.5 sec
15. Character file size: 256 bytes
16. Template size: 512 bytes
17. Storage capacity: 250
18. Security level: 5 (1,2,3,4,5(highest))
19. False Acceptance Rate (FAR) :0.0001%
20. False Rejection Rate (FRR): 0.1%
21. Resolution 500 DPI
22. Voltage :3.6-6.0 VDC
23. Working current: Typical 90 mA, Peak 150mA
24. Matching Method: 1: N

Operating Environment Temperature: -20 to 45° centigrades

3)DS3231 RTC Module:



RTC is an electronic device in the form of an Integrated Chip (IC) available in various packaging options. The purpose of an RTC or a real- time clock is to provide precise time and date which can be used for various applications.

DS3231 is a six terminal device, out of them two pins are not compulsory to use. So we have mainly four pins. These four pins are given out on other side of module sharing the same name.

### Specifications

RTC counts seconds, minutes, hours and year

Accuracy: +2ppm to -2ppm for 0ºC to +40ºC , +3.5ppm to -3.5ppm for - 40ºC to +85ºC

Digital temperature sensor with ±3ºC accuracy Two Time-of-day alarms Programmable square wave output

Register for Aging trim 400Khz I2C

interface Low power consumption

Automatic power failure battery switch circuitry CR2032 battery backup with two-to-three-year life Potable size

## LCD Display



An electronic device that is used to display data and the message is known as LCD 16×2. As the name suggests, it includes 16 Columns & 2 Rows so it can display 32 characters (16×2=32) in total & every character will be made with 5×8 (40) Pixel Dots. So, the total pixels within this LCD can be calculated as 32 x 40 otherwise 1280 pixels.

16 X2 displays mostly depend on multi-segment LEDs. There are different types of displays available in the market with different combinations such as 8×2, 8×1, 16×1, and 10×2, however, the LCD 16×2 is broadly used in devices, DIY circuits, electronic projects due to less cost, programmable friendly & simple to access

### Specifications

The operating voltage of this display ranges from 4.7V to 5.3V The display bezel is 72 x 25mm

The operating current is 1mA without a backlight PCB size of the module is 80L x 36W x 10H mm HD47780 controller

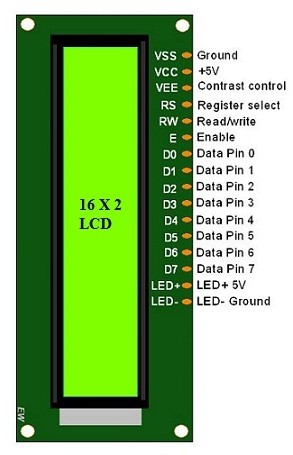
LED color for backlight is green or blue Number of columns – 16

Number of rows – 2 Number of LCD pins – 16 Characters – 32

It works in 4-bit and 8-bit modes

Pixel box of each character is 5×8 pixel

Font size of character is 0.125Width x 0.200height

**Pin Configuration**

Pin1 (Ground): This pin connects the ground terminal. Pin2 (+5 Volt): This pin provides a +5V supply to the LCD Pin3 (VE): This pin selects the contrast of the LCD.

Pin4 (Register Select): This pin is used to connect a data pin of an MCU & gets either 1 or 0. Here, data mode = 0 and command mode =1.

Pin5 (Read & Write): This pin is used to read/write data.

Pin6 (Enable): This enables the pin must be high to perform the Read/Write procedure. This pin is connected to the data pin of the microcontroller to be held high constantly.

Pin7 (Data Pin): The data pins are from 0-7 which are connected through the microcontroller for data transmission. The LCD module can also

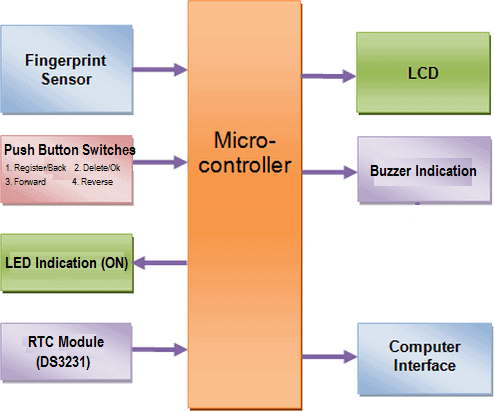
work on the 4-bit mode through working on pins 1, 2, 3 & other pins are free.

Pin8 – Data Pin 1 Pin9 – Data Pin 2 Pin10 – Data Pin 3 Pin11 – Data Pin 4 Pin12 – Data Pin 5 Pin13 – Data Pin 6 Pin14 – Data Pin 7

Pin15 (LED Positive): This is a +Ve terminal of the backlight LED of the display & it is connected to +5V to activate the LED backlight.

Pin16 (LED Negative): This is a -Ve terminal of a backlight LED of the display & it is connected to the GND terminal to activate the LED backlight.

1. **Schematic Diagram**



* **WORKING:**

The working of the Fingerprint Sensor Based Biometric Attendance System. In this project, we have used a DS3231 RTC Module for time & date display. We used 1 LED for power indication, 1 buzzer for different function indication. We have interfaced 16\*2 LCD which displays everything whenever the finger is placed or removed, or registering attendance or downloading data.

We have used 4 push buttons which are used to control the entire system. The functions of each button are:

1. Register/Back Button – Used for enrolling new fingerprint as well as reversing the back process or going back
2. Delete/OK Button – This Button is used for deleting the earlier stored fingerprint system as well as granting access as an OK selection.
3. Forward Button – Used for moving forward while selecting the memory location for storing or deleting fingerprints.
4. Reverse Button – Used for moving backward while selecting memory location for storing or deleting fingerprints.

Enrolling New Fingerprint:

To enroll New Fingerprint, click on the Enroll button. Then select the memory location where you want to store your fingerprint using the UP/DOWN button. Then click on OK. Put your finger and remove your finger as the LCD instructs. Put your finger again. So finally, your fingerprint gets stored.

## Deleting Stored Fingerprint:

To delete the fingerprint which is already clicked on DEL Button. Then select the memory location where your fingerprint was stored earlier using the UP/DOWN button. Then click on OK. So finally, your fingerprint is deleted.

## Downloading Data:

Simply click on Register/Back Button and reset the button together. At this movement, the serial monitor should be opened.

* **CODE:**

#include <RTClib.h>

#include <config.h>

#include <ds3231.h>

#include <Adafruit\_Fingerprint.h>

#include <Adafruit\_BusIO\_Register.h>

#include <Adafruit\_I2CDevice.h>

#include <Adafruit\_I2CRegister.h>

#include <Adafruit\_SPIDevice.h>

#include <Adafruit\_BusIO\_Register.h>

#include <Adafruit\_I2CDevice.h>

#include <Adafruit\_I2CRegister.h>

#include <Adafruit\_SPIDevice.h>

#include <Adafruit\_Fingerprint.h>

#include <config.h>

#include <ds3231.h>

#include <config.h>

#include <ds3231.h>

#include <Adafruit\_Fingerprint.h>

#include "Adafruit\_Fingerprint.h" //fingerprint library header file

#include<EEPROM.h> //command for storing data

#include<LiquidCrystal.h> //lcd header file

LiquidCrystal lcd(8,9,10,11,12,13);

#include <SoftwareSerial.h>

SoftwareSerial fingerPrint(2, 3); //for tx/rx communication between arduino & r305 fingerprint sensor

#include <Wire.h>

#include "RTClib.h" //library file for DS3231 RTC Module

RTC\_DS3231 rtc;

uint8\_t id;

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

#define register\_back 14

#define delete\_ok 15

#define forward 16

#define reverse 17

#define match 5

#define indFinger 7

#define buzzer 5

#define records 10 // 10 for 10 user

int user1,user2,user3,user4,user5,user6,user7,user8,user9,user10;

DateTime now;

void setup()

{

delay(1000);

lcd.begin(16,2);

Serial.begin(9600);

pinMode(register\_back, INPUT\_PULLUP);

pinMode(forward, INPUT\_PULLUP);

pinMode(reverse, INPUT\_PULLUP);

pinMode(delete\_ok, INPUT\_PULLUP);

pinMode(match, INPUT\_PULLUP);

pinMode(buzzer, OUTPUT);

pinMode(indFinger, OUTPUT);

digitalWrite(buzzer, LOW);

if(digitalRead(register\_back) == 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;

download(eepIndex);

eepIndex=(i\*7)+1050;

download(eepIndex);

eepIndex=(i\*7)+1260;

download(eepIndex);

eepIndex=(i\*7)+1470;

download(eepIndex);

eepIndex=(i\*7)+1680;

download(eepIndex);

Serial.println();

}

}

if(digitalRead(delete\_ok) == 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(" Fingerprint ");

lcd.setCursor(0,1);

lcd.print("Attendance System");

delay(2000);

lcd.clear();

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(2000);

if (finger.verifyPassword())

{

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

lcd.clear();

lcd.print(" Module Found");

delay(2000);

}

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.lostPower())

{

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

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

rtc.adjust(DateTime(2018, 6, 7, 11, 0, 0));

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

// June 7, 2018 at 11am you would call:

// rtc.adjust(DateTime(2018, 6, 7, 11, 0, 0));

}

lcd.setCursor(0,0);

lcd.print(" Press Match to ");

lcd.setCursor(0,1);

lcd.print(" Start System");

delay(3000);

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("Registered");

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)

{

eepLoc=0;

user=user5++;

}

else if(id == 6)

{

eepLoc=840;

user=user5++;

}

else if(id == 7)

{

eepLoc=1050;

user=user7++;

}

else if(id == 8)

{

eepLoc=1260;

user=user8++;

}

else if(id == 9)

{

eepLoc=1470;

user=user9++;

}

else if(id == 10)

{

eepLoc=1680;

user=user8++;

}

/\*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(register\_back) == 0)

{

lcd.clear();

lcd.print("Please Wait");

delay(1000);

while(digitalRead(register\_back) == 0);

Enroll();

}

else if(digitalRead(delete\_ok) == 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(forward) == 0)

{

count++;

if(count>records)

count=1;

delay(500);

}

else if(digitalRead(reverse) == 0)

{

count--;

if(count<1)

count=records;

delay(500);

}

else if(digitalRead(delete\_ok) == 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(register\_back) == 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(forward) == 0)

{

count++;

if(count>records)

count=1;

delay(500);

}

else if(digitalRead(reverse) == 0)

{

count--;

if(count<1)

count=records;

delay(500);

}

else if(digitalRead(delete\_ok) == 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(register\_back) == 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 Found");

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(" Finger 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("Finger 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(" ");

}

# Cost

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Component Name** | **Price** |
| 1. | Arduino UNO | 700 |
| 2. | Fingerprint Sensor | 1100 |
| 3. | RTC Module | 300 |
| 4. | LCD Display | 300 |
| 5. | Potentiometer | 70 |
| 6. | Push Buttons | 20 |
| 7. | Buzzer | 10 |
| 8. | LED | 10 |
| 9. | Jumper Wires | 100 |

**Advantages**

* Portable System
* Easy to use
* Light Weight
* User Friendly
* Convenient

# Bibliography

* https://create.arduino.cc/projecthub
* https://electronicsforu.com/
* https://how2electronics.com/
* https://electronicshub.org/
* https://[www.wikipedia.org/](http://www.wikipedia.org/)

***THANK YOU***