Fingerprint Based Biometric Attendance System using Arduino

Submitted by,

Sonu Kumar Kushwaha

Roll No. – 2K19/CO/383

Ayush Karn

Roll No. – 2K19/CO/454

Submitted to,

Mrs. Lavi Tanwar

Department of Electronics & Communication Engineering Delhi Technological University, Delhi Delhi-110042

May, 2021

CERTIFICATE

This is to certify that Sonu K. Kushwaha and Ayush Karn of Computer Science and Engineering Department (COE) having Roll No 2K19/CO/383 & 2K19/CO/454 respectively have successfully completed the project work entitled "**Fingerprint Based Biometric Attendance System using Arduino**" on Digital Electronics for Fourth Semester which is to be evaluated as the Mid Term Component.

Name: Sonu K Kushwaha & Ayush Karn

Signature: Sonu K Kushwaha & Ayush Karn

Date: 12-05-2021

Electronics & Communication Engg. Deptt.

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Bawana Road-Delhi-42

Candidate's Declaration

We, hereby, declare that the work embodied in this project entitled "Fingerprint Based Biometric Attendance System using Arduino" submitted to the Department of Electronics & Communication Engineering, Delhi Technological University, Delhi is an authentic record of my own bona-fide work and is correct to the best of my knowledge and belief. This work has been undertaken taking care of engineering ethics.

Name: Sonu Kumar Kushwaha & Ayush Karn

Signature: Sonu K Kushwaha & Ayush Karn

ACKNOWLEDGEMENT

We would like to express our special thanks of gratitude to teacher Mrs. Lavi Tanwar as well as our college (Delhi Technological University, Delhi) which gave us the golden opportunity to do this wonderful project on the topic “Fingerprint Based Biometric Attendance System using Arduino”, which also helped us in doing a lot of Research and we came to know about so many new things. We are really thankful to them.

Secondly, we would also like to thank our parents and friends who helped us a lot, in finalizing and completing this project within the limited time frame.

TABLE OF CONTENT

Page No

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

DEL:	**D**elete
EEPROM:	**E**lectrically **E**rasable **P**rogrammable **R**ead-only **M**emory
GUI:	**G**raphical **U**ser **I**nterface
IDE:	**I**ntegrated **D**evelopment **E**nvironment
ISO:	**I**nternational **O**rganization for **S**tandardization
IEC:	**I**nternational **E**lectrotechnical **C**ommission
LCD:	**L**iquid **C**rystal **D**isplay
LED:	**L**ight **E**mitting **D**iode
RTC:	**R**eal **T**ime **C**lock
USB:	**U**niversal **S**erial **B**us

LIST OF SYMBOLS

Greek

ρ	Resistivity
Ω	Electrical resistance
π	Pi Universally used for 3.1416

ABSTRACT

Attendance systems are commonly used systems to mark the presence in offices and schools. From manually marking the attendance in attendance registers is not so efficient in compare to high-tech applications and biometric systems, these systems have improved significantly in past years by introduction of biometric security systems in small mobile equipment like smartphones. In this project, we used fingerprint Module and Arduino to take and keep attendance data and records. By using fingerprint sensor, the system will become more secure for the users as well as efficient for the database management. Fingerprint attendance system aims to automate the attendance procedure of an educational institution using biometric technology. This will save time wasted on calling out names and it gives a fool-proof method of attendance marking. A hand-held device is used to mark the attendance without the intervention of teacher. The device can be passed and students can mark attendance during the lecture time. Students would be made to place their finger over the sensor so as to mark their presence in the class. It can communicate with a host computer using its USB interface. This device operates from a rechargeable battery. GUI application in host computer helps the teacher to manage the device and attendance.

INTRODUCTION

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.

In this fingerprint attendance system circuit, we used Fingerprint Sensor module to authenticate a true person or employee by taking their finger input in the system. Here we are using 4 push buttons to enroll, Delete, UP/Down. ENROLL and DEL key has triple features. ENROLL key is used for enrollment of a new person into the system. So, when the user wants to enroll new finger then he/she need to press ENROLL key then LCD asks for the ID, where user want to be store the finger print image. Now if at this time user does not want to proceed further then he/she can press ENROLL key again to go back. This time ENROLL key behave as Back key, i.e. ENROLL key has both enrollment and back function. Besides enroll key is also used to download attendance data over serial monitor. Similarly, DEL/OK key also has the same double function like when user enrolls new finger, then he/she need to select finger ID by using another two key namely UP and DOWN. Now user need to press DEL/OK key (this time this key behave like OK) to proceed with selected ID. And Del key is used for reset or delete data from EEPROM of Arduino.

METHODOLOGY AND IMPLEMENTATION

Concepts of Digital Electronics Used

1) Image of the Finger is converted into Computer understandable binary using some Algorithm that we we'll be explaining in Final Review.

2) EEPROM of Arduino is used to store the data scanned by fingerprint module.

3) Comparison of fingerprints is also done using some Algorithm which is also a part of Digital Electronics

4) Arduino is used which itself is a part of Digital Electronic

APPROACH

Considering the Pandemic Situation from very beginning of the 2nd year of our engineering, we are not being able to use any Laboratory hardware. So, we planned to go for Hardware implementation in this project. Even simple Bread-Board was new for us to use in our project. We firstly learned about the about the working Mechanism of the components, and then tried to make the complete circuit. Even the circuit Diagram was available with us, but Hardware implementation was quite a different experience than the Simulation. Final experience is shared in Conclusion.

ARDUINO

What is Arduino?

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board.

Why Arduino?

It's simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

- Inexpensive
- Cross-platform
- Simple, clear programming environment
- Open source and extensible software
- Open source and extensible hardware

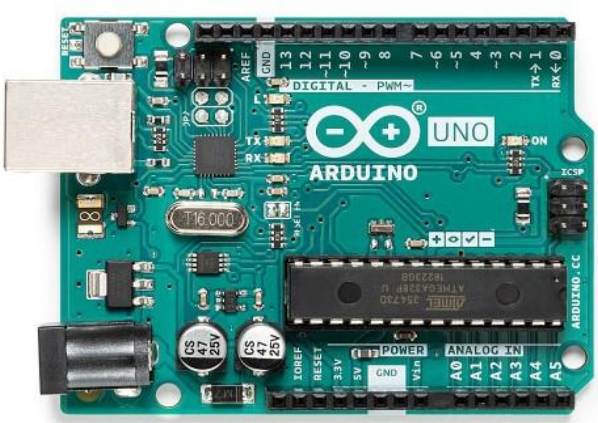


Figure 1 Arduino UNO

FINGERPRINT MODULE

Fingerprint sensor module captures finger's print image and then converts it into the equivalent template and saves them into its memory as per selected ID by Arduino. All the process is commanded by Arduino like taking an image of finger's print, convert it into templates and storing as ID etc.

Features:

- Integrated image collecting and algorithm chip together, ALL-in-One
- Fingerprint can conduct secondary development & embedded into a variety of end products
- Low power consumption, low cost, small size, excellent performance
- Professional optical technology, precise module manufacturing techniques
- Good image processing capabilities can successfully capture image up to resolution 500 dpi

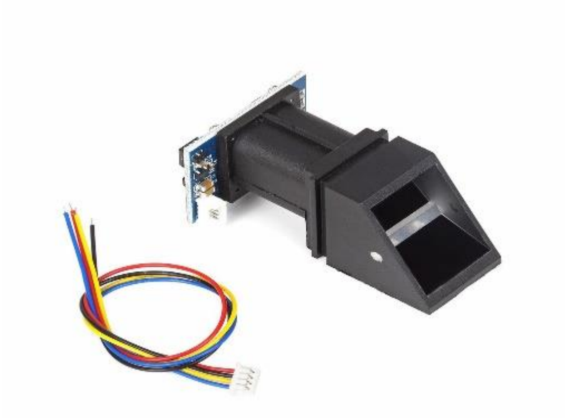


Figure 2 Fingerprint Module

FINGER’S PRINT IMAGE TO THE EQUIVALENT TEMPLATE

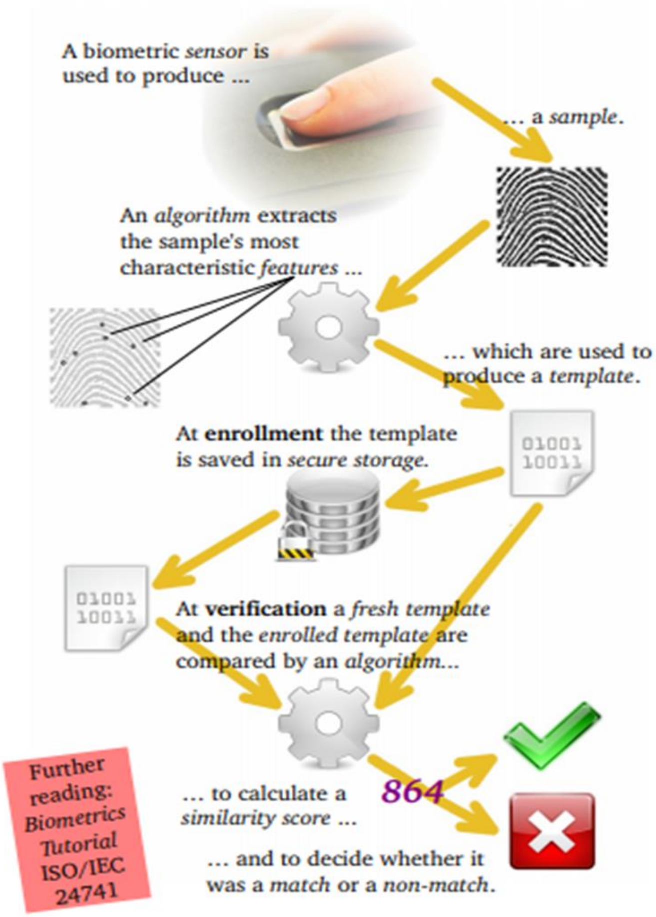

A biometric sensor is used to produce ...
... a sample.
An algorithm extracts the sample's most characteristic features ...
... which are used to produce a template.
01001
10011
At enrollment the template is saved in secure storage.
01001
10011
At verification a fresh template and the enrolled template are compared by an algorithm...
... to calculate a similarity score ...
864
... and to decide whether it was a match or a non-match.
Further reading: Biometrics Tutorial ISO/IEC 24741

Figure 3 Finger’s print image to the equivalent template

1. A biometric sensor is used to produce a sample image of fingerprint.
2. An algorithm extracts the most characteristic feature of the sample image.
3. These characteristic features are used to produce templates.
4. These templates are saved in secure place, of EEPROM of the Arduino.
5. At verification, a fresh template and the enrolled templates are compared by an algorithm.
6. The algorithm is used to calculate the similarity score and to decide whether it was a match or a non-match.

REQUIRED COMPONENTS

1. Arduino -1
2. Finger print module -1
3. Push Button - 4
4. LEDs -1
5. Power
6. Connecting wires
7. Buzzer -1
8. 16x2 LCD -1
9. Bread Board -1
10. RTC Module -1

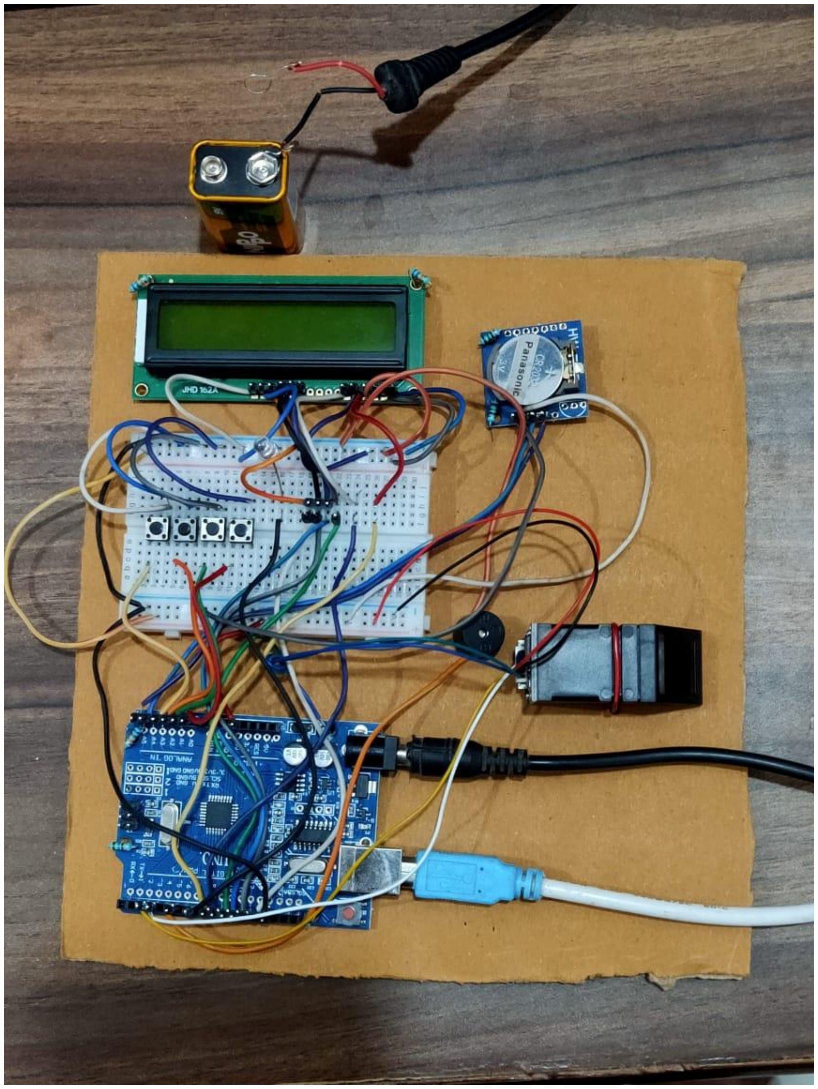


Figure 4 Hardware Setup

BLOCK-DIAGRAM

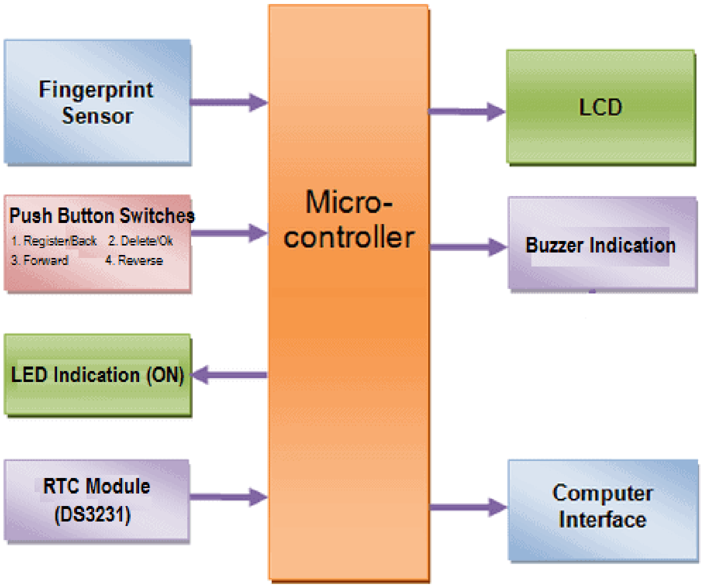


Figure 5 Block Diagram

In this Fingerprint Sensor Based Biometric Attendance System using Arduino, we used a Fingerprint Sensor module to authenticate a true person or employee by taking their finger input in the system. Here we are using 4 push buttons to register new fingerprint or delete stored fingerprint or match stored fingerprint. The 4 push buttons are used as an input unit for these tasks. Similarly, RTC Module DS3231 is used for registering scanning/entering/existing time of the user.

The LCD displays the time record and every function happening via push button. Buzzer indicates different functions and happening whenever an interrupt is detected. The LED is used for power indication.

CIRCUIT DIAGRAM

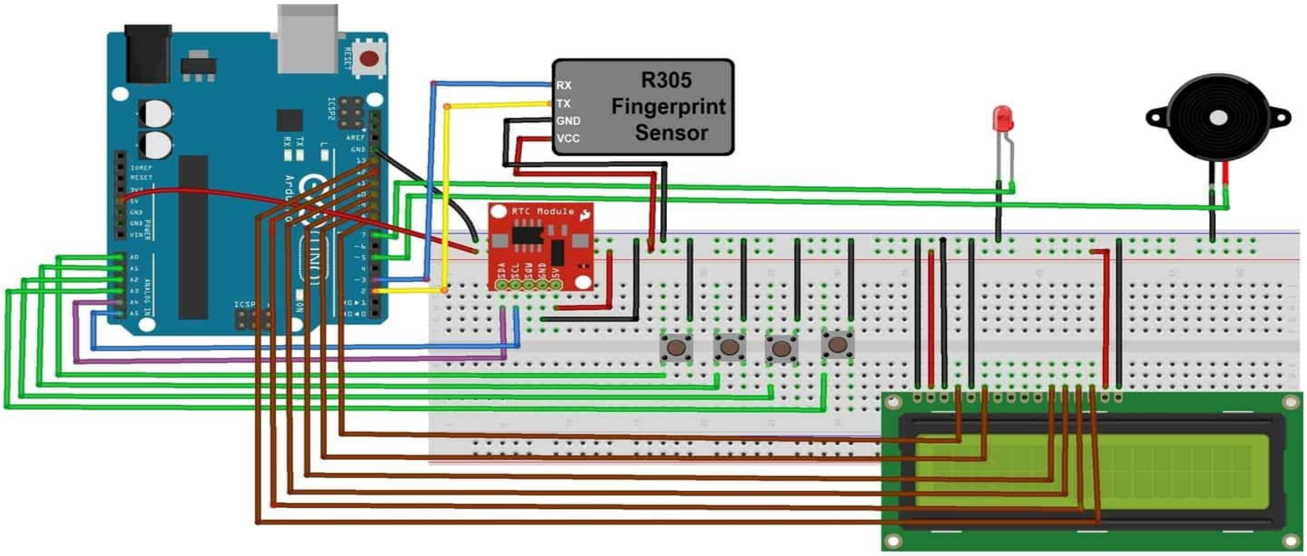


Figure 6 Circuit Diagram

The circuit of this fingerprint-based attendance system has Arduino for controlling all the process of the project, push button for enrolling, deleting, selecting IDs and for attendance, a buzzer for alerting, LEDs for indication and LCD to instruct user and showing the resultant messages. A push button is directly connected to pin A0(ENROL), A1(DEL), A2(UP), A3(DOWN) of Arduino with respect to the ground And Yellow LED is connected at Digital pin D7 of Arduino with respect to ground. Fingerprint module's Rx and Tx directly connected at Serial pin D2 and D3 of Arduino. 5v supply is used for powering finger print module taken from Arduino board. A buzzer is also connected at pin A5. A 16x2 LCD is configured in 4-bit mode and its RS, EN, D4, D5, D6, and D7 are directly connected at Digital pin D13, D12, D11, D10, D9, and D8 of Arduino.

WORKING EXPLAINATION

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.

SOURCE CODE PROGRAM

Header Files Used:

Adafruit_Fingerprint.h -------------------------------------- fingerprint library

RTClib.h --- library file for RTC Module

EEPROM.h -- command for storing data

LiquidCrystal.h --- LCD header file

The program is coded as to register 4 Fingerprints and can record each user’s attendance data for 30 times or Days, and every attendance will record time and date so this becomes 7-byte data.

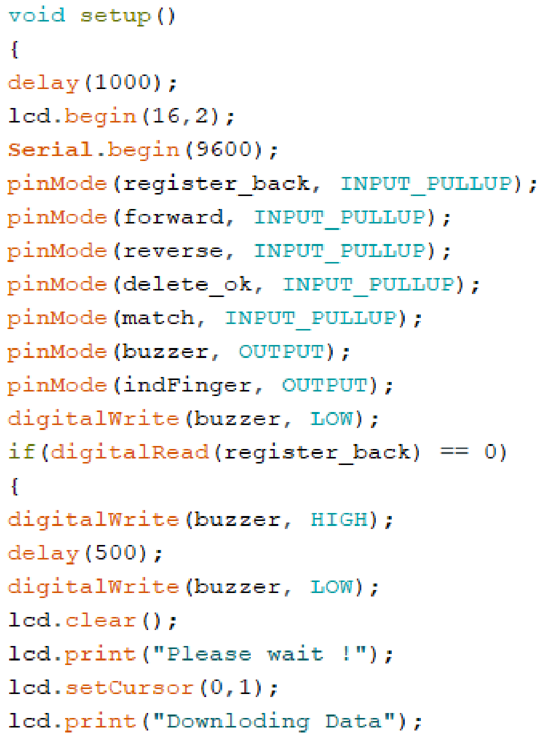
So total memory required is:

4*30*7=840 bytes.

CODE EXPLAINATION

First of all, we include the header file and defines input and output pin and define the macro and declared variables. After this, in setup function, we give direction to defined pin and initiate LCD and finger print module

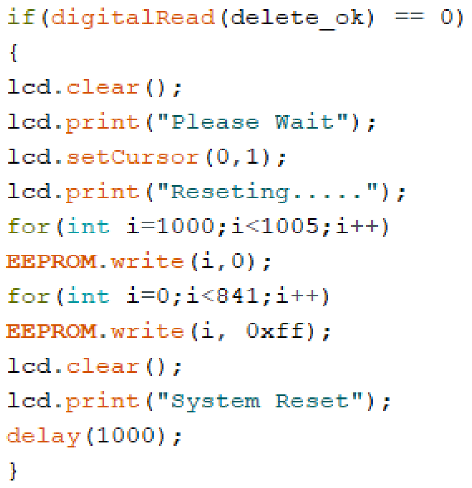
After it, we have to write code for downloading attendance data.



```
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");
```

Figure 7 Code for Downloading Attendance Data

After it, we have to write code for clearing attendance data from EEPROM.

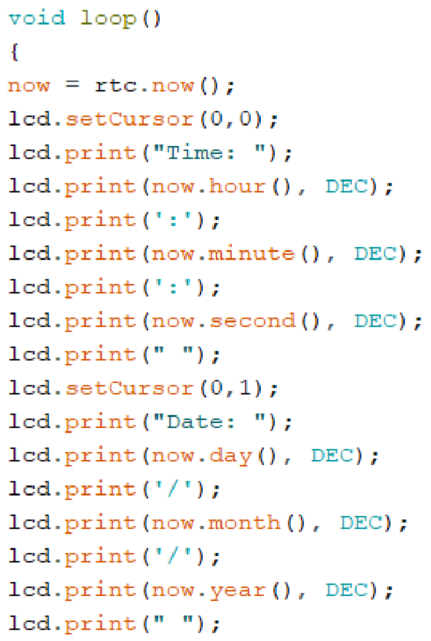


```
if(digitalRead(delete_ok) == 0)
{
lcd.clear();
lcd.print("Please Wait");
lcd.setCursor(0,1);
lcd.print("Reseting.....");
for(int i=1000;i<1005;i++)
EEPROM.write(i,0);
for(int i=0;i<841;i++)
EEPROM.write(i, 0xff);
lcd.clear();
lcd.print("System Reset");
delay(1000);
}
```

Figure 8 Code for Clearing Attendance Data from EEPROM

After it, we initiate finger print module, showing welcome message over LCD and also initiated RTC module.

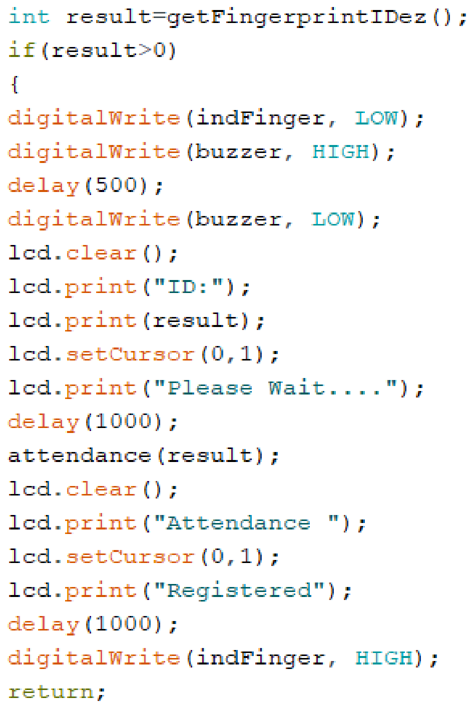
After it, in loop function, we have read RTC time and displayed it on LCD



```
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(" ");
```

Figure 9 Code to Read RTC and Display Time

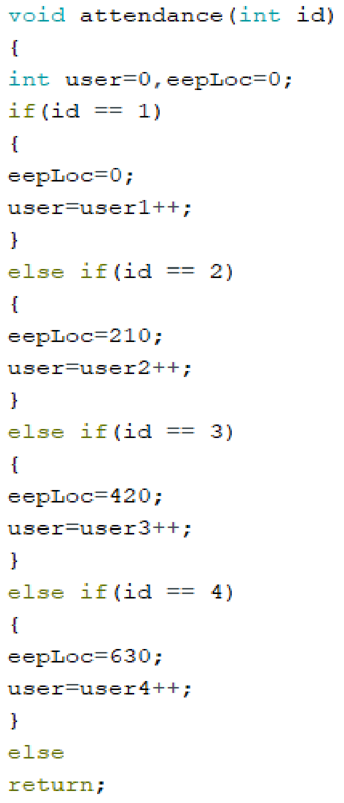
After it, waiting for the finger print to take input and compare captured image ID with stored IDs. If a match occurs then proceed with next step. And checking enroll del keys as well



```
int result=getFingerprintIDez();
if(result>0)
{
digitalWrite(indFinger, LOW);
digitalWrite(buzzer, HIGH);
delay(500);
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;
```

Figure 10 Code to Compare Fingerprints

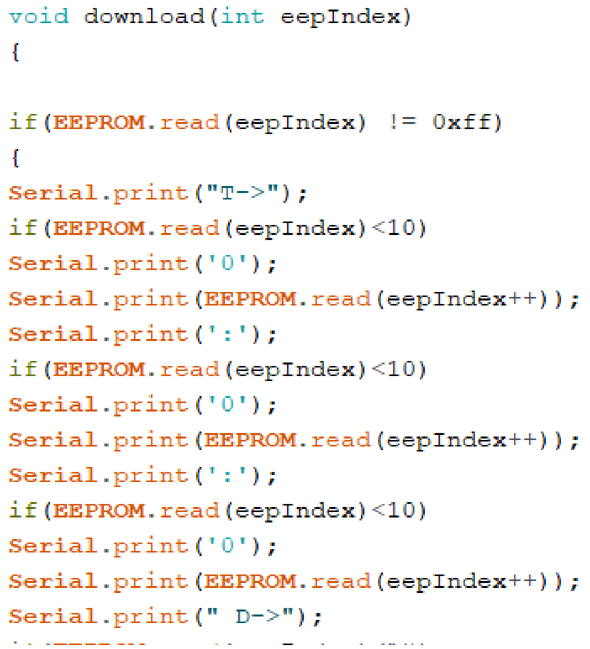
Given function is used for storing attendance time and date in the allotted slot of EEPROM



```
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
return;
```

Figure 11 Code to Store Attendance Time and Date

Given function is used to fetching data from EEPROM and send to serial monitor



```
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->");
```

Figure 12 Code to Fetch Data from EEPROM

COMPLETE CODE

```
#include "Adafruit_Fingerprint.h"
#include<EEPROM.h>
#include<LiquidCrystal.h>
LiquidCrystal lcd(8,9,10,11,12,13);
#include <SoftwareSerial.h>
SoftwareSerial fingerPrint(2, 3);
#include <Wire.h>
#include "RTClib.h"
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 4
int user1,user2,user3,user4;
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(F("Please wait"));
Serial.println(F("Downloding Data.."));
Serial.println();

Serial.print("S.No. ");
for(int i=0;i<records;i++)
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);

Serial.println();

}

}

if(digitalRead(delete_ok) == 0)

{

lcd.clear();

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

```
delay(3000);

user1=EEPROM.read(1000);

user2=EEPROM.read(1001);

user3=EEPROM.read(1002);

user4=EEPROM.read(1003);

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);
```

```
Serial.begin(9600);
lcd.clear();
lcd.print("Finding Module..");
lcd.setCursor(0,1);
delay(2000);
if (finger.verifyPassword())
{
Serial.println(F("Found fingerprint sensor!"));
lcd.clear();
lcd.print(" Module Found");
delay(2000);
}
else
{
Serial.println(F("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(F("Couldn't find RTC"));
rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
lcd.setCursor(0,0);
lcd.print(" Press Match to ");
lcd.setCursor(0,1);
lcd.print(" Start System");
```

```
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
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());
```

```
int result=getFingerprintIDez();
if(result>0)
{
digitalWrite(indFinger, LOW);
digitalWrite(buzzer, HIGH);
delay(500);
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);
}
void attendance(int id)
{
int user=0,eepLoc=0;
```

```
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--;
);
}
else if(digitalRead(delete_ok) == 0)
{
id=count;
getFingerprintEnroll();
for(int i=0;i<records;i++)
{
if(EEPROM.read(i) != 0xff)
```

```
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);
}
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()
{
```

```
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;
}
```

```
{
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;
```

```
lcd.print("Comm Error");
break;
case FINGERPRINT_IMAGEFAIL:
Serial.println(F("Imaging error"));
lcd.clear();
lcd.print("Imaging Error");
break;
default:
Serial.println(F("Unknown error"));
lcd.clear();
lcd.print("Unknown Error");
break;
}
}
p = finger.image2Tz(1);
switch (p) {
:
Serial.println(F("Image too messy"));
lcd.clear();
lcd.print("Image too messy");
return p;
case FINGERPRINT_PACKETRECIEVEERR:
Serial.println(F("Communication error"));
p = finger.storeModel(id);
if (p == FINGERPRINT_OK) {
Serial.println(F("Stored!"));
```

```
}
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(F("Image taken"));
lcd.clear();
lcd.print("Image taken");
break;
case FINGERPRINT_NOFINGER:
Serial.println(F("No Finger"));
lcd.clear();
lcd.print("No Finger Found");
break;
case FINGERPRINT_PACKETRECIEVEERR:
Serial.println(F("Communication error"));`
lcd.clear();
```

```
p = -1;
Serial.println(F("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(F("Image taken"));
break;
case FINGERPRINT_NOFINGER:
Serial.print(".");
break;
case FINGERPRINT_PACKETRECIEVEERR:
Serial.println(F("Communication error"));
break;
case FINGERPRINT_IMAGEFAIL:
Serial.println(F("Imaging error"));
break;
default:
Serial.println(F("Unknown error"));
return p;
}
}
p = finger.image2Tz(2);
switch (p) {
```

```
lcd.clear();

lcd.print("Comm Error");

return p;

case FINGERPRINT_FEATUREFAIL:

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

lcd.clear();

lcd.print("Feature Not Found");

return p;

case FINGERPRINT_INVALIDIMAGE:

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

lcd.clear();

lcd.print("Feature Not Found");

return p;

default:

Serial.println(F("Unknown error"));

lcd.clear();

lcd.print("Unknown Error");

return p;

}

Serial.println(F("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);

return p;

} else {

Serial.println(F("Unknown error"));

return p;

}

Serial.print("ID "); Serial.println(id);

p = finger.storeModel(id);

if (p == FINGERPRINT_OK) {

Serial.println(F("Stored!"));

lcd.clear();

lcd.print(" Finger Stored!");

delay(2000);

} else if (p == FINGERPRINT_PACKETRECIEVEERR) {

Serial.println(F("Communication error"));

return p;

} else if (p == FINGERPRINT_BADLOCATION) {

Serial.println(F("Could not store in that location"));

return p;

} else if (p == FINGERPRINT_FLASHERR) {

Serial.println(F("Error writing to flash"));

return p;

}

else {

Serial.println(F("Unknown error"));

return p;

}

}

int getFingerprintIDez()
```

```
case FINGERPRINT_OK:
Serial.println(F("Image converted"));
break;
case FINGERPRINT_IMAGEMESS:
Serial.println(F("Image too messy"));
return p;
case FINGERPRINT_PACKETRECIEVEERR:
Serial.println(F("Communication error"));
return p;
case FINGERPRINT_FEATUREFAIL:
Serial.println(F("Could not find fingerprint features"));
return p;
case FINGERPRINT_INVALIDIMAGE:
Serial.println(F("Could not find fingerprint features"));
return p;
default:
Serial.println(F("Unknown error"));
return p;
}
Serial.print("Creating model for #"); Serial.println(id);
p = finger.createModel();
if (p == FINGERPRINT_OK) {
Serial.println(F("Prints matched!"));
} else if (p == FINGERPRINT_PACKETRECIEVEERR) {
Serial.println(F("Communication error"));
return p;
} else if (p == FINGERPRINT_ENROLLMISMATCH) {
Serial.println(F("Fingerprints did not match"));
```

```
if (p == FINGERPRINT_OK)
{
Serial.println(F("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++));
```

```
{
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;
}
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);
```

```
p = -1;
Serial.println(F("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(F("Image taken"));
break;
case FINGERPRINT_NOFINGER:
Serial.print(".");
break;
case FINGERPRINT_OK:
Serial.println(F("Image converted"));
lcd.clear();
lcd.print("Image converted");
break;
case FINGERPRINT_IMAGEMESS
case FINGERPRINT_PACKETRECIEVEERR:
Serial.println(F("Communication error"));
break;
}
}
p = finger.image2Tz(2);
```

```
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("         ");

}
```

RESULT ANALYSIS CONCLUSION

To show the working of our project and result we got, we made two videos. But, as we cannot include the video in this report so we have included some screenshots of those videos.

FIRST VIDEO

First video is to show the Registering of Fingerprint in the module. Video will show just one register but we added 3 fingerprints for the demo.

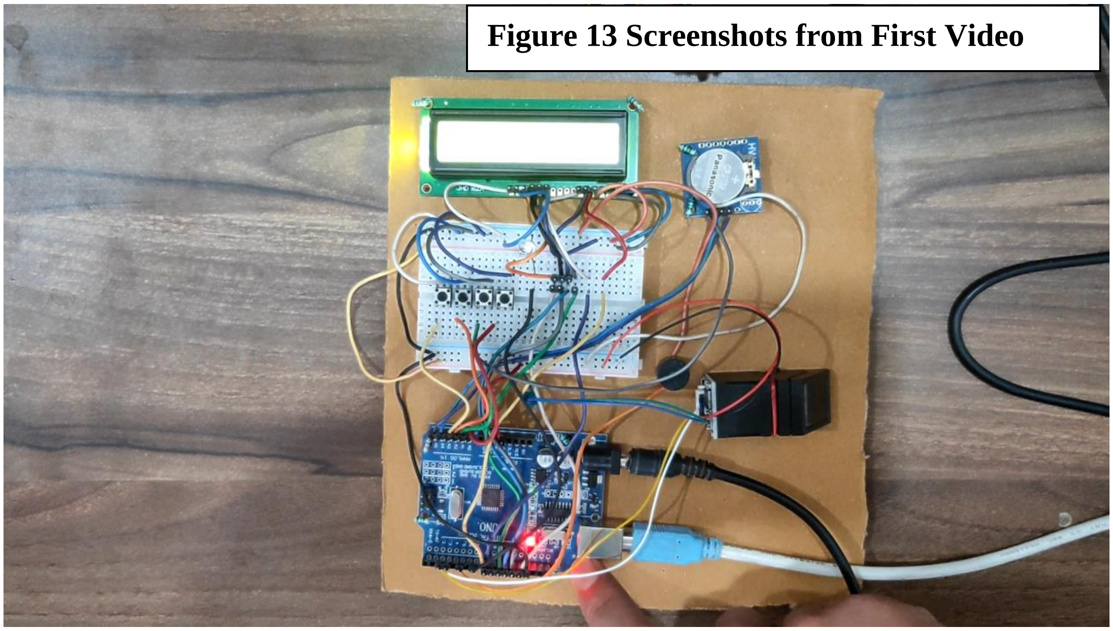


Figure 13 Screenshots from First Video

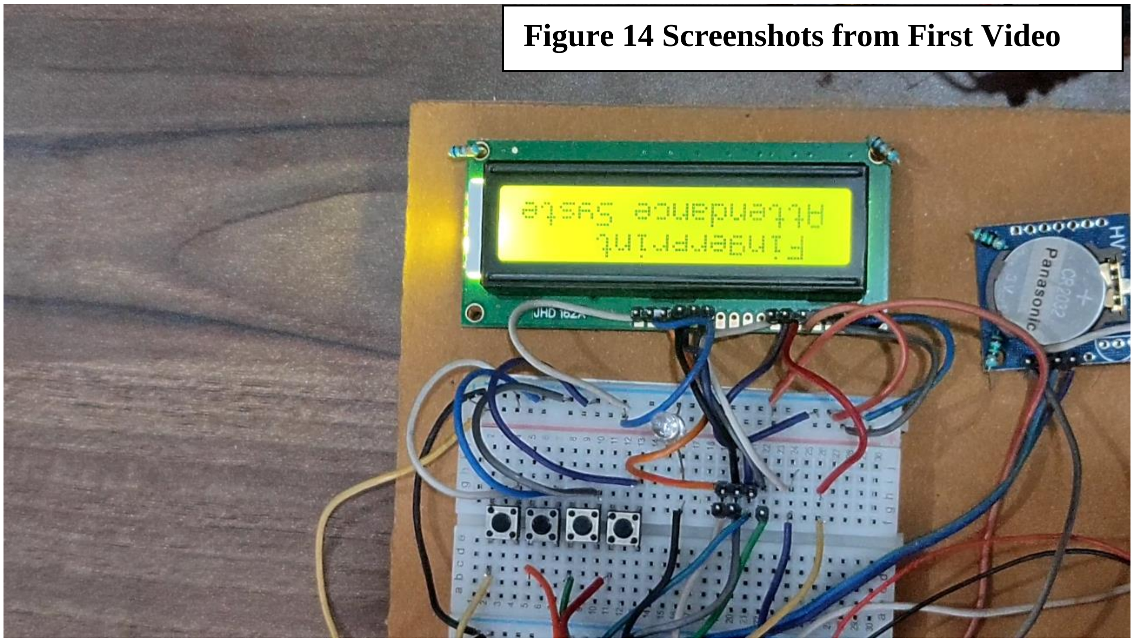


Figure 14 Screenshots from First Video

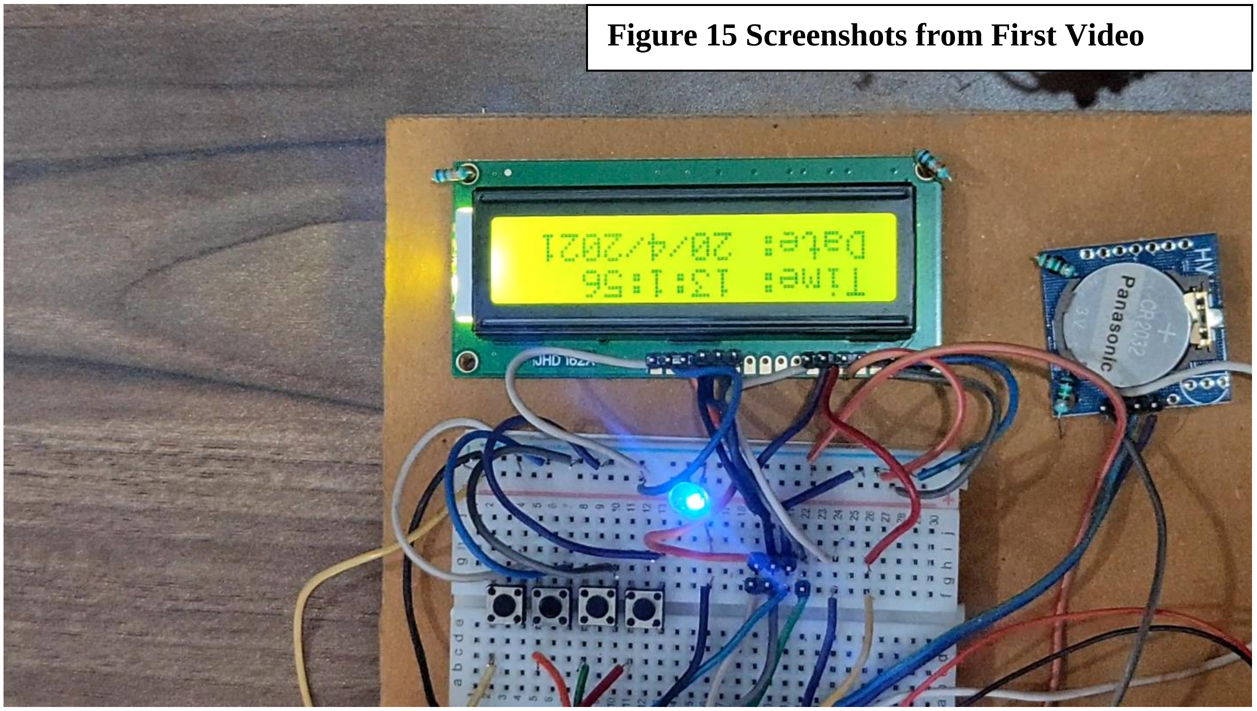


Figure 15 Screenshots from First Video

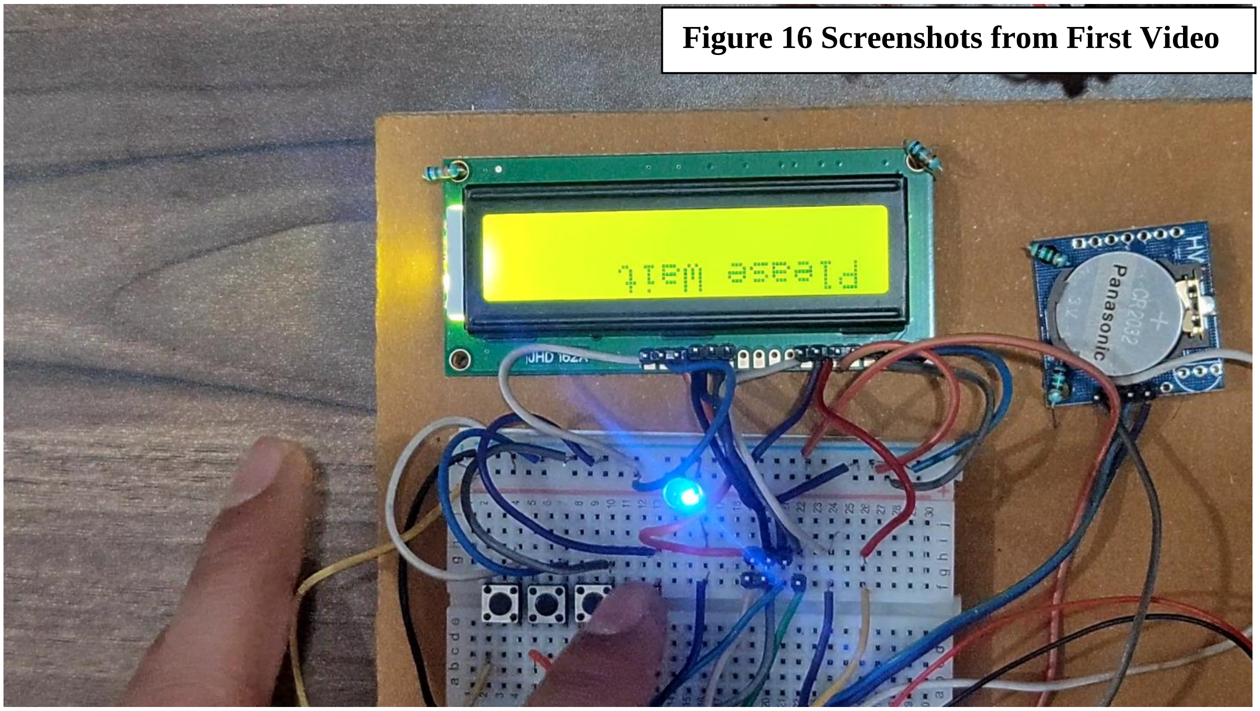


Figure 16 Screenshots from First Video

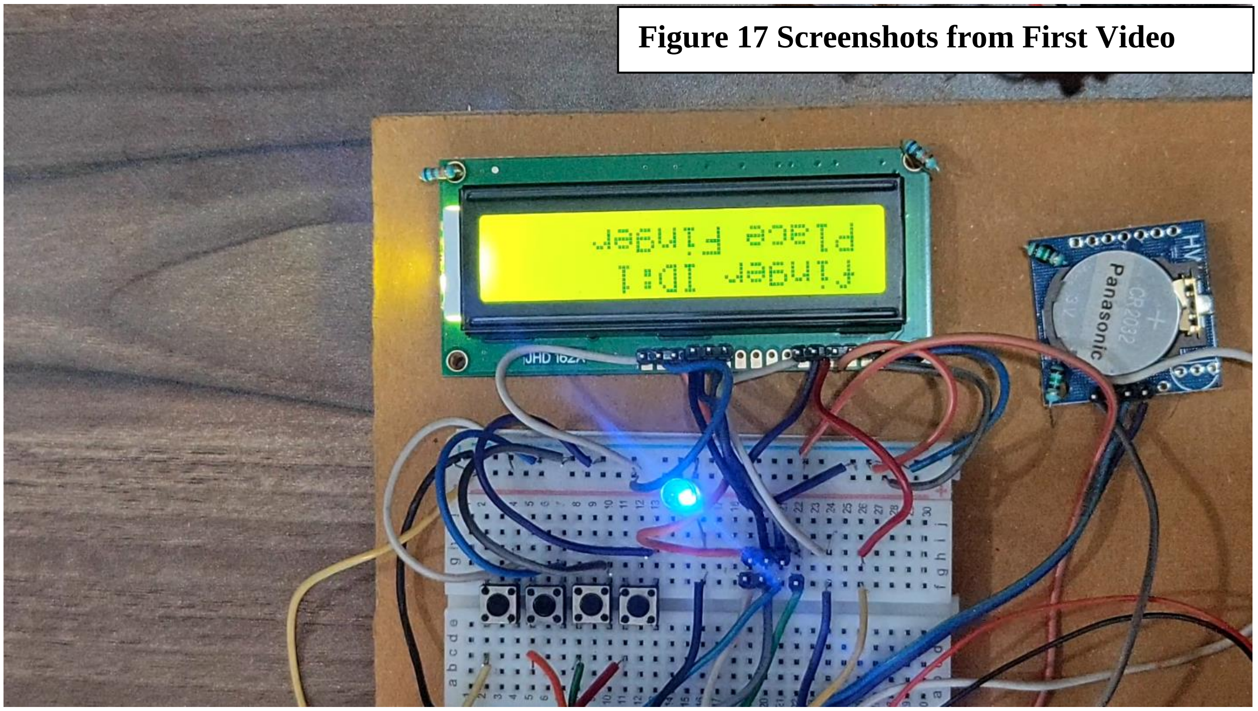


Figure 17 Screenshots from First Video

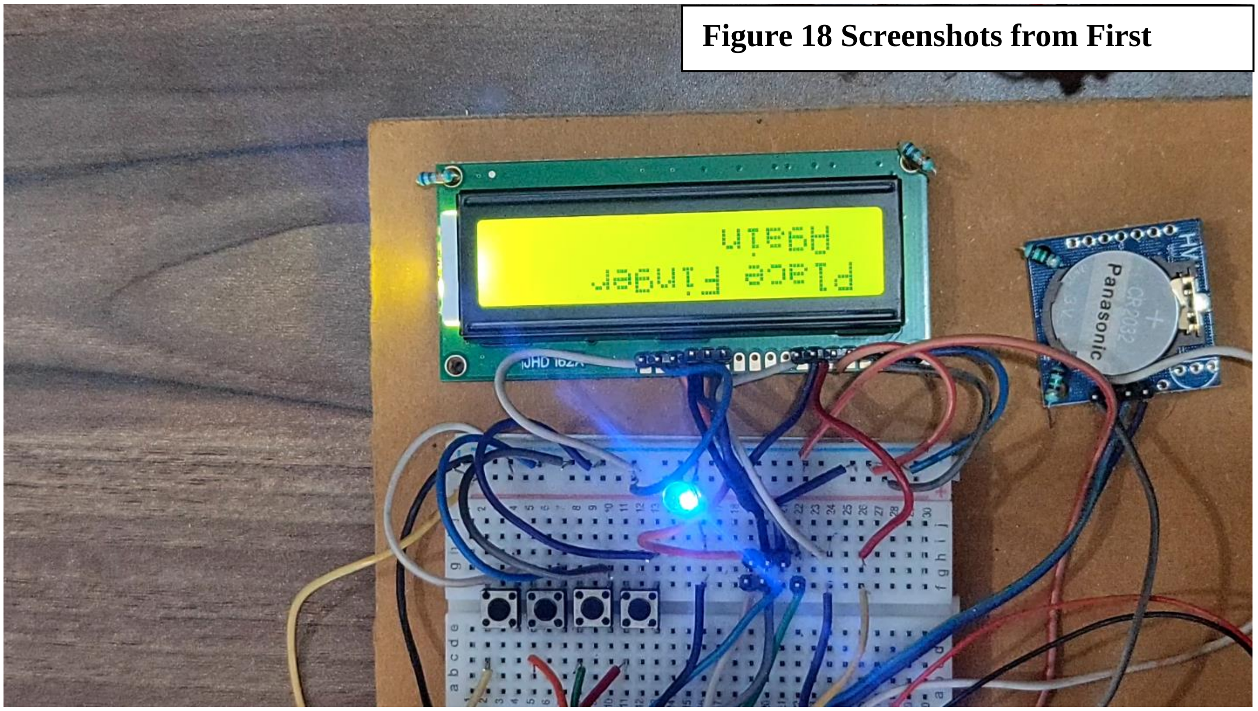


Figure 18 Screenshots from First

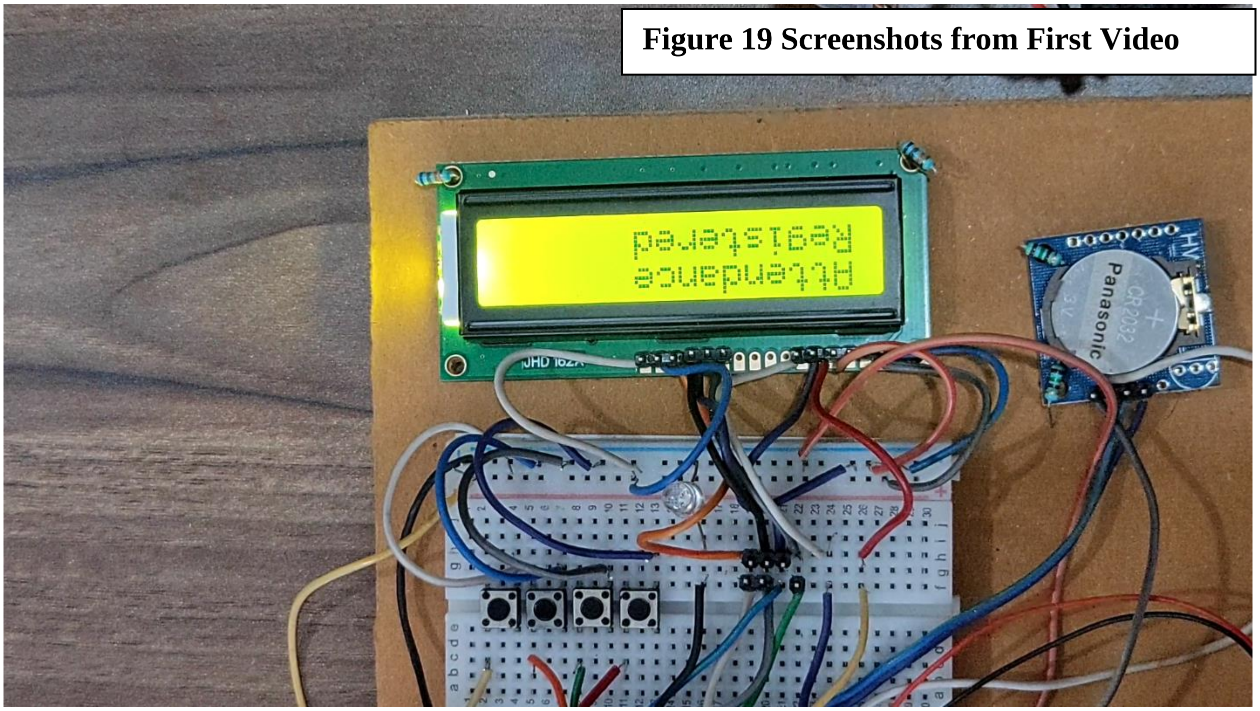


Figure 19 Screenshots from First Video

SECOND VIDEO

Second video shows the actual attendance being recorded by module. As I put my different finger on the Fingerprint Scanner, Arduino registers the attendance.

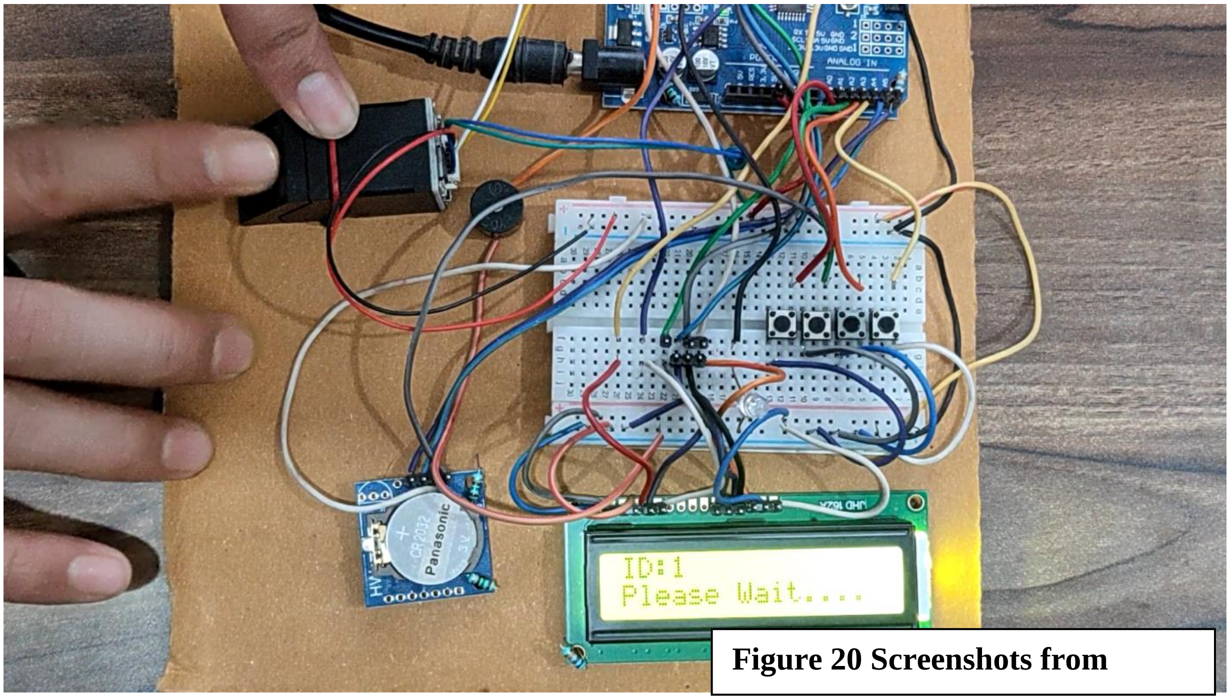


Figure 20 Screenshots from

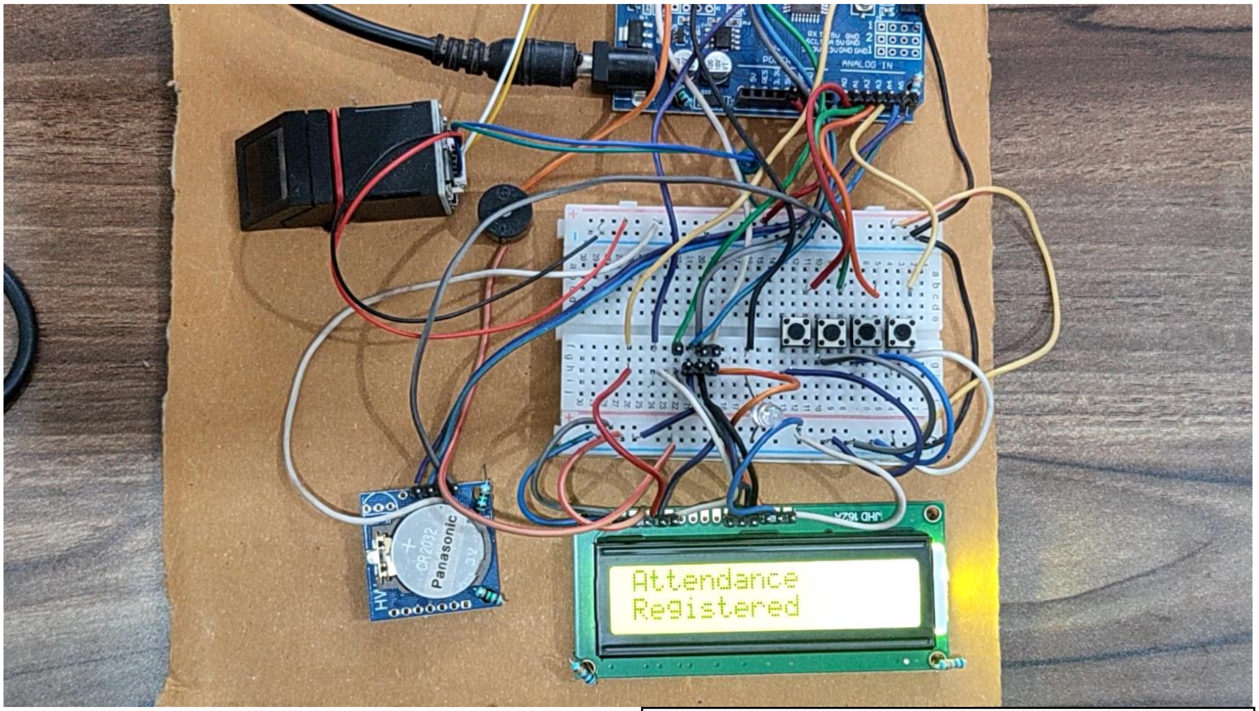


Figure 21 Screenshots from Second Video

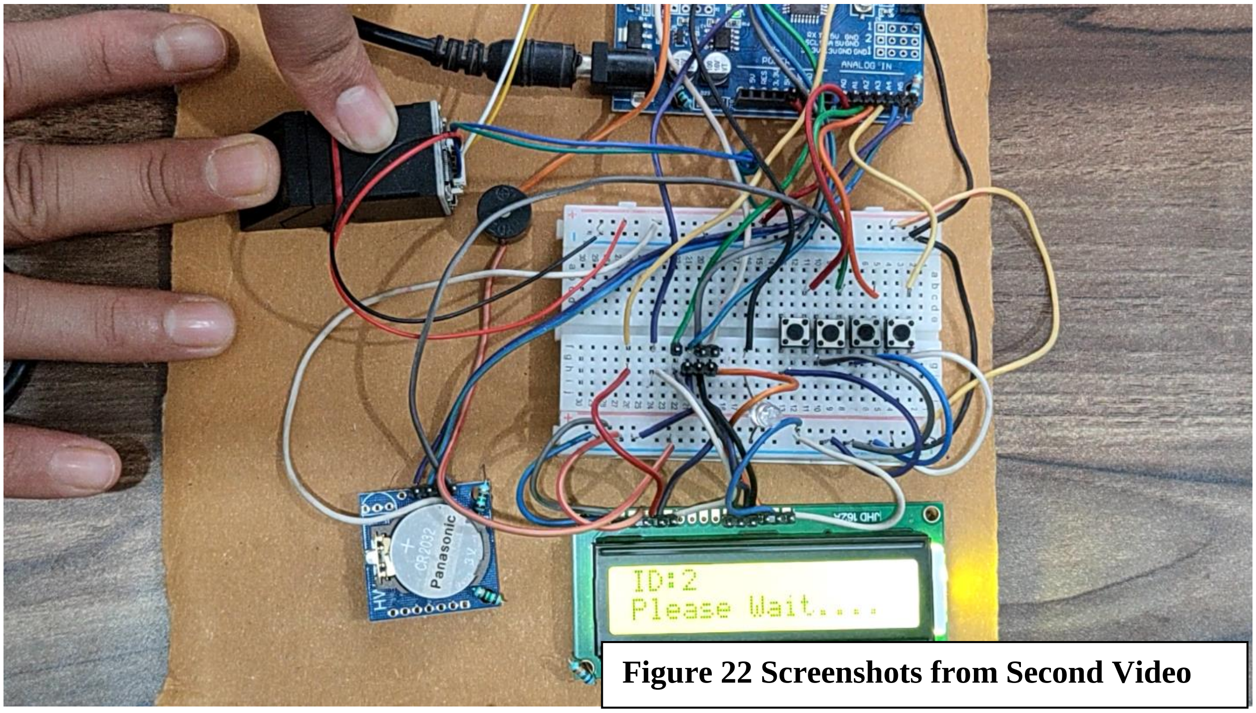


Figure 22 Screenshots from Second Video

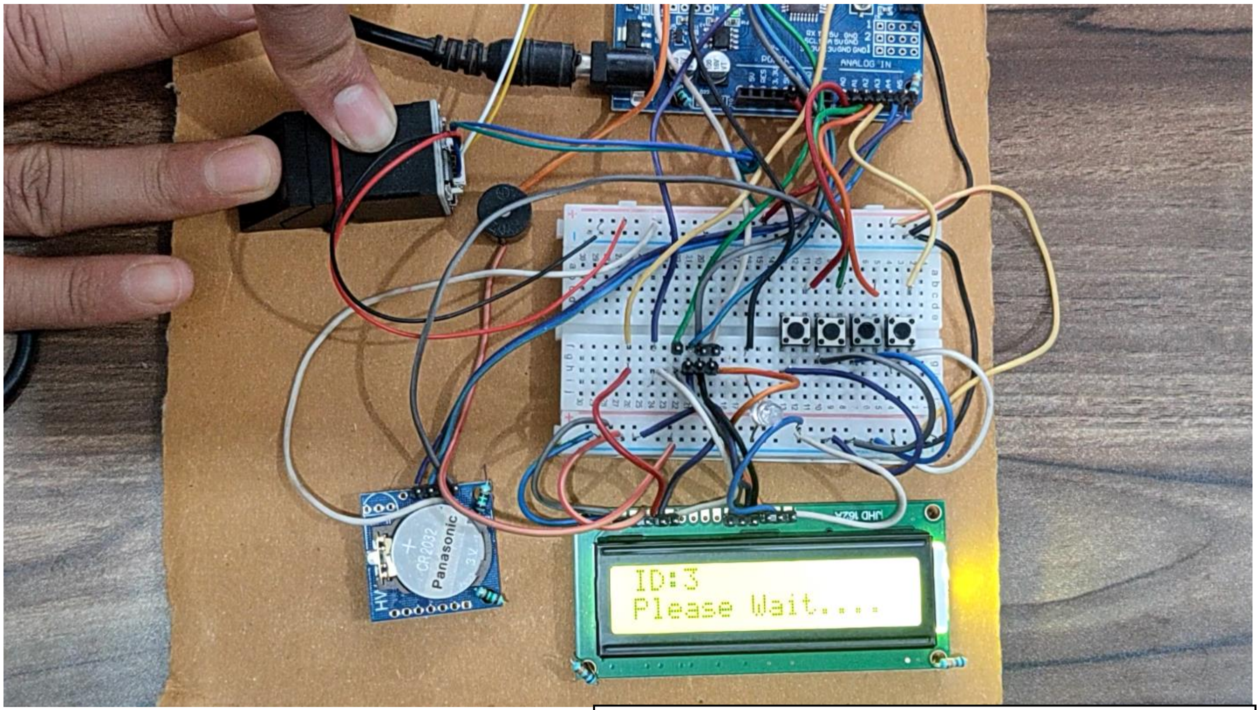
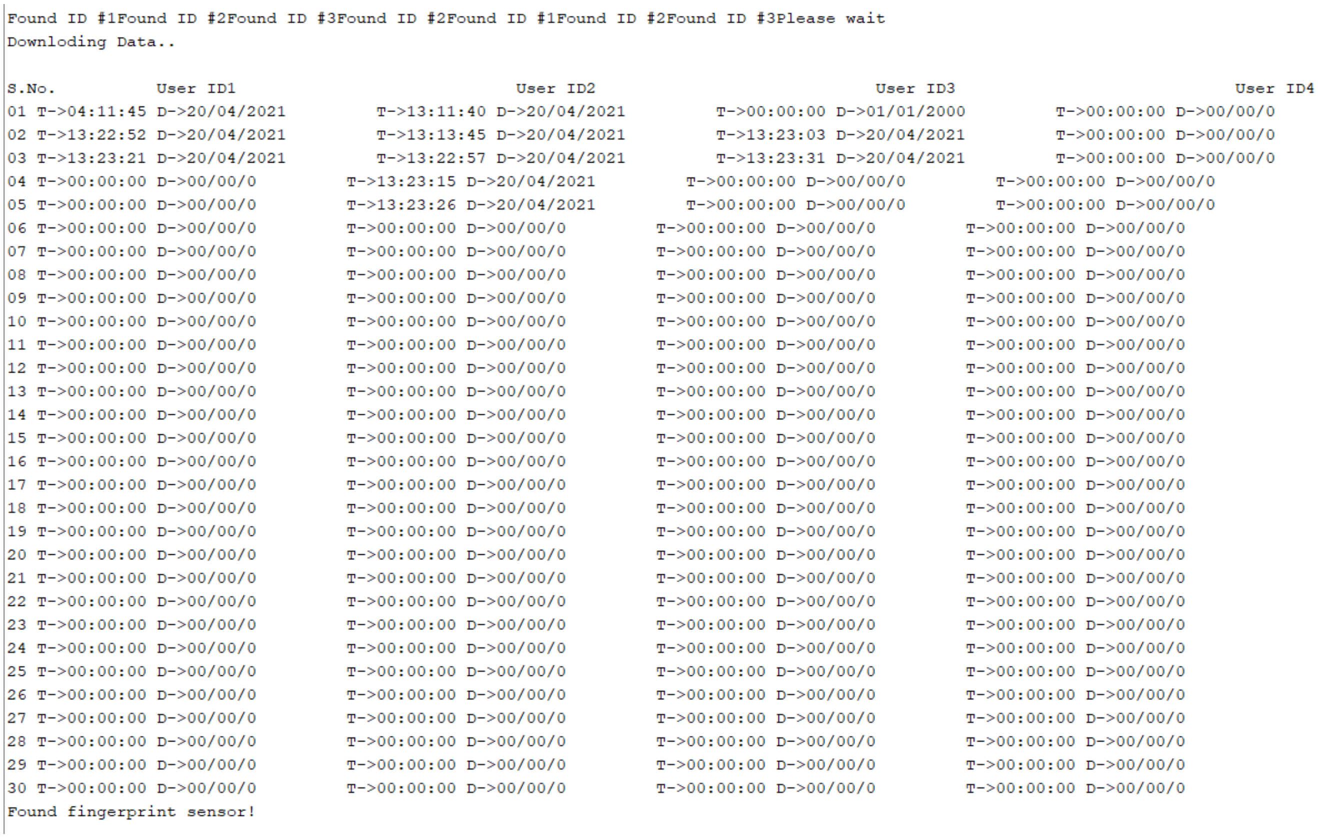


Figure 23 Screenshots from Second Video

And finally, at last of second video the Database of attendance was downloaded by pressing the RESET key followed by INSERT key (1st key).

DATABASE



```
Found ID #1Found ID #2Found ID #3Found ID #2Found ID #1Found ID #2Found ID #3Please wait
Downloding Data..

S.No.          User ID1                            User ID2                            User ID3                            User ID4
01 T->04:11:45 D->20/04/2021         T->13:11:40 D->20/04/2021         T->00:00:00 D->01/01/2000         T->00:00:00 D->00/00/0
02 T->13:22:52 D->20/04/2021         T->13:13:45 D->20/04/2021         T->13:23:03 D->20/04/2021         T->00:00:00 D->00/00/0
03 T->13:23:21 D->20/04/2021         T->13:22:57 D->20/04/2021         T->13:23:31 D->20/04/2021         T->00:00:00 D->00/00/0
04 T->00:00:00 D->00/00/0         T->13:23:15 D->20/04/2021         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
05 T->00:00:00 D->00/00/0         T->13:23:26 D->20/04/2021         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
06 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
07 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
08 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
09 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
10 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
11 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
12 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
13 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
14 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
15 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
16 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
17 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
18 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
19 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
20 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
21 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
22 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
23 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
24 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
25 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
26 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
27 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
28 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
29 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
30 T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0         T->00:00:00 D->00/00/0
Found fingerprint sensor!
```

Figure 24 Database Displayed on Monitor

CONCLUSION

Hardware implementation was quite a bit different experience from simulation. To achieve the output, we wanted in this project we gave multiple try, because it was not that easy to work with hardware just like simulation.

In the first attempt, the LCD was not able to display any output, may be due to improper connection. Then we tried reconnecting every connection from Beginning. In the second attempt the Display worked, but not quite as we wanted. The RTC input was not proper and the Registration of fingerprint was also not working. I figured out the use of Resistors was quite complicating, so I tried for the 3rd time without using any Resistors and Finally in the 3rd attempt our finger-print attendance module worked as desired.

The main aim of this project was to work with hardware as well as software and to build a working prototype of figure attendance machine which could be really useful in schools, colleges and offices for maintaining daily attendances. Working with hardware and software together, we understood the working of hardware according to the software commands. Just by making changes in the software, we may be able to convert this project to take the Biometric Voting Machine or Fingerprint Based Security System. The flexibility of this project will allow us to do more with this innovative project in future.

REFERENCES

[1] B. K. P. Mohamed, C V Raghu “Fingerprint attendance system for classroom needs” 2012 Annual IEEE India Conference (INDICON), 7-9 Dec. 2012.

[2] L. Louis, “Working Principle of Arduino and Using it as a Tool for Study and Research” International Journal of Control, Automation, Communication and Systems (IJCACS) July 2018.

[3] M. M.H. Ali, V. H. Mahale, P. Yannawar, A. Gaikwad, “Overview of Fingerprint Recognition System” 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT) March 2016.

[4] V. Badami “Arduino programming for beginners-1“[Hacker Earth].

[5] Tutorials Point Arduino [www.tutorialspoint.com/arduino].