Dibya Dahal

Electronic Fingerprint Voting System

Helsinki Metropolia University of Applied Sciences

Bachelor’s Degree

Electronics Engineering

Bachelor’s Thesis

12.12.2019

Author(s) Title Number of Pages Date	Dibya Dahal Electronic Fingerprint Voting System 27 pages + 14 appendices 12 December 2019
Degree	Bachelor of Engineering
Degree Programme	Electronics Engineering
Instructor(s)	Heikki Valmu, Principal Lecturer

Electoral fraud is common in most countries, especially in countries with weak rule of law. Fair and impartial election can change the way a country or an organization is run. Therefore, the aim of this thesis project was to design a voting system that could aid in fair and swift election. Since every human being has a unique fingerprint, the voting device was based on fingerprint authentication system.

This study reports mainly focuses on developing a prototype of embedded system that deals with Fingerprint voting system. The system integrates different hardware components like Arduino Uno microcontroller, fingerprint module (DY50), LCD (16x2), LEDs, switches, buzzer and resistors that facilitates a flawless voting system.

Firstly, schematic design was made using fritzing application with all connections before translating the project plan into physical prototype. Different components were then integrated using breadboard. After successful integration, the microcontroller was programmed using Arduino Software 1.8.10.
Finally, the system was tested several times where processes like enrolment, authentication, vote submission and result were carried out flawlessly.

Keywords	Fingerprint Module, Arduino Uno, Voting, Election, LCD.

Contents

Appendices

List of Abbreviations

EEPROM	Electrically Erasable Programmable Read Only Memory
LED	Light Emitting Diode
ARM	Advanced RISC Machine
IDE	Integrated Development Environment
ADC	Analog Digital Converter
TTL	Transistor-Transistor Logic
LAN	Local Area Network

1 Introduction

Election is a feature of democratic government in which people govern themselves and are able to express their choices regarding various issues, such as constitutional amendments, piece of legislation or choosing the right person as their leader. An electoral system is present to layout the rules of the election. Political election is the most common form of election but there are many different fields where election is vital part of their organizational functions. Election is vital for business, informal organizations and non-profit organizations.

Election is the way of democratic world but conducting an election in fair manner has been the challenge of every electoral body especially in the country with high corruption, weak rule or law and less transparency. On the top of that conducting election cost millions and billions of dollars.

This study report mainly focuses on developing a prototype of embedded system that deals with fingerprint voting system which can help in progression of election in robust and secure manner. The system integrates different hardware components like microcontroller, fingerprint module, LEDs, switches that facilitates a flawless voting system. For the implementation of this system, DY50 fingerprint sensor is used to take user finger print image and store in internal memory, these images are further processed and analyzed using Arduino. The user interface is implemented using LCD screen, which is mainly used to print user instructions during the execution of the voting process and the result.

2 Objective of the Thesis Project

Tampering with the election process is not uncommon, even in some of the countries with strong rule of law. Ballot stuffing, voter suppression, multiple voting and destruction of legitimately cast ballot still covers the headlines of newspapers.

Since every person has the unique fingerprint in this world, their fingerprint can be used to cast their vote with electronic fingerprint voting system. The proposed system is offline version. The finger-print voting system requires to register their fingerprint at the polling booth. The person can now vote on the Election Day just by verifying their identity using their fingerprint. The system uses Arduino and fingerprint technology. Voting does not only include political but can be social, business and educational as well. A flow diagram of such system is shown in figure 1.

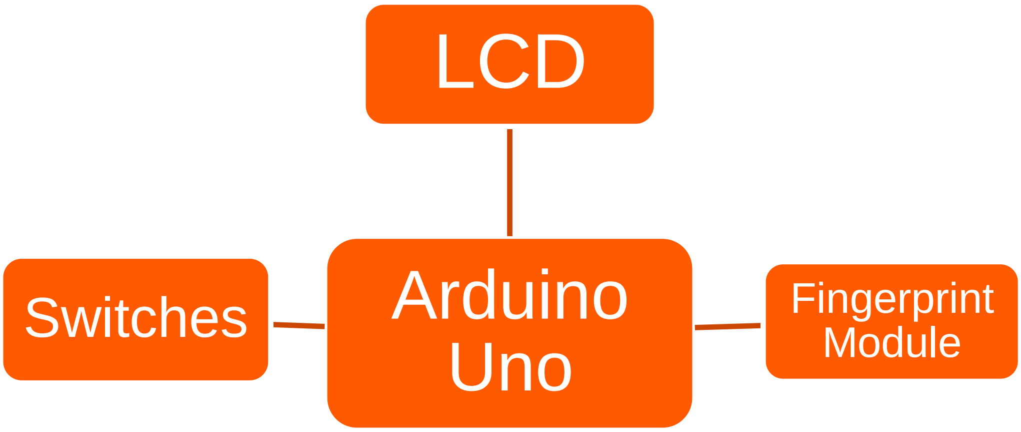


Figure 1. Flow diagram of proposed system.

2.1 Methodology

Methodology as word suggest is the development steps or method that outlines the way to achieve certain goal or in this case design and build a prototype Fingerprint voting system. In other words, it is a constructive framework. Below in figure 2, methodology flow block diagram of this project is shown.

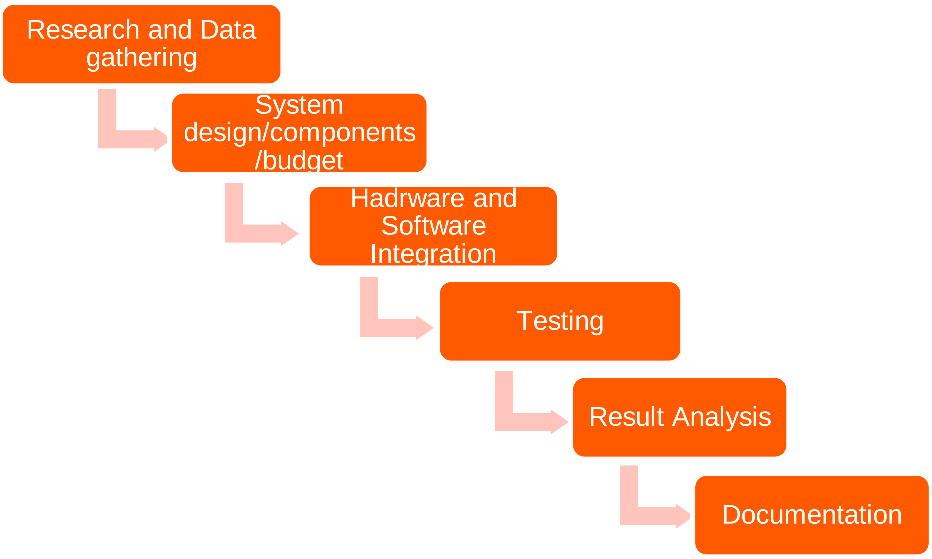

Research and Data gathering
System design/components /budget
Hadrware and Software Integration
Testing
Result Analysis
Documentation

Figure 2. Flow diagram of methodology.

3 Fingerprint Voting System as an Embedded System

The fingerprint voting system is a prototype for an Embedded system. It integrates hardware namely fingerprint sensor (DY50), LCD screen for the display, switches, push buttons for the successful implementation of various inputs and outputs functions. These hardware components are later embedded with processing unit i.e. ARM based microcontroller, Arduino device which is programed in C-programming language to store data, execute different user input commands and produce results based on user inputs. More over microcontroller operates in dynamic way to control all corresponding hardware peripherals.

4 Fingerprint Module

Fingerprint identification is also known as dactyloscopy. Fingerprint identification is the process of comparing two examples of friction ridge skin impression from human fingers, palm or toes [1]. Today fingerprints are considered being one of the oldest and popular among other biometric technologies [2].

The major hardware used in this system comprises of fingerprint sensor as shown in figure 3, also known as biometric identification module. This sensor produces a digital print of the ridges in the skin of the fingers which could be uniquely defined for authentication.
The major application of biometric identification for this study purpose is to read the fingerprint of users so that each image can be uniquely defined and stored in its internal memory. These unique images of fingerprint are assigned with unique numerical values that can be easily be retrieved.

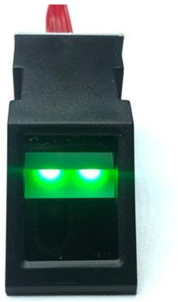


Figure 3. Fingerprint module DY50 [3]

An optical biometric fingerprint (DY50) is used to input and read the fingerprint data. It uses TTL serial and can connect to a microcontroller to send data, detect prints, hash and search. The module has FLASH memory of its own and can store 162 fingerprints in it.

Technical specifications of DY50 is written below in table 1 [4].

Table 1. Technical Specification of DY50.

Supply voltage	3.6 - 6.0VDC
Operating current	120mA max
Peak current	150mA max
Fingerprint imaging time	<1.0 seconds
Signature file	256 bytes
Template file	512 bytes
Storage capacity	162 templates
Safety ratings	(1-5 low to high safety)
Interface	TTL Serial
Baud rate	9600, 19200, 28800, 38400, 57600 (default is 57600)
Working temperature rating	-20C to +50C
Full Dimensions	56 x 20 x 21.5cm

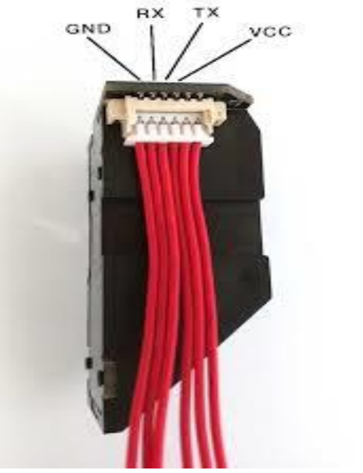


Figure 4. Pin configuration of DY50.

The different pins of fingerprint sensor are shown in figure 4 and these pins are serially connected to Arduino Uno as shown in figure 5 for receiving and transmitting data.

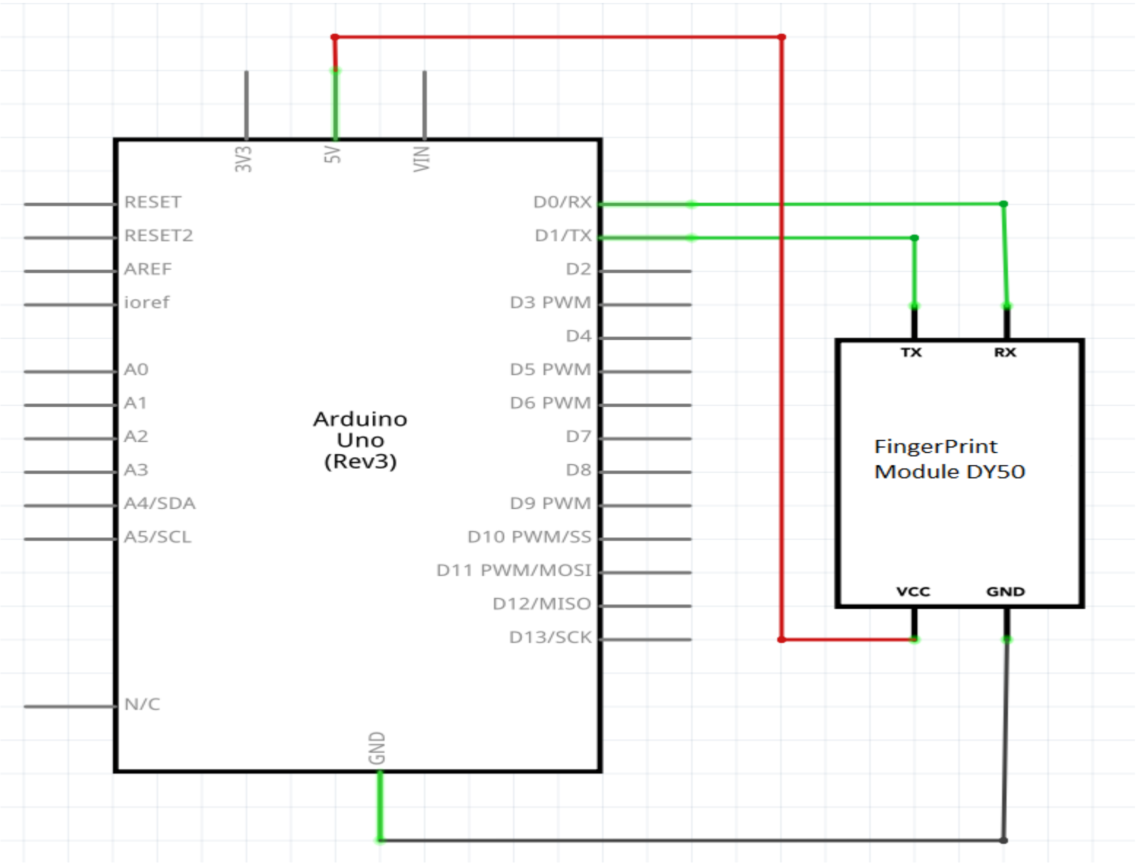


Figure 5. Pin Connection of DY50 with Arduino.

4.1 Extraction Algorithm

Correct representation of the fingerprint image is crucial to automatic fingerprint identification system. Therefore, a fingerprint image goes through various processes like enhancement, analysis, binarizing, thinning and ridge construction before the minutiae is extracted. The process chart of this process can be seen in figure 6.

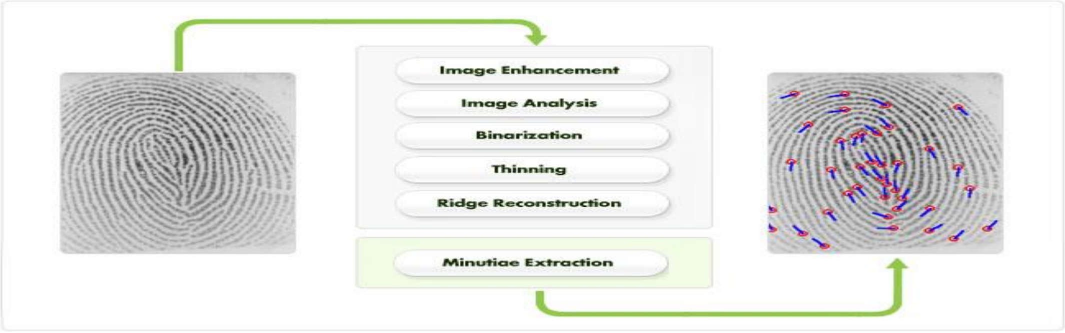


Figure 6. Extraction algorithm with process [3].

4.2 Matching Algorithm

This algorithm detects similarities between two templates extracted by characteristic point extraction algorithm. It is done by comparing the position of each structure and characteristics point as shown in figure 7.

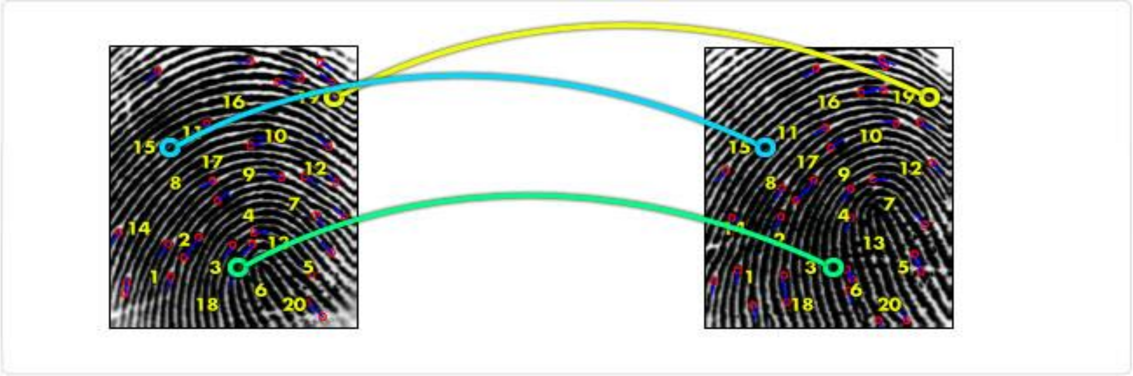


Figure 7. Characteristics matching [3].

5 Microcontroller

Microcontroller is the brain and heart of todays technologically advanced world. Almost every application that is in use today uses microcontroller as its central processing unit. It is heart and brain of the system. There are many kinds of microcontroller available for example Arduino, Teensy, MSP430, etc. Top view of Arduino UNO microcontroller can be seen in figure 8 with at its digital, analog and power pins.

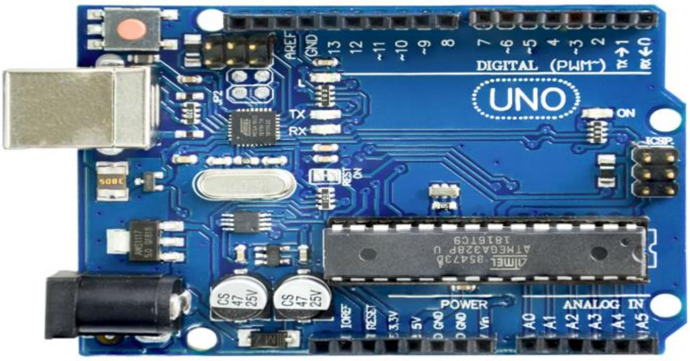


Figure 8. Top view of an Arduino UNO. [5]

This project also uses Arduino microcontroller. It is an electronics platform which is open source platform, easy to use and easily available. Arduino is used as microcontroller kit around the world by students to conduct various electronics interactive projects. Arduino offers different kinds of microcontroller kit. Arduino Uno, Arduino Mega, Arduino BT, Arduino Leonardo are some examples of microcontroller kit. Arduino is preferred by students, programmers, professionals and hobbyists around the world because of its simplicity and availability. Ivrea Interaction Design Institute developed Arduino as a tranquil tool [6].

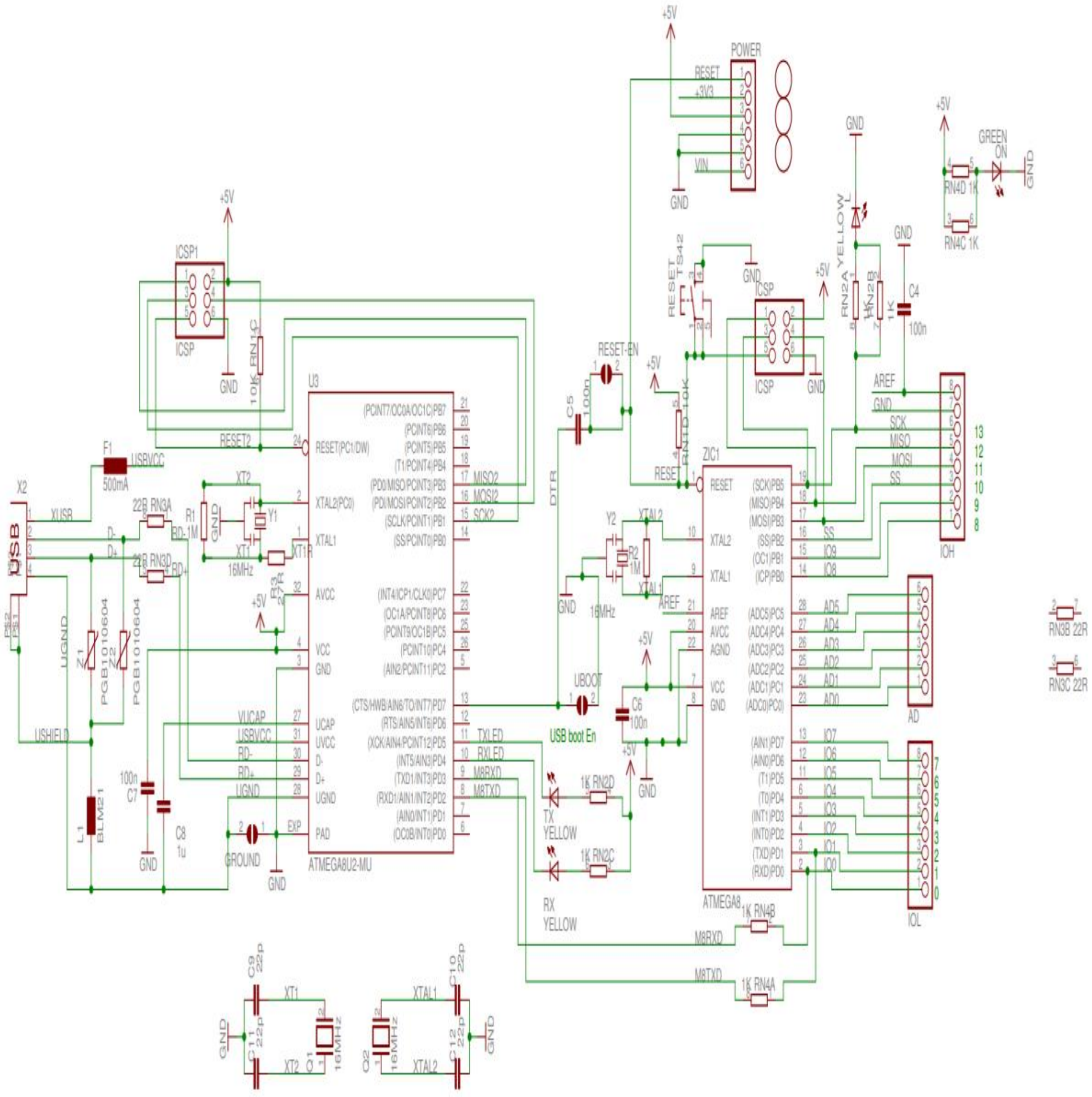


Figure 9. Schematic diagram of Arduino Uno [5].

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board [7].

5.1 Arduino UNO

Arduino Uno platform was chosen for this thesis project because of its pin response speed, simple and clear programming environment, open source and availability. Arduino Uno is a microcontroller board with ATmega328P microprocessor. It comes with 14 digital input/output pins, 6 analog inputs, a 16 MHz quartz crystal, a power jack, a USB connection, an ICP header and a reset button which can be seen in figure 8. Arduino Uno can be powered by battery using power socket or by connecting through A/B USB cable. The official schematic diagram of Arduino UNO can be seen in figure 9. The pins and their function in detail is shown in table 2.

Table 2. The list of pins in UNO and their function [8].

Pin Category	Pins	Description
Power	Vin 3.3V 5V GND	Vin: Input voltage to Arduino using external power source. 5V: Regulated power supply to MC and components. 3.3V: On board voltage regulator generated 3.3 V. GND: Ground pins
Reset	Reset	Resets the Microcontroller
Analog Pins	A0 – A5	Used to provide analog input in the range of 0-5V
Input/Output Pins	Digital pins 0 - 13	Can be used as input or output pins.
Serial	0(Rx), 1(Tx)	Used to receive and transmit TTL serial data.
External Interrupts	2, 3	To trigger an interrupt.
PWM	3,5,6,9,11	Provides 8-bit PWM output.
SPI	10,11,12,13	Used for SPI communication.
Inbuilt LED	13	To turn on the inbuilt LED.
TWI	A4(SDA), A5 (SCA)	Used for TWI communication.
AREF	AREF	To provide reference voltage for input voltage.

The technical specifications of Arduino Uno can be seen in table 3.

Table 3. Technical Specification of Arduino Uno [7].

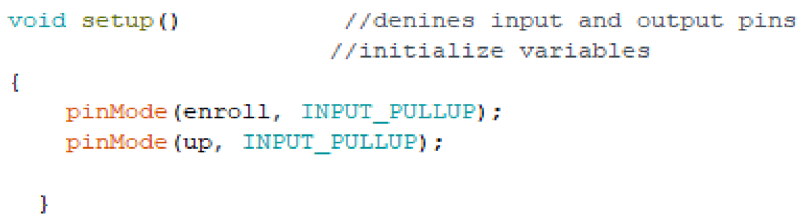
Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz

5.2 Programming

Arduino Uno can be programmed using Arduino software. Arduino IDE can be downloaded and installed freely from https://www.arduino.cc/en/Main/Software. ATMega328 microcontroller in Arduino Uno comes pre burned with bootloader so its easy to equip the programs in the board without the use of the external hardware programmer.

5.2.1 Basics of Arduino Programming

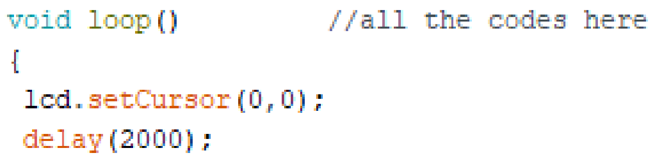
Arduino coding operates with two major blocks. The first one is called void setup. Void setup defines input and output pins. It helps start initializing variables and also start using library. An example of void setup can be seen below in listing 1.



```
void setup()            //denines input and output pins
                        //initialize variables
{
    pinMode(enroll, INPUT_PULLUP);
    pinMode(up, INPUT_PULLUP);

  }
```

Listing 1. Example code of void setup.

The second block is called void loop. Void loop is where all the code and logics are placed in the brackets. Unlike void setup the functions in this loop runs infinitely, restarting after every end until certain condition is met. An example of void loop can be seen below:



```
void loop()           //all the codes here
{
 lcd.setCursor(0,0);
 delay(2000);
```

Listing 2. Example code of void loop.

6 LCD

A liquid crystal display is a combination of states of matter, the liquid and solid. It uses liquid crystal to display visible image on a thin and flat technology display screen. It is widely used in most common electrical and electronic devices like laptop, cell phones, TVs, etc. LCD display operates using two polarizing sheets with liquid crystal solution in between them. The liquid crystal acts like a shutter, when an electric current is passed through the liquid, the crystals aligns so that the light can not pass through them. LCD's do not emit light themselves, so backlight is required to produce visible display.

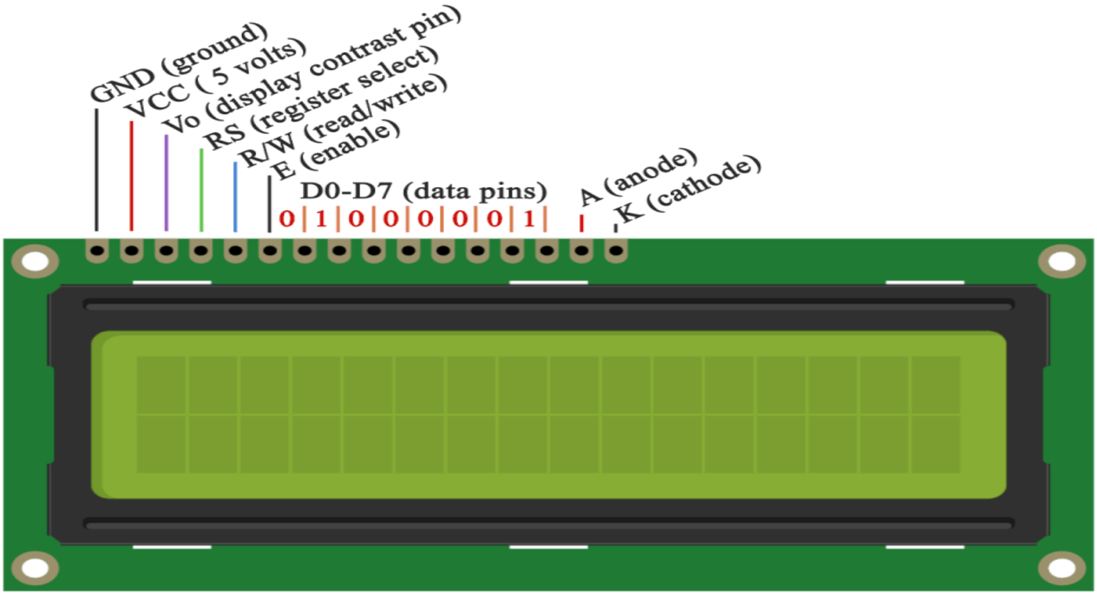


Figure 10. Typical 16×2 LCD Display win pins [9].

A Typical LCD with 16×2 display is shown in figure 10, it is used to create interface between user and microcontroller. It displays messages which guides the user to complete voting processes. It shows when to put your fingerprint on the module, when to press which button, and also whether their vote is accepted or not.

The pin connections between LCD and Arduino Uno can be seen in schematic diagram in figure 11 and are also given below [9]:

- PIN1/VSS to ground
- PIN2/VDD or VCC to +5v power
- PIN3/VEE to ground
- PIN4/RS (Register Selection) to PIN13 of Arduino Uno.
- PIN5/RW to ground (puts LCD in read mode eases the communication for user).
- PIN6/E (Enable) to PIN12 of Arduino Uno.
- PIN11/D4 to PIN11 of Arduino Uno.
- PIN12/D5 to PIN10 of Arduino Uno.
- PIN13/D6 to PIN9 of Arduino Uno.
- PIN14/D7 to PIN8 of Arduino Uno.

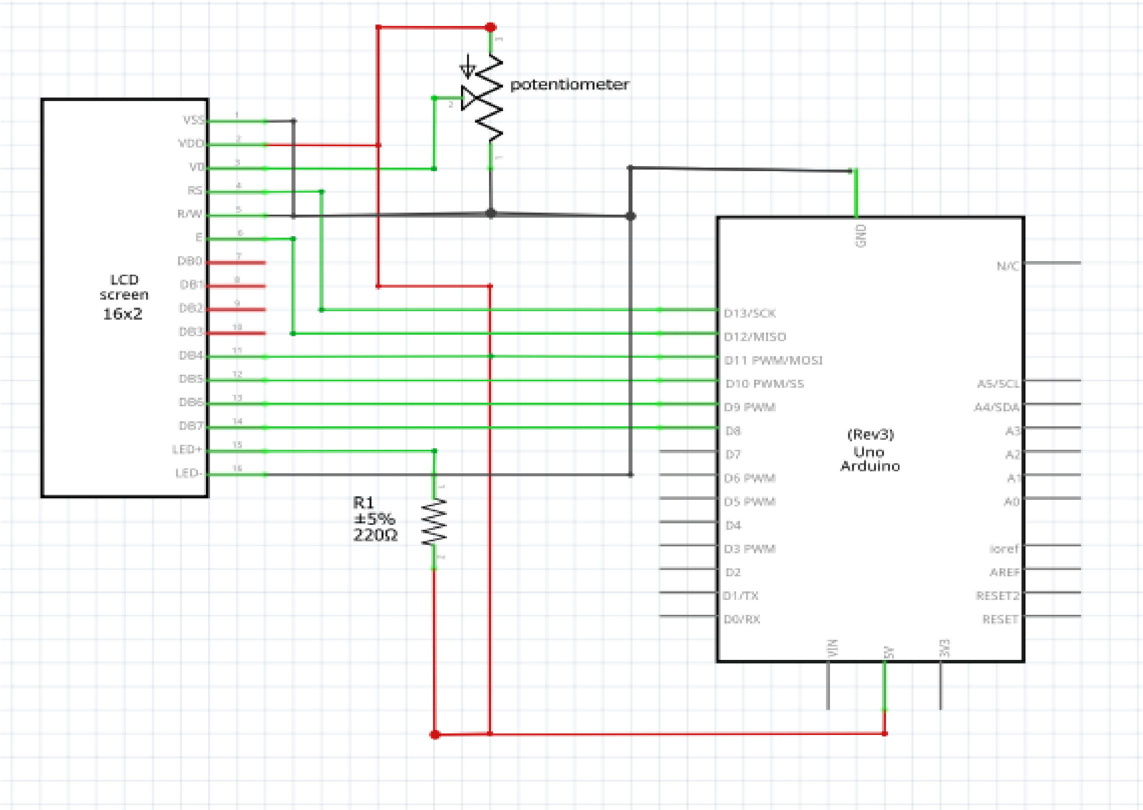


Figure 11. Circuit diagram of 16x2 LCD connected to Arduino Uno.

7 Power Source

Arduino Uno comes with intelligent switching circuitry which ensures the highest voltage is selected and sent to the voltage regulator or board, which eventually power up the board, allowing multiple source of power to be connected simultaneously [10].

Two power supply subsystem is present within Arduino Uno, namely voltage regulator and multiple input management system. Voltage regulator subsystem generates 5V, on which microcontroller runs. The official schematic diagram of voltage regulator subsystem is shown in figure 12.

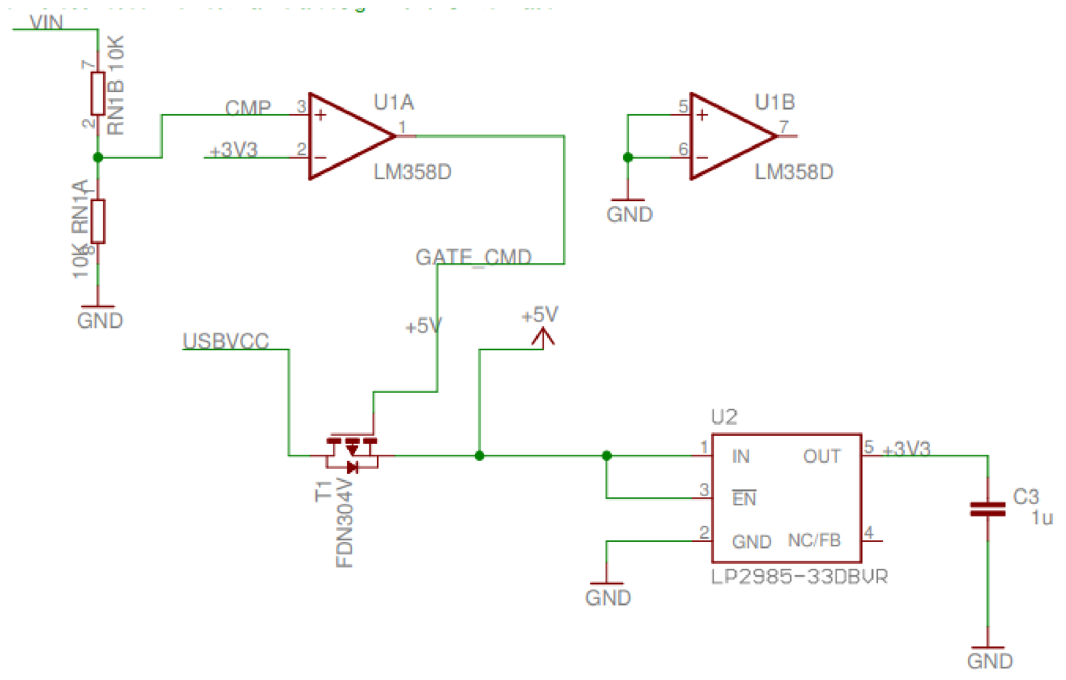


Figure 12. Schematic diagram of voltage regulator system of Arduino Uno.

Multiple input management system is responsible for managing power supply when several inputs are connected to the Arduino. The official schematic diagram of multiple input management system can be seen in figure 13.

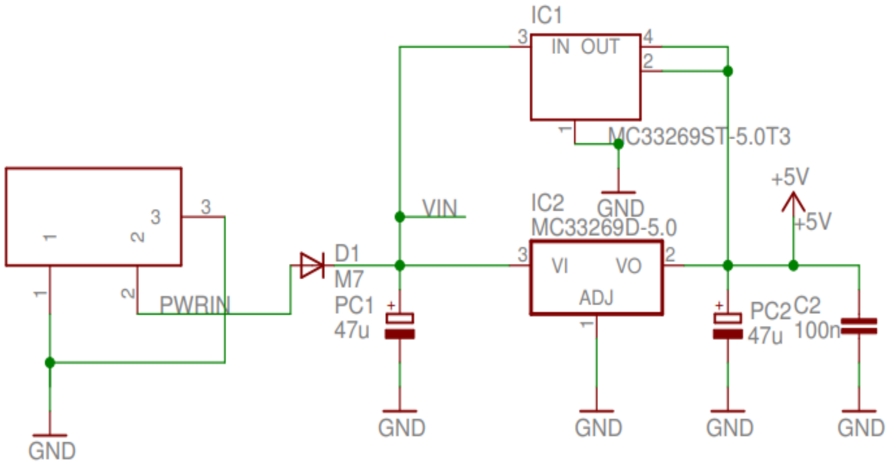


Figure 13. Schematic diagram of multiple input management system of Arduino Uno.

There are four ways to power up the Arduino Uno, which is discussed below:

7.1 Via USB A/B Cable

Arduino Uno comes with the onboard USB port which can be connected to the laptop/desktop using A/B USB cable. Figure 14 shows the pinout of A/B USB port, which has four shielded wires, two (+,-) for power and D+ and D- are used for differential data signals. If the host computer identifies the device, 500mA current at 5V is supplied to the board. If the device is not recognized, 100mA current at 5V is supplied. In case the device is not recognized by the host computer, one can manually install the Arduino Uno. Green power LED labelled as PWR indicates when USB is connected to computer.

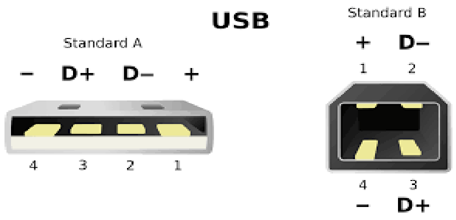


Figure 14. Pins of A/B USB port [11].

Entire prototyping and testing of the system was done in this thesis study by powering Arduino via USB.

7.2 Via Barrel Connector

Another way to power up Arduino Uno is by using an AC to DC adapter plugged into the barrel connector. Onboard voltage regulator regulates the input voltage fed through the barrel connector and power on the device. The input voltage that can be supplied to barrel connector is 7-12V.

7.3 Control Via Steady 5V Input

Arduino Uno can also be powered using the 5V and ground pins. But the input voltage has to be steady and regulated at 5V. However, the 5V pin on board bypasses the voltage regulator and all other safety features of Arduino Uno. So, it is very important not to exceed input voltage over 5V. Otherwise the Arduino board can damage. The maximum upper limit input voltage of Arduino Uno is 5.5V.

7.4 Batteries

A battery can be used power up the Uno board. The Vin pin can be connected with the positive terminal of 7-12V battery and the ground pin connected to the negative terminal of the battery. Input voltage can vary from 7-12V but the value of current must be around 500mA.

8 Hardware Implementation of Embedded System

In order to build a working prototype system, all the selected components should be integrated. Before starting the hardware integration, the schematic circuit was created which can be seen in figure 15. The schematic diagram helps to understand and visualize the connections between components. The schematic diagram was made using Fritzing application which can be downloaded freely from https://fritzing.org/download/ .

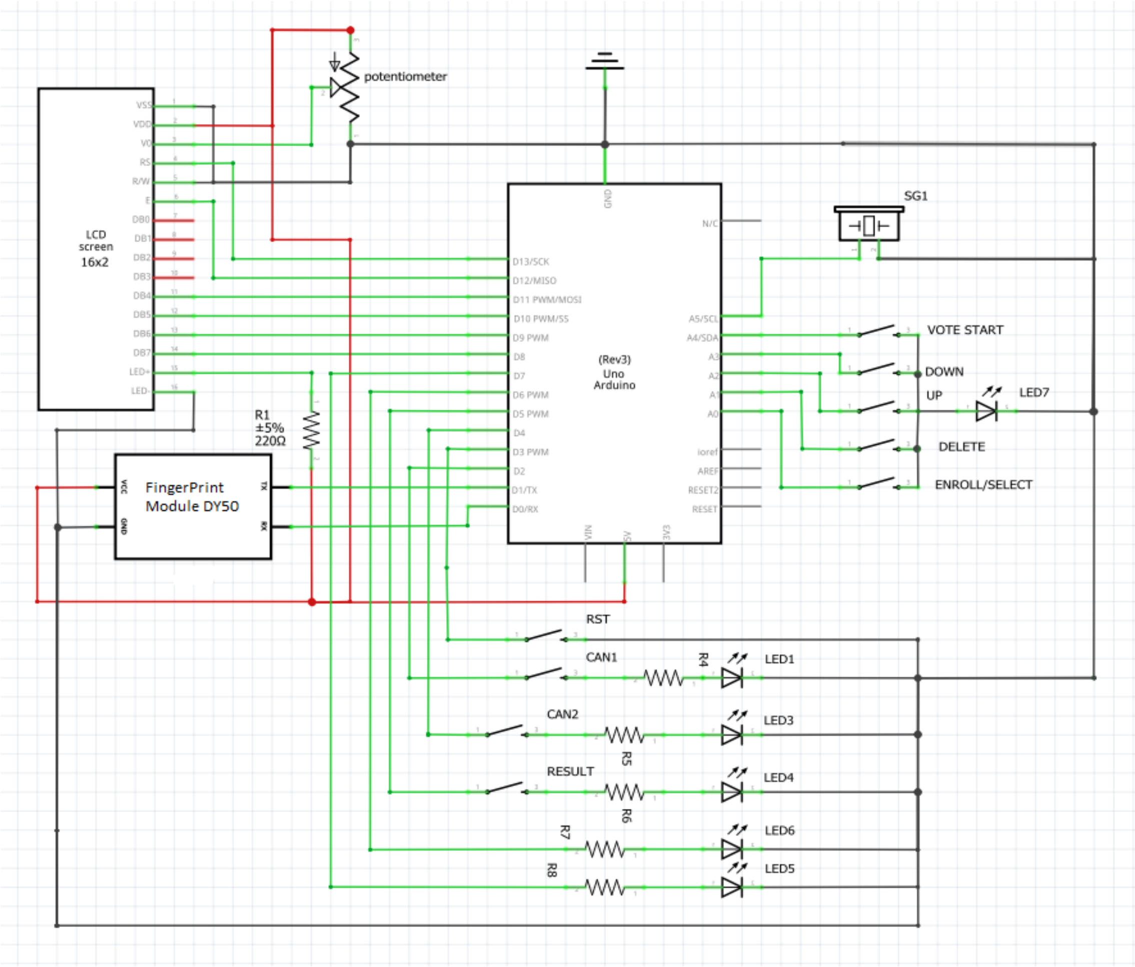


Figure 15. Schematic diagram of the Fingerprint voting system.

In addition, all the major components were first tested individually in laboratory. This was done to make sure that all components perform its task and do not affect the function of other components. Also, mid-way test were carried out when assembling the components.

Arduino Uno has built in analog to digital converter (ADC). ADC converts analog signals into digital. The analog pins of Arduino are connected to push switches as explained below:

- A0 is connected to ‘Enroll’ switch that controls the enrollment of fingerprints into the system.
- A1 is connected to ‘Delete’ switch, it is designed to operate two crucial functions in the system, delete of fingerprints that are no longer of significance and authorize the functions invoked by other switches connected to analog pins.
- A2 is connected to ‘Up’ switch which is used for the increment of number. It can be either to assigned numerical value for fingerprint during enrollment process or during deletion of any fingerprint.
- A3 is connected to ‘Down’, it is used to decrease the numerical value during enrollment and delete of fingerprint.
- The fifth pin in analog I/O pins of Arduino A4 is connected to ‘Vote Start’. When pressed, it enables system to run in voting mode, mainly invoke the vote function and gives user the instructions via LCD screen to cast vote.
- Finally, pin A5 is connected to a ‘SG1’ which is a piezo buzzer. It produces buzz sound during operation to give physical indication of button being pressed.

Digital pins present on the Arduino configures inputs or outputs according to user’s requirement. If pins are configured as output, the value is set to 0 or 5 volts. And if the pins are configured as input, an external voltage is supplied and can vary between 0-5 volts, which is then converted into digital value (0 or 1). The threshold to determine this is if voltage is below 0.8V. It is considered as 0 and if the voltage exceed 2V, digital value is considered as 1. The digital pins of Arduino Uno are connected to following components as mentioned below:

- D0/RX and D1/TX is connected to the RX and TX pins of fingerprint module respectively. These serial communication pins use TTL logic levels (5V or

3.3V). These pins are only used for connecting Arduino board and a computer or other devices.

- D2 is connected to ‘CAN1’ switch, which when pressed gives instruction to microcontroller to increase 1 vote count to candidate one.
- D3 is connected to ‘RST’ switch, this button is used to erase all the EEprom memory which is vote count in our device case. However, the program stored is not affected in any way.
- D4 is connected to ‘CAN2’ switch, when pressed increases the vote count of candidate two by single value.
- D5 is connected to ‘Result’ switch, which is used to view the result after voting has been conducted. Result in this case is votes received by two candidates.
- D6 and D7 are connected to led6 and led5 respectively.
- Digital pins D8 to D13 are connected to corresponding pins of LCD which is described already under LCD heading.

Every switch has an LED connected to it so, whenever pressed a visual confirmation is received by the user. A total of 6 resistors of 220ohms are used to limit the current through the LEDs.

The microcontroller board is powered using A/B USB port which is connected to the computer. It supplies 5V at 500mA current. The working prototype system can be seen in figure 16.

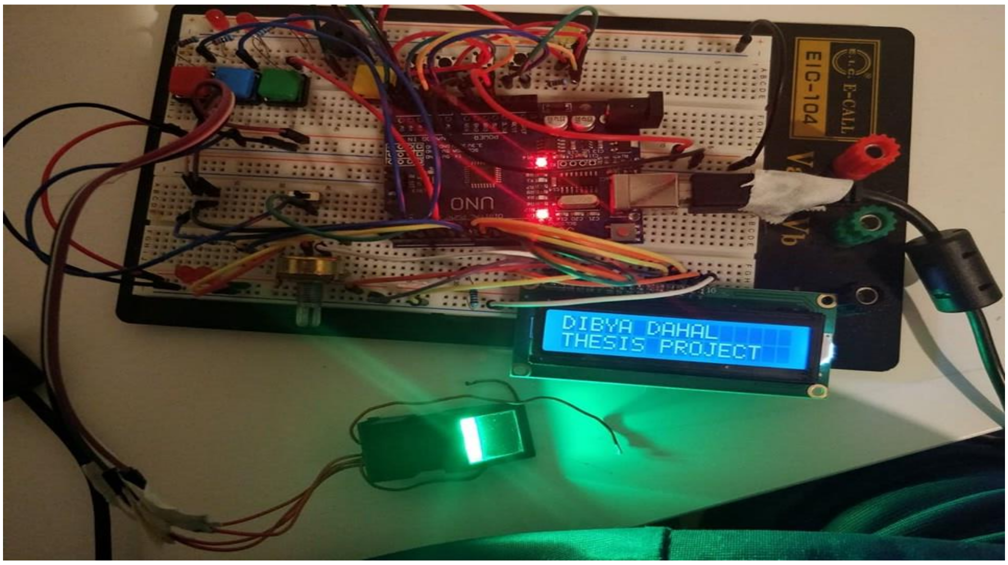


Figure 16. The working prototype system.

9 The Working Principle of the System

Prior to the voting process, fingerprints of all possible voters are required to be registered to the system to give them access to cast their votes during the voting process. Once the fingerprints are registered, the voting process can be started where voters cast their votes to desired candidate. The flow chart of working process of finger print voting system is shown in fig.17.

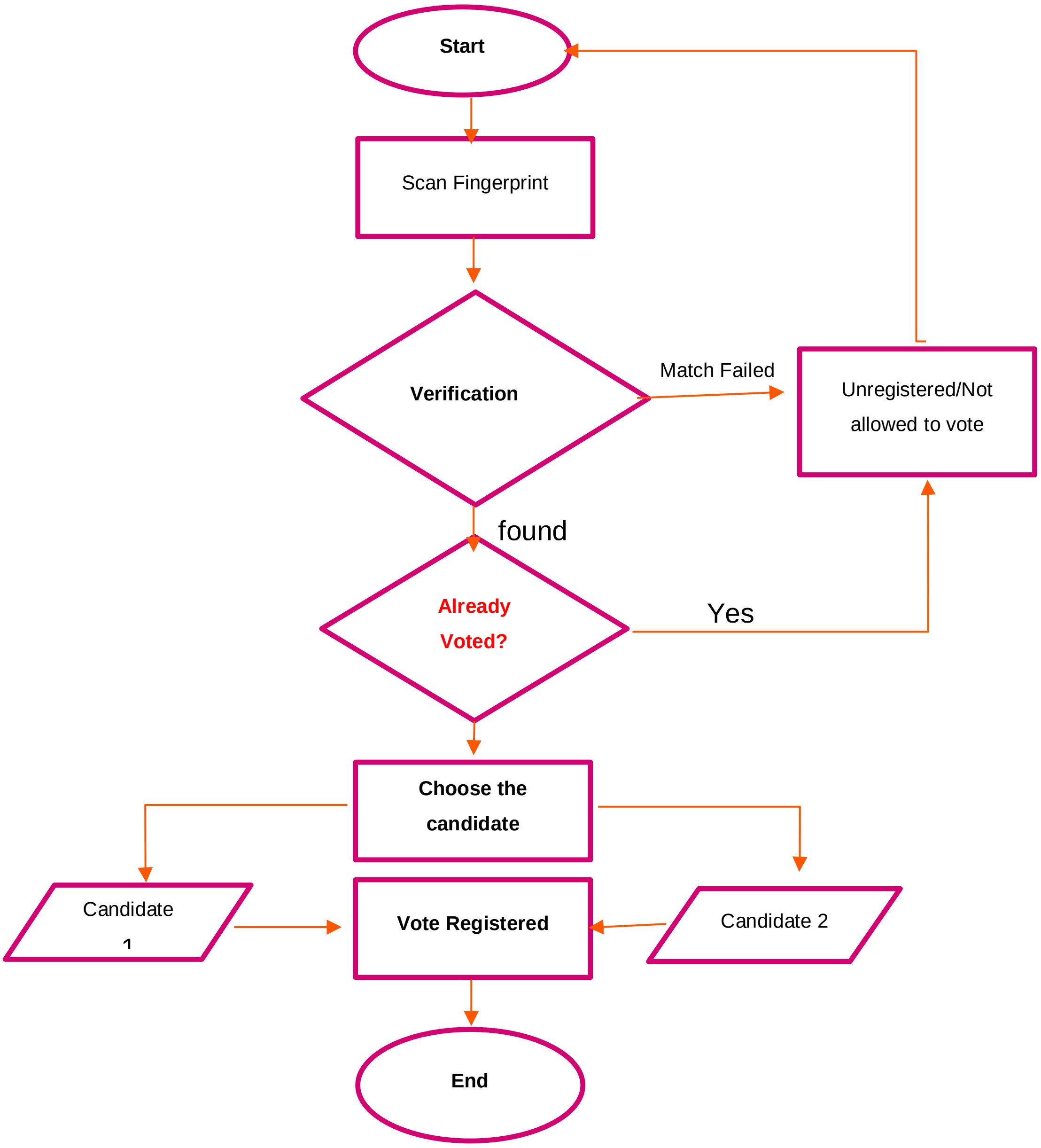


Figure 17. Flowchart of working process of fingerprint voting machine.

9.1 Enrollment

To scan and store fingerprint of the possible voters, first the enroll switch connected to A0 pin of Arduino needs to be pressed, once pressed a red LED lights up followed by the welcome message in LCD.

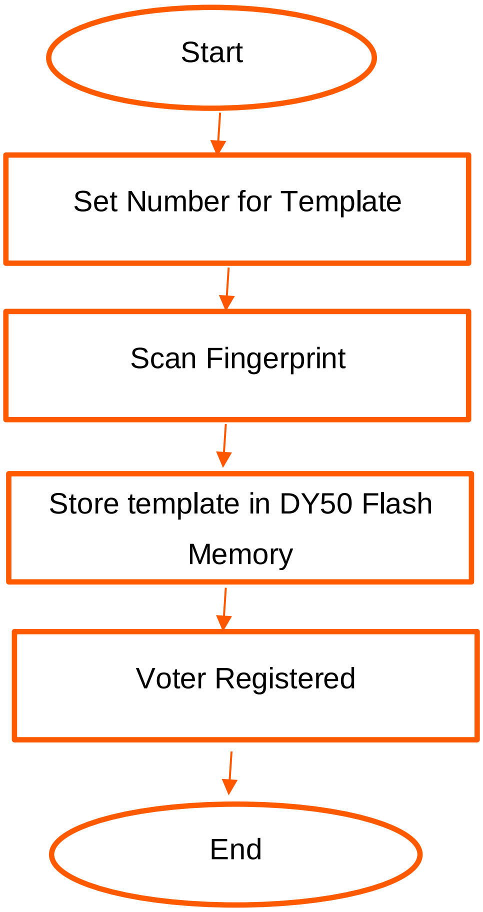


Figure 18. Flowchart of Fingerprint enrolment process.

Before scanning the finger, a numerical equivalent value needs to be assigned to the possible fingerprint so that the fingerprint can be saved in a fixed unique integer value location and can be easily accessed during the voting process. DY50 has internal memory of 512bytes and is capable to store up to 162 different fingerprints templates, therefore the unique numerical identification location to the fingerprint can be manipulated by pressing ‘Up’ and ‘Down’ switch, pressing these buttons increases and decreases the numerical value to be assigned to the fingerprint respectively. After each possible number to be assigned ‘Select’ switch needs to be pressed to authorize the action of assigning number to the fingerprint. The process of registering a fingerprint into the system in shown in figure 18 in the form of flow chart.

To execute these functions in voting system microcontroller is programmed in C language. The portion of the code that execute enrollment of voters is shown in mention under vote function. The code in structured inside Enroll function. Count is the number given for the fingerprint and its default value is set to 0. The value is increased and decreased by pressing ‘Up’ and ‘Down’ switch respectively.

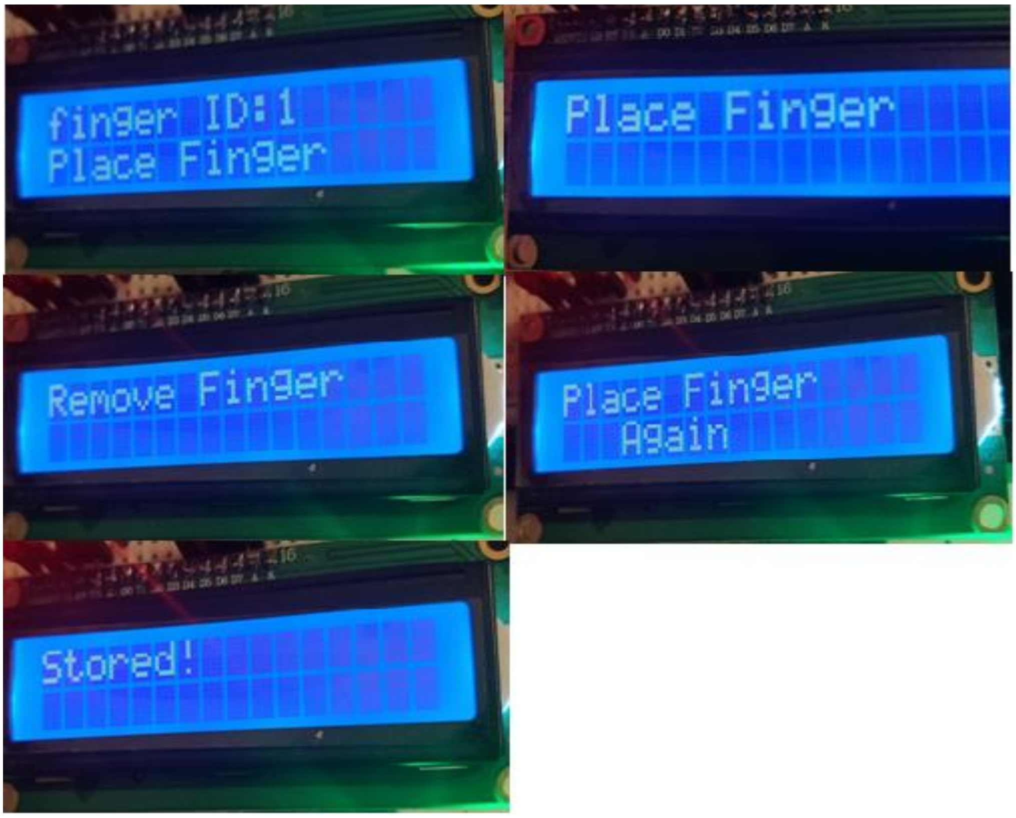
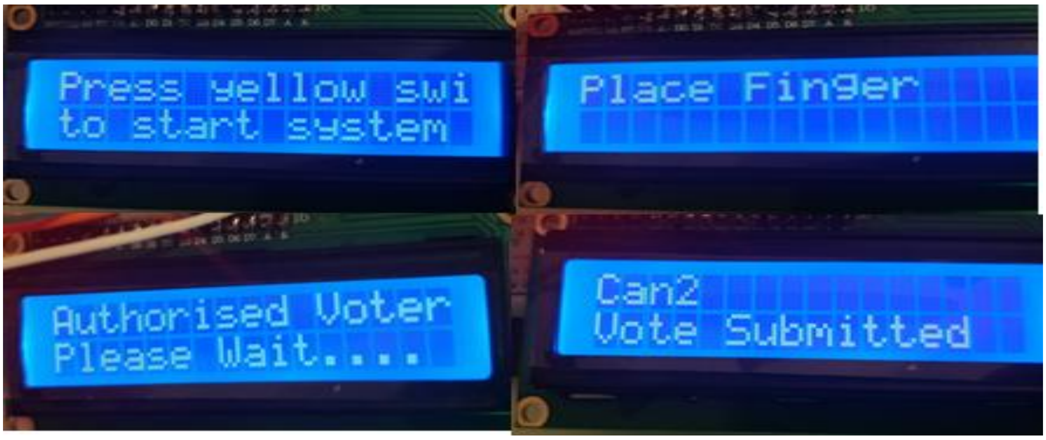


Figure 19. Messages shown in LCD during enrolment process.

Once the count is set to a unique number get FingerprintEnroll() code function is called inside Enroll function to store finger being scanned in sensor to a location inside senor memory with the integer value as selected before. Under the execution of the code for enrollment, first the number of finger id is displayed as set before and user is instructed to place finger in the sensor by popup message in LCD. If the image is not read properly user is re-instructed to place finger. Once the image is taken, the image is analyzed using finger.getImage function as provided in ada fruit library. The different status of image is analyzed using case structures (FINGERPRINT_OK, FINGERPRINT_NOFINGER, FINGERPRINT_PACKETRECIEVER, FINGERPRINT_IMAGEFAIL). Once the first image of finger is read by the scanner, user is instructed to pull the finger and put again as shown in fig.19 for the authentication of same finger before storing to the numerical location set prior to the scanning phase.Finally, after these procedures the image of finger as scanned by the scanner is converted into templates and stored as integer value in the DY50 module, the system is designed to give access to the user registered as described above. Figure 19 contains pictures of all messages that can be seen in LCD during enrolment process.

9.2 Voting Procedure

After successful registration of fingerprints, users with registered finger image are eligible to vote. The voting process is controlled by the microcontroller as instructed by code (see appendix). LCD screen then displays instruction to cast the vote if the voter authentication is passed. In this project two candidates are designed to collect the casted votes therefore user can select either candidate 1 or candidate 2 by pressing ‘CAN1’ or ‘CAN2’ switches which are connected to digital I/O pins of Arduino. After pressing switches, buzz sound is audible to notify user about vote being casted. The integral count of vote is increased and further stored in EEPROM memory of microcontroller. The instructions displayed in LCD during voting process is shown in fig. 20.



Press yellow swi
to start system

Place Finger

Authorised Voter
Please Wait....

Can2
Vote Submitted

Figure 20. Messages (left to right) displayed during voting process.

9.3 Authentication

The eligibility of a voter to vote is limited to only one time for one voting session, as the system is designed to deny multiple voting by aborting access to vote for the same finger image more than once. The portion of the code that checks the status of voter either has already voted or not is defined inside the void loop. The status is monitored by flag register, when the vote is casted the flag is assigned to 0 integer, so if the same user tries to vote for second time the flag value is evaluated before authorizing vote access to vote, and if flag has value 1, the procedure is aborted. Thus, the authenticity of the system is maintained.

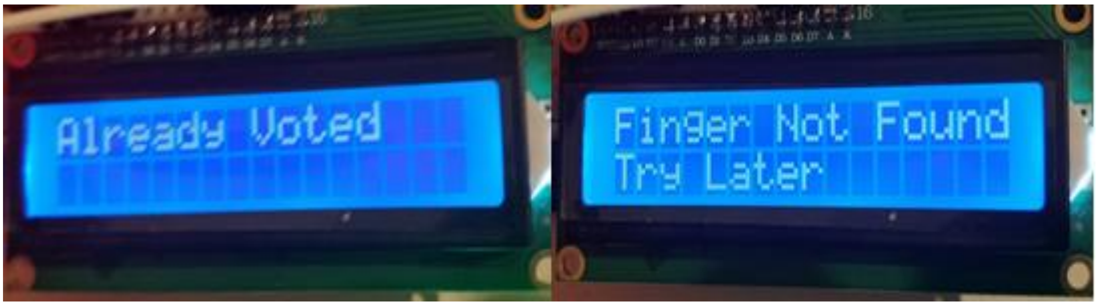


Figure 21. Message in LCD denying voting.

Also, if a person whose fingerprint is not registered in the system tries to vote the message 'Finger Not Found Try Later' is displayed in LCD screen as show in fig. 21. Thus, the person is barred from voting.

9.4 Result

All the casted votes are collected and stored in EEPROM memory and analyzed at the end of the voting process. 'Result' switch connected to digital I/O pin 2 when pressed invokes function to decide and display the result. Result function basically calculates the total count of votes of each candidate by adding number of time switches allocated for each individual candidate were pressed, in this project only two candidates were implemented. Therefore, after numerical comparison candidate with highest number of integer value is decided as winner result is displayed on the screen. This can also be seen in figure 22. Simultaneously, the result is also stored in EEPROM memory of Arduino so that it can be accessed after the voting process. As shown in figure 22, 'No Voting' message is displayed in LCD if the result function is called before the voting have been done.

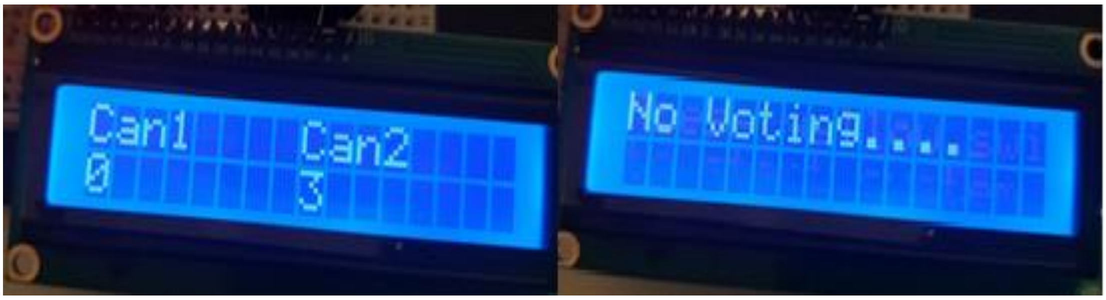


Figure 22. Two possible condition of result displayed in LCD

10 Application and Benefits of study

This biometric fingerprint system can make huge difference in overall election especially in the developing or underdeveloped countries where paper ballot system is still used and people have very little faith in election process because of various irregularities and corruption.

10.1 Effectiveness of This System in Context of Nepal

Nepal, being a democratic country, elects leader through electoral system. Nepal is among countries where paper ballot system is still in use. With high level of corruption and very low transparency, election often invites turmoil in the country. Electoral fraud, sometimes referred to as election fraud, election manipulation or vote rigging, is illegal interference with the process of an election, either by increasing the vote share of the favored candidate, depressing the vote share of the rival candidates, or both [12]. Multiple voting by a person, ballot box theft and vote rigging are also very common.

Nepal being one of the poorest country in the world, spent about 275 million euros [13] in the election in 2017.

Fingerprint voting system, if implemented, could solve many of Nepal’s electoral problems. Some advantages of using this system are discussed below:

- Safe: It is an utmost important that the voting process be secure and no one should be able to tamper with the result before, during and after voting process. Since the system is not connected to internet, no online external influence can occur. Also, it is very easy for security personnel to secure the device if needed, since it is compact, and light compared to many ballot boxes.

- Cost: The conventional paper voting system used in Nepal is very expensive because of papers and printings, transportation, staff expenses and it takes several days to count the votes. On the other hand, the biometric fingerprint system is cheap, compact and can store any amount of data with proper upgrade.

- Accuracy and Reliability: The ink used to mark people who have voted already can be erased with modern technologies but the fingerprint voting system erases the chance of multiple voting and it is very precise. Also, there is a very slim chance of errors happening from electronic system so, people can rely on results to be accurate.

- Time Saving: It takes weeks to count the votes by election personnel sometimes. This time frame is not peaceful for that area so it is ideal to get result as soon as possible. The manual vote counting technique can be seen in figure 20. With electronic device election results can be produced is minutes rather than days or weeks.

  The voters do not have to carry their ID with them since the fingerprint acts as their ID when voting so queues can clear faster. Unauthorized or voters not enrolled in system before hand is not allowed to vote.

- Trees Conservation: Paper ballot system uses paper for voting process which is printed in millions of quantities. An example of sheer size of the ballot paper is shown in the fig. 23. Little to no paper is used in electronic voting so it helps trees conservation.

Figure 23. Manual vote counting in Dang district, Nepal on June 29,2017 [14].

10.2 Possible Risks

The risk of tampering is undeniable with any physical device. The device is light and made up of plastic fiber, therefore can easily break on impact. In such scenario, data might lose permanently or would be difficult to retrieve.

System and software can have glitch and other problems that may result in delay or even halt the voting process. Also, process of collecting fingerprint data of millions of people is tough and time consuming. Illiterate people who have not used electronic devices and are used to paper voting might find it difficult to understand the voting process. Adequate pre-election information or classes must be conducted which adds to the election cost.

11 Scope For Upgrade

The focus of the project was to design a fingerprint voting system which makes the voting process swift, sound and eradicate defrauding caused by manual voting process. Since the system made in this project is just a basic prototype, there are certainly huge scopes for upgrading with better funding and research. Some ways this project can be upgraded are discussed below:

- It is possible to add a WIFI module to this system. Many voting machines can then be connected with host server with secure LAN. This will allow to store data in host server placed at secure location in real time. Physical damaging the device will not cause in loss of data since the data are now transferred to host server in real time. An example of various voting machine connected to host server is shown in figure 24. Also, a GSM module can be added to send the result to various partakers.

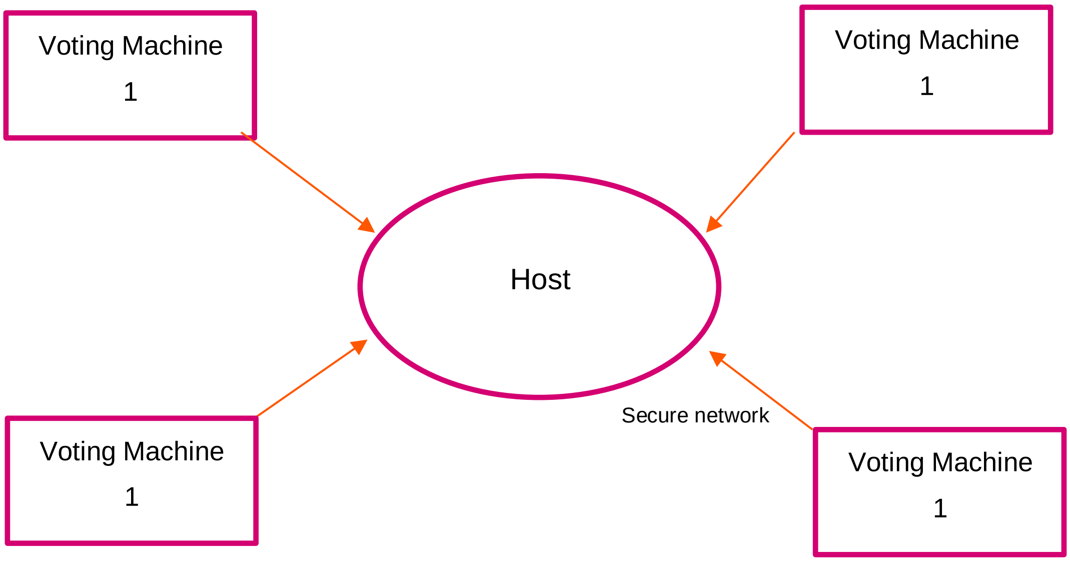


Figure 24. Network of voting machine connected to server

- Although the prototype made in this thesis project aims to facilitate safe and swift voting process, with some hardware modification the same device can also be used for various other things like finger print based access control, attendance, billing, etc.

- Better components with higher functioning calibre can be used to make the device more reliable in practical situations. For example, the fingerprint module in this prototype can accommodate only 162 fingerprint templates. This amount can be increased to any number by adding an extra memory space such as SD card.

12 Conclusion

This thesis study aimed to design and develop a prototype fingerprint voting system that ensures the voting process be safe and swift. For this a prototype device was made namely Arduino Uno based fingerprint voting system using DY50 fingerprint module, LED (16×2) and EEProm internal memory storage technique. The final system is the result of various successful hardware and software integration. The process includes review and analysis, designing the system and algorithm, hardwiring, hardware and software integration, test and troubleshooting and result analysis.

To summarize, the prototype device was successfully able to enroll the fingerprint of the voters in DY50 fingerprint module flash memory, verify the status of voters (registration and multiple voting), matching the new fingerprint input with saved fingerprint template, authorize the voter to cast the vote and was able to generate result. To conclude, the device is great alternative to other lengthy election processes especially ballot paper voting system.

Further improvement of the prototype device could be done at the later development stage. For instance, an addition of WIFI module could help send result wirelessly to host computer and adding external memory space could help store any amount of fingerprint data.

13 Bibliography

[1] A. K. Sharma, A. Raghuwanshi and V. K. Sharma, "Biometric System- A Review," *International Journal of Computer Science and Information Technology,* vol. 6, p. 1, 2015.

[2] B. Aleksandra, "handle/10024/44684," 2012. [Online]. Available: https://www.theseus.fi/handle/10024/44684.

[3] "technology," 2013. [Online]. Available: http://www.nitgen.com/eng/technology/technology_02.html. [Accessed 18 10 2019].

[4] "Adafruit," Adafruit, [Online]. Available: https://www.adafruit.com/product/751?gclid=CjwKEAj. [Accessed 06 11 2019].

[5] "arduino," 11 11 2019. [Online]. Available: https://store.arduino.cc/.

[6] B. Sigdel, "Thesis," Metropolia UAS, Helsinki, 2017.

[7] "software," 11 11 2019. [Online]. Available: https://www.arduino.cc/.

[8] "components101," [Online]. Available: https://components101.com/microcontrollers/arduino-uno. [Accessed 14 11 2019].

[9] Dejan, "lcd tutorials," [Online]. Available: https://howtomechatronics.com/tutorials/arduino/lcd-tutorial/. [Accessed 08 10 2019].

[10] U. Hussaini, "technobyte," [Online]. Available: https://www.technobyte.org. [Accessed 17 11 2019].

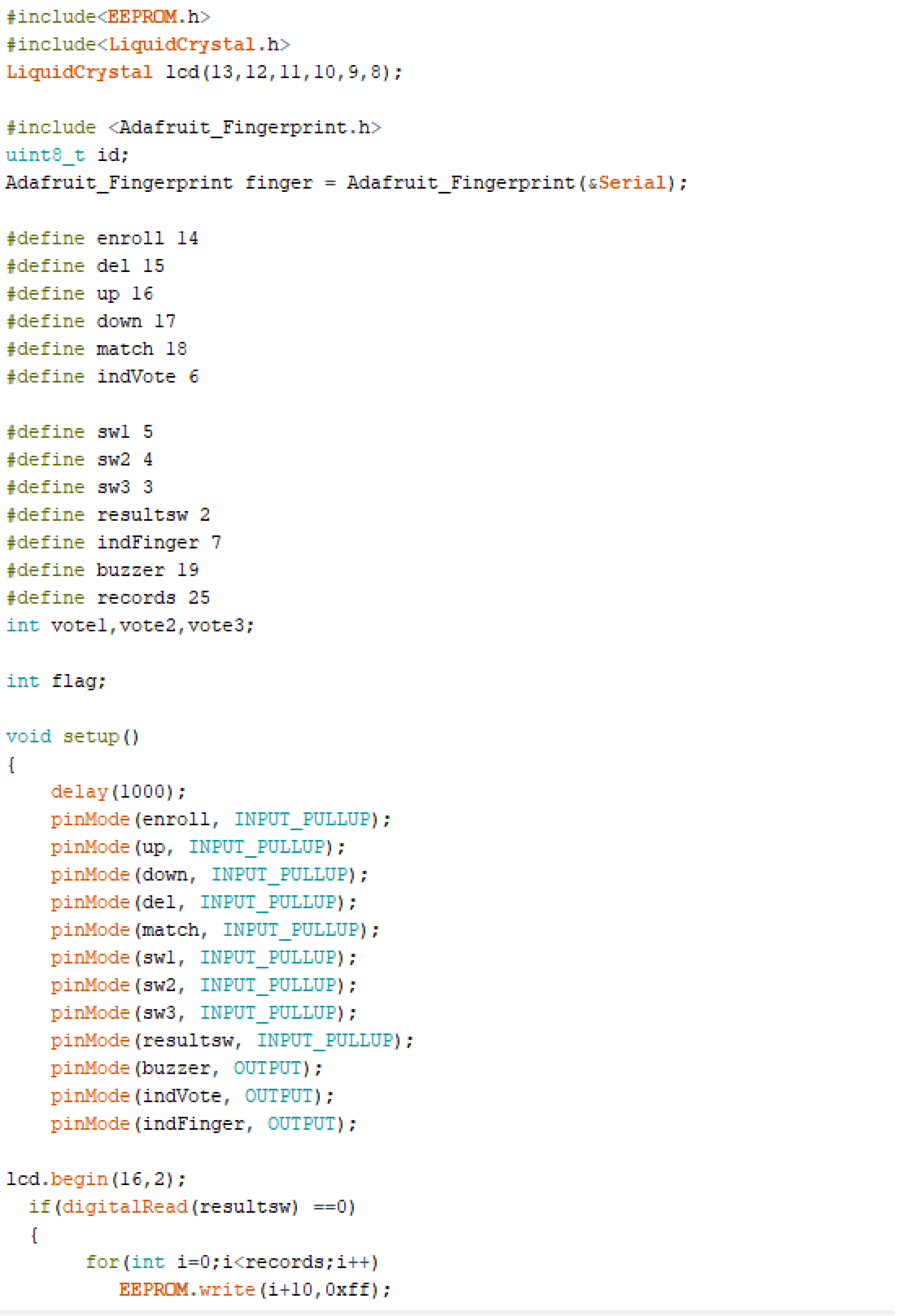
[11] "wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/USB. [Accessed 17 11 2019].

[12] 19 11 2019. [Online]. Available: https://en.wikipedia.org/wiki/Electoral_fraud. [Accessed 20 11 2019].

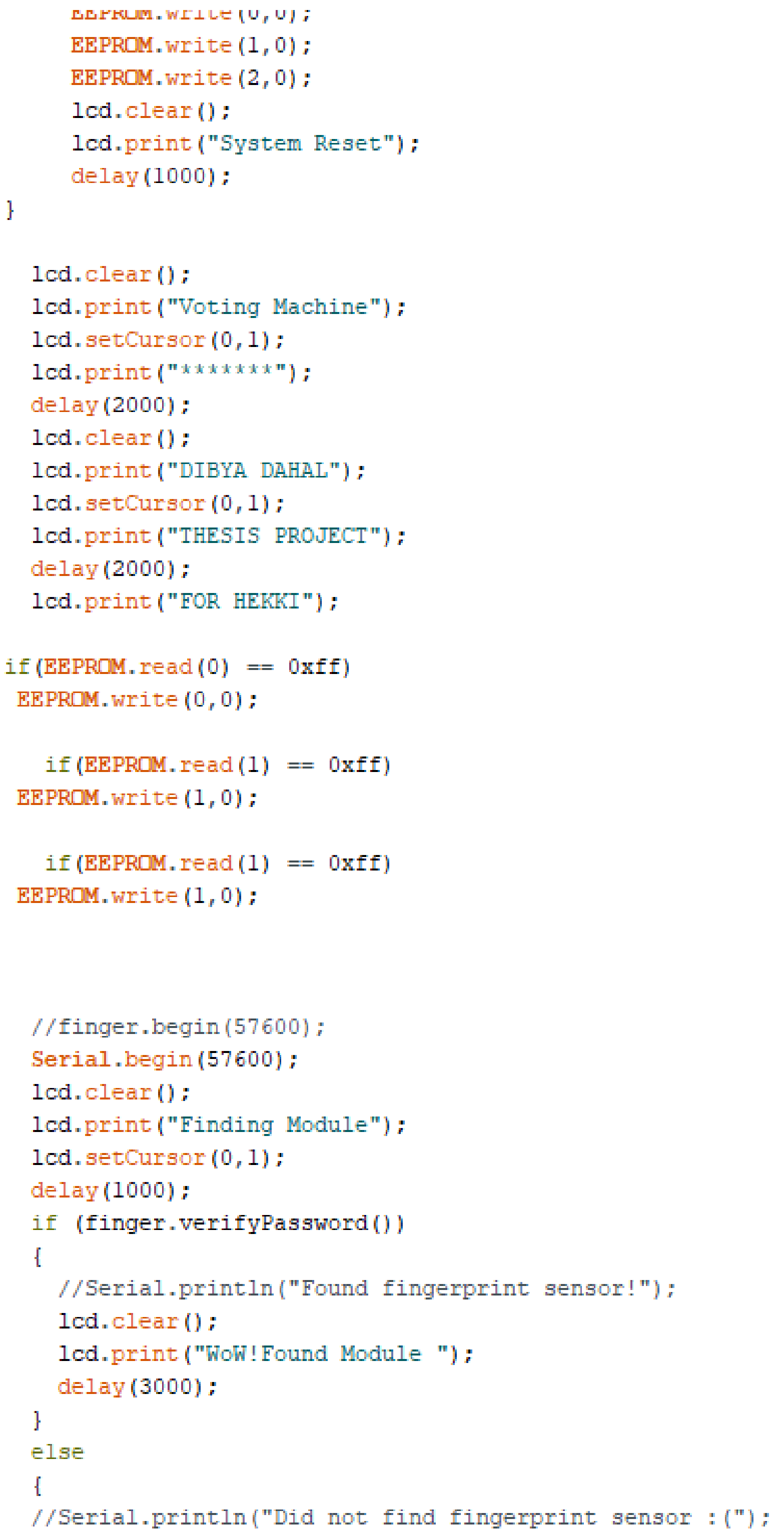
[13] P. Pokharel, "Study on the Election Campaign Finance of Local, Provincial, and Federal Elections in Nepal," Pradip Pokharel, Kathmandu, 2017.

[14] "Kathmandu Post," 14 05 2017. [Online]. Available: https://kathmandupost.com/national/2017/05/15/ballot-papers-confusing.

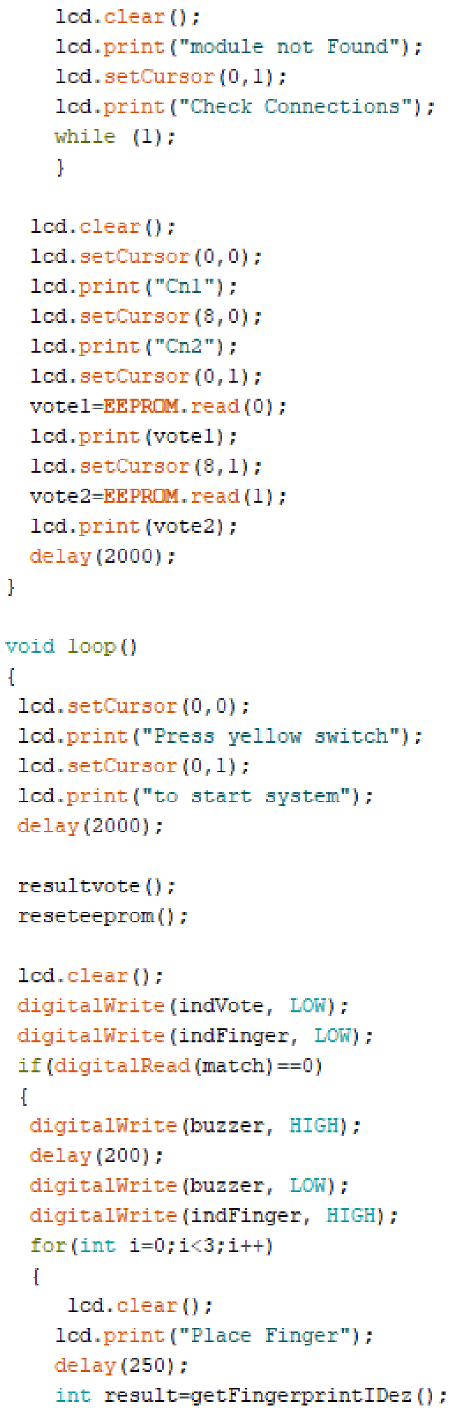
Programming Codes



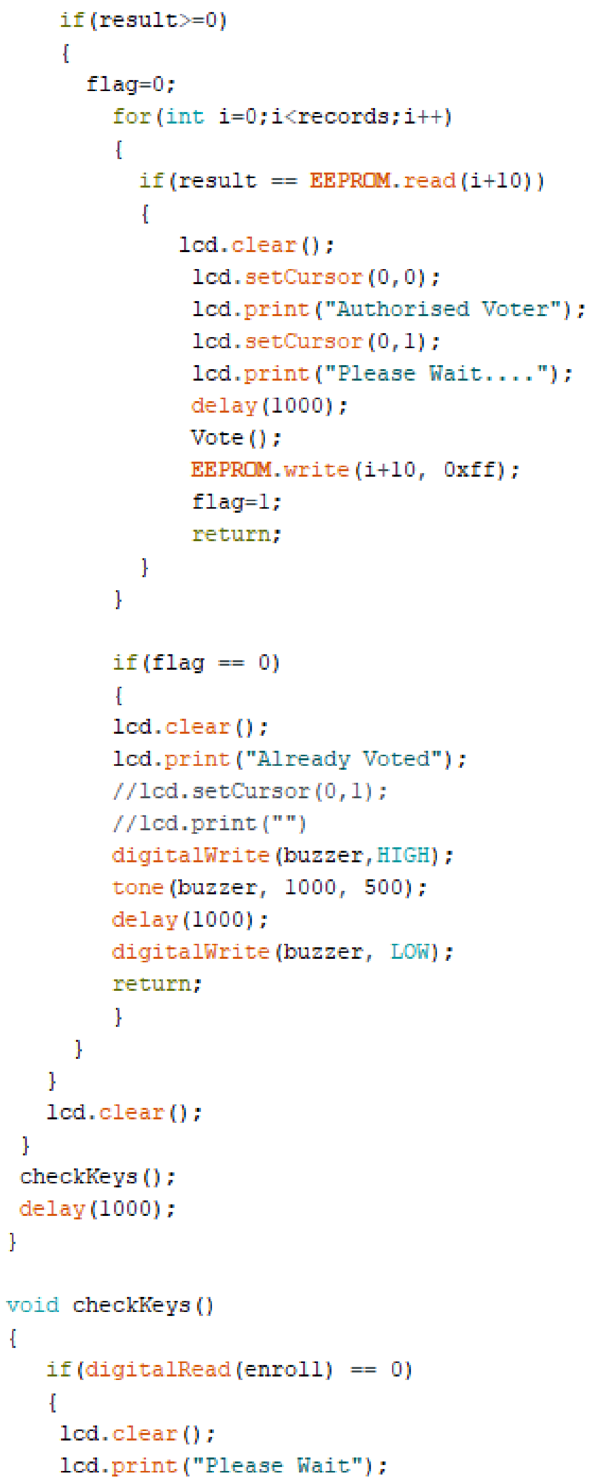
```
#include<EEPROM.h>
#include<LiquidCrystal.h>
LiquidCrystal lcd(13,12,11,10,9,8);

#include <Adafruit_Fingerprint.h>
uint8_t id;
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial);

#define enroll 14
#define del 15
#define up 16
#define down 17
#define match 18
#define indVote 6

#define sw1 5
#define sw2 4
#define sw3 3
#define resultsw 2
#define indFinger 7
#define buzzer 19
#define records 25
int vote1,vote2,vote3;

int flag;

void setup()
{
    delay(1000);
    pinMode(enroll, INPUT_PULLUP);
    pinMode(up, INPUT_PULLUP);
    pinMode(down, INPUT_PULLUP);
    pinMode(del, INPUT_PULLUP);
    pinMode(match, INPUT_PULLUP);
    pinMode(sw1, INPUT_PULLUP);
    pinMode(sw2, INPUT_PULLUP);
    pinMode(sw3, INPUT_PULLUP);
    pinMode(resultsw, INPUT_PULLUP);
    pinMode(buzzer, OUTPUT);
    pinMode(indVote, OUTPUT);
    pinMode(indFinger, OUTPUT);

lcd.begin(16,2);
  if(digitalRead(resultsw) ==0)
  {
       for(int i=0;i<records;i++)
          EEPROM.write(i+10,0xff);
```



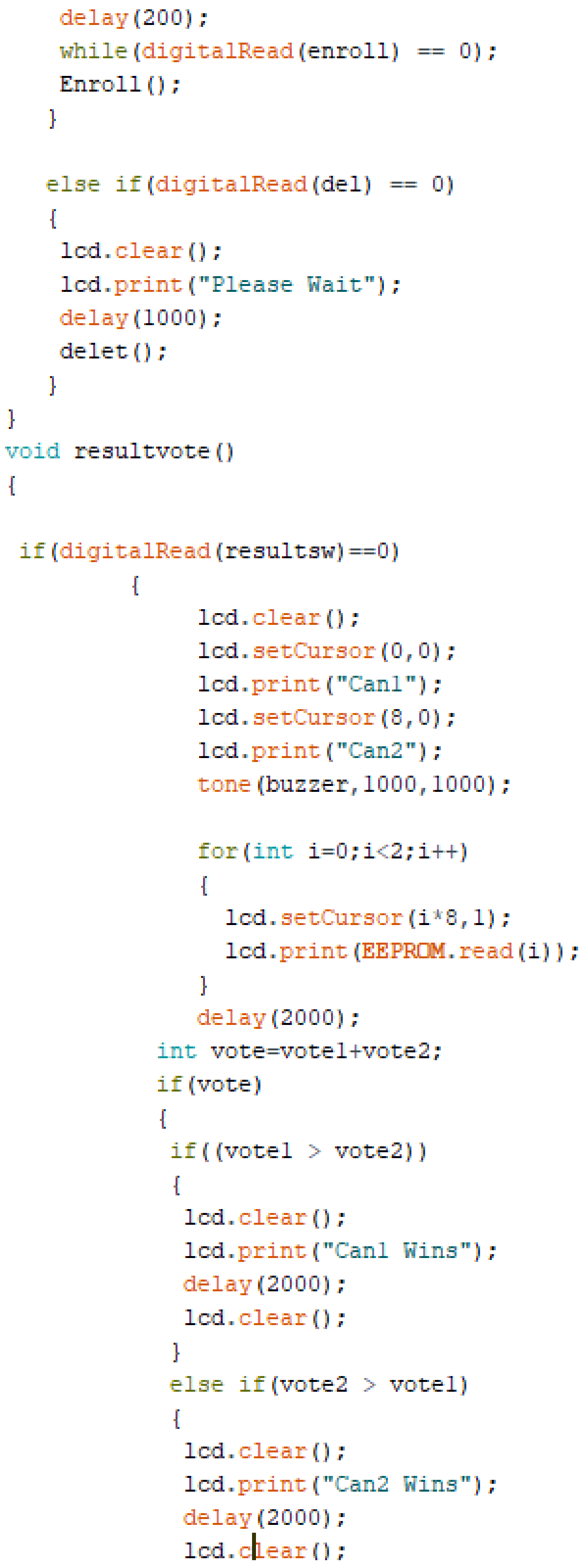
```
     EEPROM.write(0,0);
     EEPROM.write(1,0);
     EEPROM.write(2,0);
     lcd.clear();
     lcd.print("System Reset");
     delay(1000);
}

  lcd.clear();
  lcd.print("Voting Machine");
  lcd.setCursor(0,1);
  lcd.print("*******");
  delay(2000);
  lcd.clear();
  lcd.print("DIBYA DAHAL");
  lcd.setCursor(0,1);
  lcd.print("THESIS PROJECT");
  delay(2000);
  lcd.print("FOR HEKKI");

if(EEPROM.read(0) == 0xff)
 EEPROM.write(0,0);

   if(EEPROM.read(1) == 0xff)
 EEPROM.write(1,0);

   if(EEPROM.read(1) == 0xff)
 EEPROM.write(1,0);



  //finger.begin(57600);
  Serial.begin(57600);
  lcd.clear();
  lcd.print("Finding Module");
  lcd.setCursor(0,1);
  delay(1000);
  if (finger.verifyPassword())
  {
    //Serial.println("Found fingerprint sensor!");
    lcd.clear();
    lcd.print("WoW!Found Module ");
    delay(3000);
  }
  else
  {
  //Serial.println("Did not find fingerprint sensor :(");
```



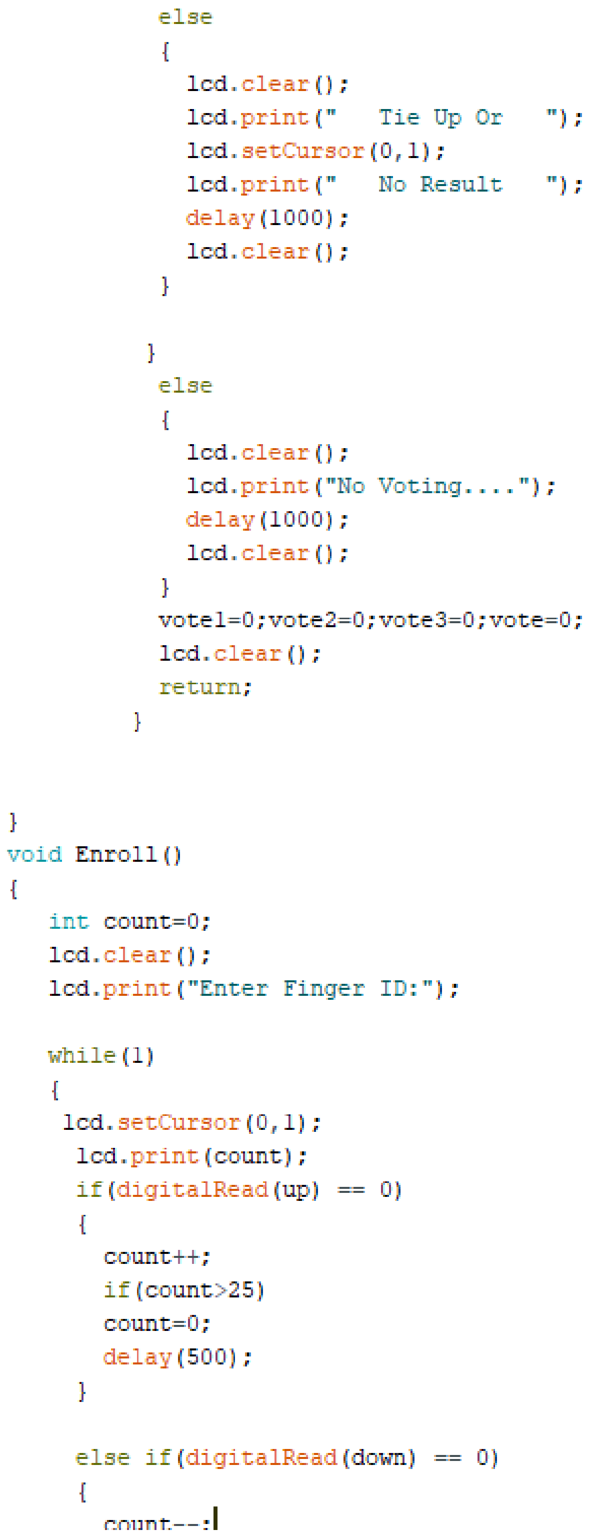
```
    lcd.clear();
    lcd.print("module not Found");
    lcd.setCursor(0,1);
    lcd.print("Check Connections");
    while (1);
    }

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Cn1");
  lcd.setCursor(8,0);
  lcd.print("Cn2");
  lcd.setCursor(0,1);
  vote1=EEPROM.read(0);
  lcd.print(vote1);
  lcd.setCursor(8,1);
  vote2=EEPROM.read(1);
  lcd.print(vote2);
  delay(2000);
}

void loop()
{
 lcd.setCursor(0,0);
 lcd.print("Press yellow switch");
 lcd.setCursor(0,1);
 lcd.print("to start system");
 delay(2000);

 resultvote();
 reseteeprom();

 lcd.clear();
 digitalWrite(indVote, LOW);
 digitalWrite(indFinger, LOW);
 if(digitalRead(match)==0)
 {
  digitalWrite(buzzer, HIGH);
  delay(200);
  digitalWrite(buzzer, LOW);
  digitalWrite(indFinger, HIGH);
  for(int i=0;i<3;i++)
  {
     lcd.clear();
    lcd.print("Place Finger");
    delay(250);
    int result=getFingerprintIDez();
```



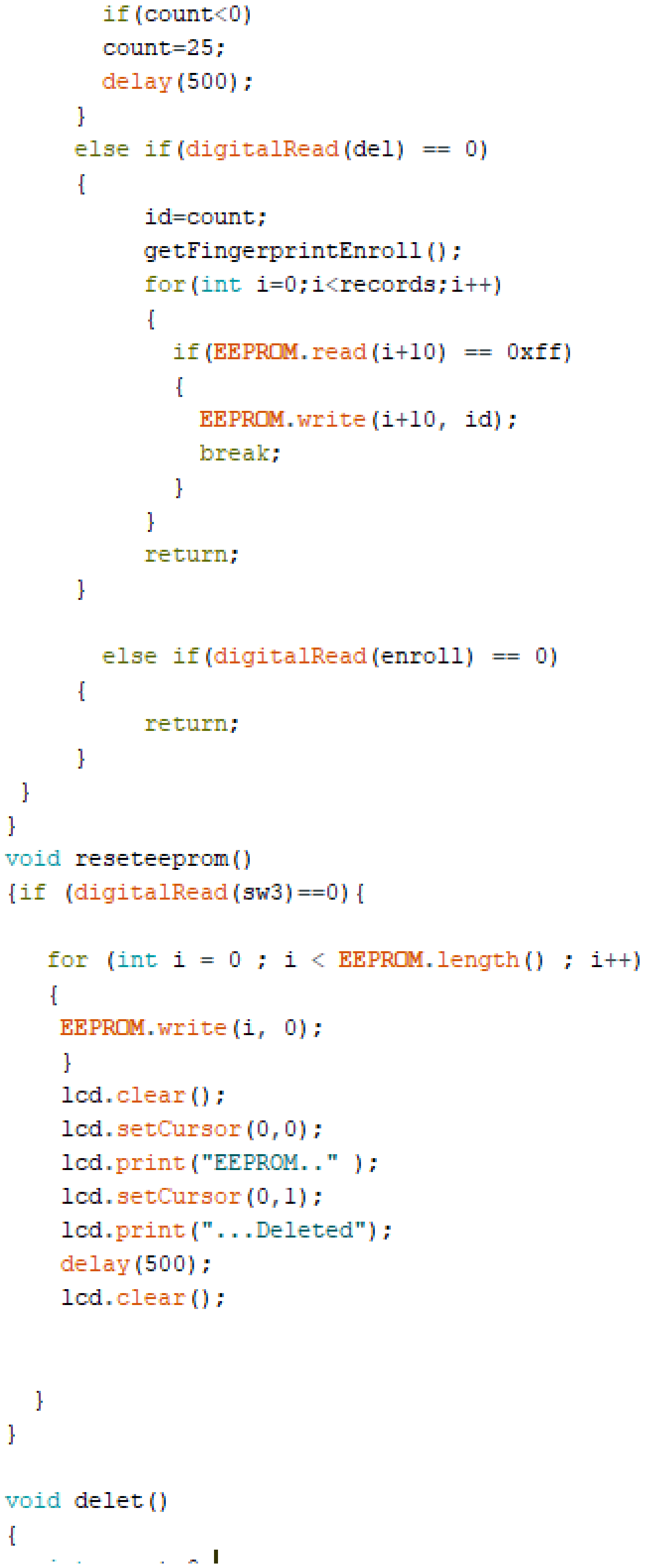
```
    if(result>=0)
    {
      flag=0;
        for(int i=0;i<records;i++)
        {
          if(result == EEPROM.read(i+10))
          {
             lcd.clear();
              lcd.setCursor(0,0);
              lcd.print("Authorised Voter");
              lcd.setCursor(0,1);
              lcd.print("Please Wait....");
              delay(1000);
              Vote();
              EEPROM.write(i+10, 0xff);
              flag=1;
              return;
          }
        }

        if(flag == 0)
        {
        lcd.clear();
        lcd.print("Already Voted");
        //lcd.setCursor(0,1);
        //lcd.print("")
        digitalWrite(buzzer,HIGH);
        tone(buzzer, 1000, 500);
        delay(1000);
        digitalWrite(buzzer, LOW);
        return;
        }
     }
   }
   lcd.clear();
 }
 checkKeys();
 delay(1000);
}

void checkKeys()
{
   if(digitalRead(enroll) == 0)
   {
    lcd.clear();
    lcd.print("Please Wait");
```



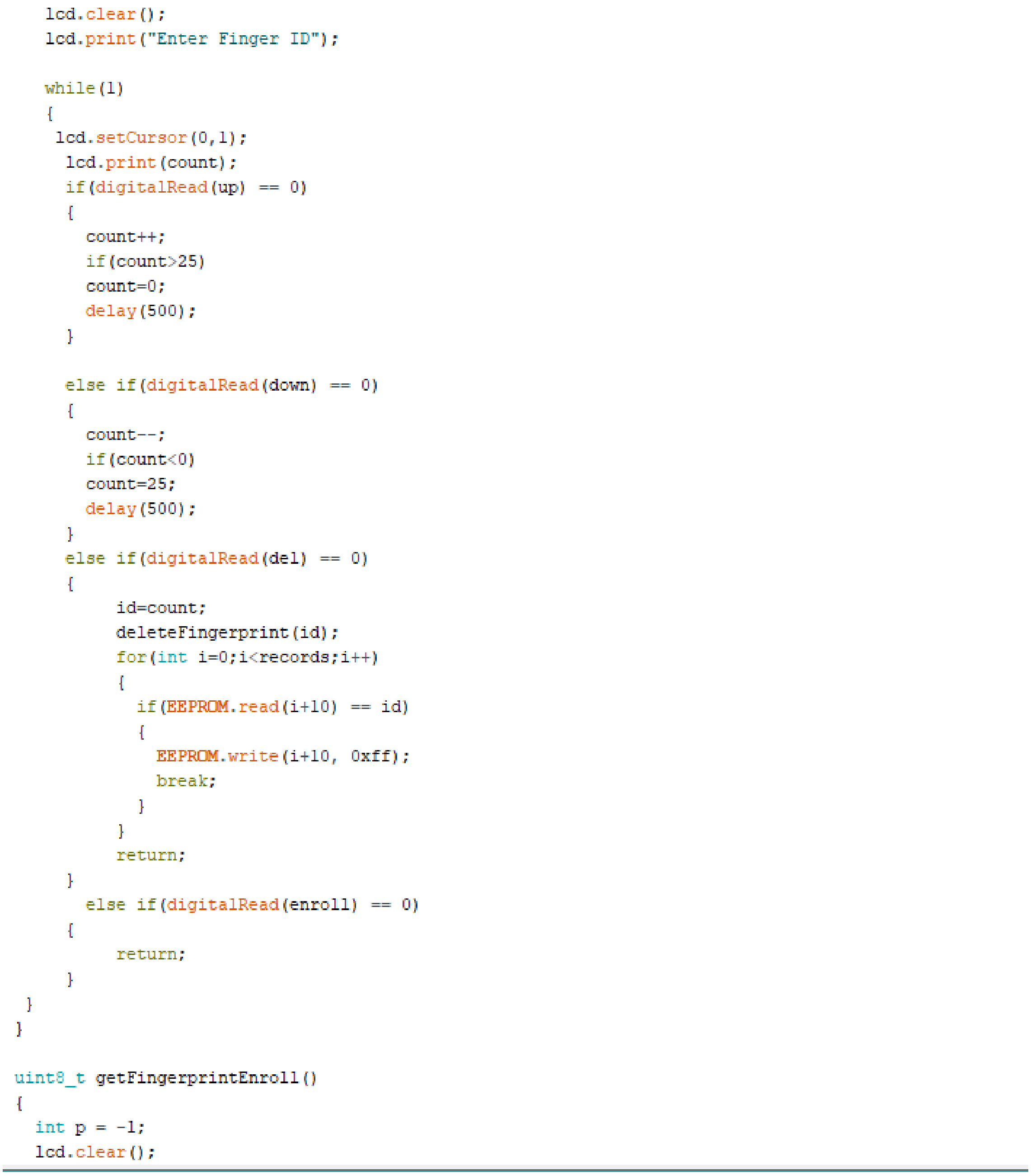
```
    delay(200);
    while(digitalRead(enroll) == 0);
    Enroll();
   }

   else if(digitalRead(del) == 0)
   {
    lcd.clear();
    lcd.print("Please Wait");
    delay(1000);
    delet();
   }
}
void resultvote()
{

 if(digitalRead(resultsw)==0)
         {
              lcd.clear();
              lcd.setCursor(0,0);
              lcd.print("Can1");
              lcd.setCursor(8,0);
              lcd.print("Can2");
              tone(buzzer,1000,1000);

              for(int i=0;i<2;i++)
              {
                lcd.setCursor(i*8,1);
                lcd.print(EEPROM.read(i));
              }
              delay(2000);
           int vote=vote1+vote2;
           if(vote)
           {
            if((vote1 > vote2))
            {
             lcd.clear();
             lcd.print("Can1 Wins");
             delay(2000);
             lcd.clear();
            }
            else if(vote2 > vote1)
            {
             lcd.clear();
             lcd.print("Can2 Wins");
             delay(2000);
             lcd.clear();
```



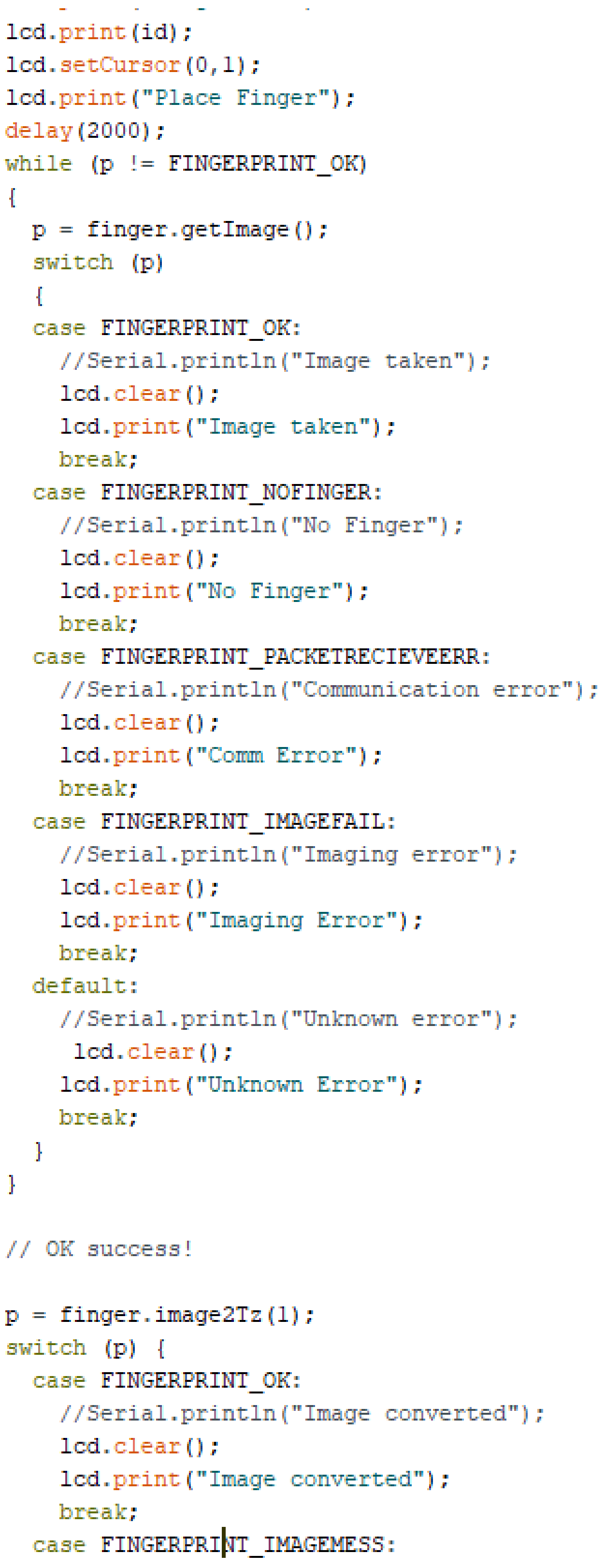
```
           else
           {
             lcd.clear();
             lcd.print("   Tie Up Or   ");
             lcd.setCursor(0,1);
             lcd.print("   No Result   ");
             delay(1000);
             lcd.clear();
           }

          }
           else
           {
             lcd.clear();
             lcd.print("No Voting....");
             delay(1000);
             lcd.clear();
           }
           vote1=0;vote2=0;vote3=0;vote=0;
           lcd.clear();
           return;
         }


}
void Enroll()
{
   int count=0;
   lcd.clear();
   lcd.print("Enter Finger ID:");

   while(1)
   {
    lcd.setCursor(0,1);
     lcd.print(count);
     if(digitalRead(up) == 0)
     {
       count++;
       if(count>25)
       count=0;
       delay(500);
     }

     else if(digitalRead(down) == 0)
     {
       count--;
```



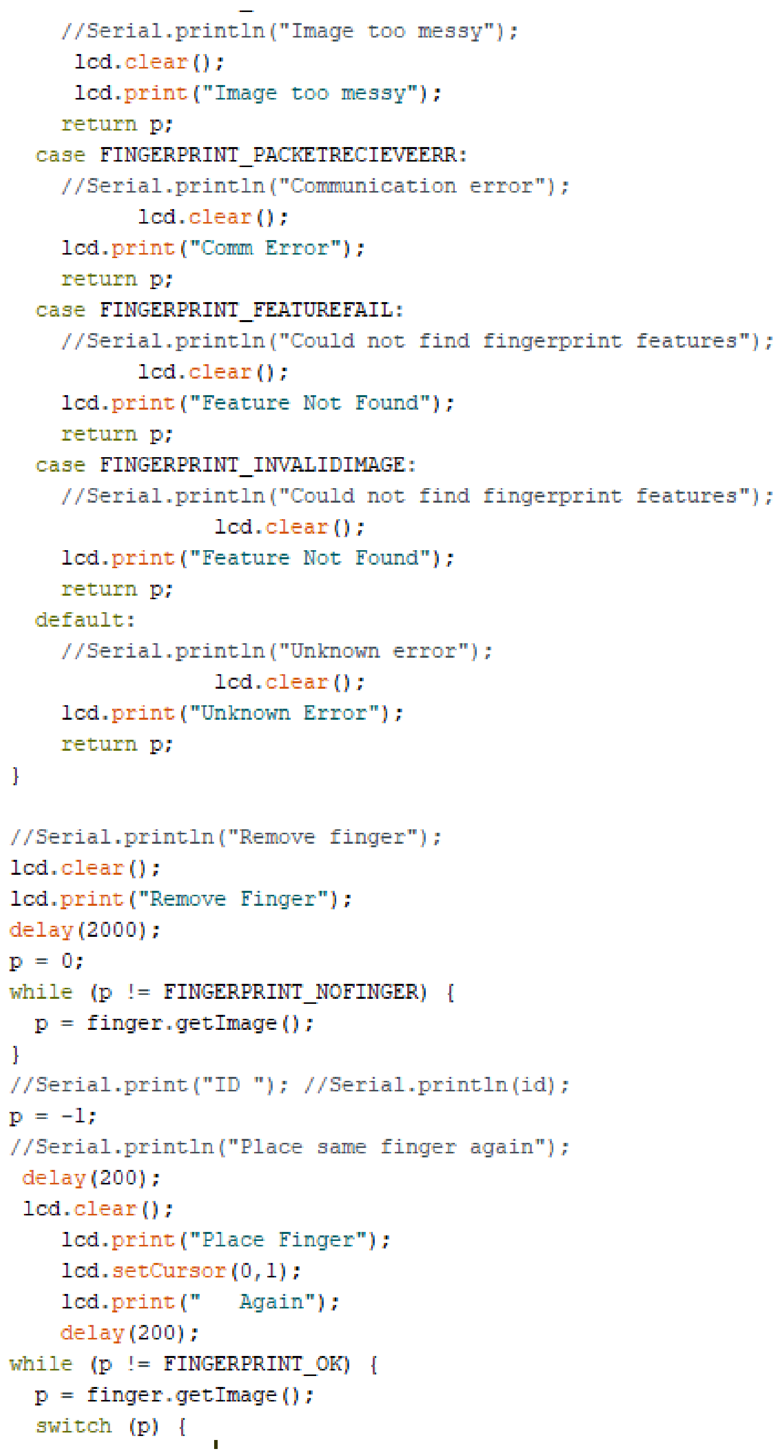
```
       if(count<0)
       count=25;
       delay(500);
     }
     else if(digitalRead(del) == 0)
     {
          id=count;
          getFingerprintEnroll();
          for(int i=0;i<records;i++)
          {
            if(EEPROM.read(i+10) == 0xff)
            {
              EEPROM.write(i+10, id);
              break;
            }
          }
          return;
     }

       else if(digitalRead(enroll) == 0)
     {
          return;
     }
 }
}
void reseteeprom()
{if (digitalRead(sw3)==0){

   for (int i = 0 ; i < EEPROM.length() ; i++)
   {
    EEPROM.write(i, 0);
    }
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("EEPROM.." );
    lcd.setCursor(0,1);
    lcd.print("...Deleted");
    delay(500);
    lcd.clear();


  }
}

void delet()
{
```



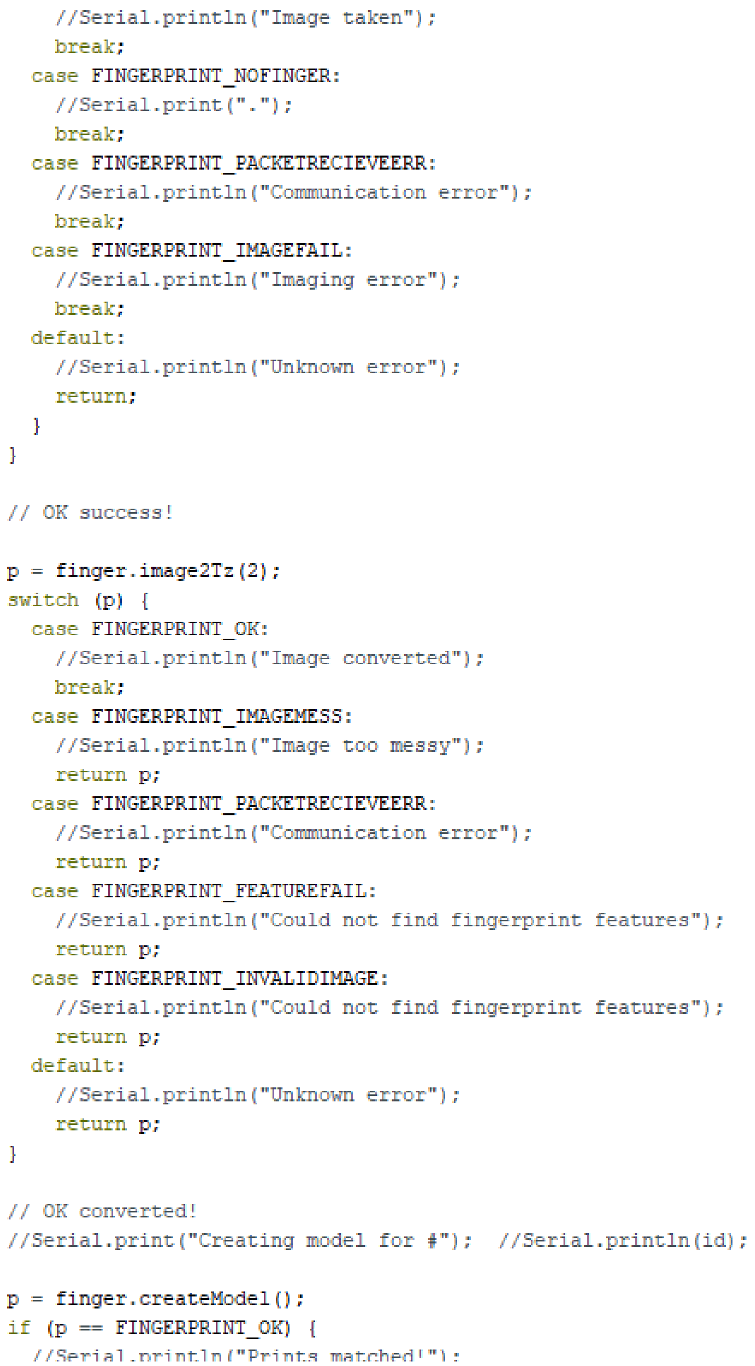
```
   lcd.clear();
   lcd.print("Enter Finger ID");

   while(1)
   {
    lcd.setCursor(0,1);
     lcd.print(count);
     if(digitalRead(up) == 0)
     {
       count++;
       if(count>25)
       count=0;
       delay(500);
     }

     else if(digitalRead(down) == 0)
     {
       count--;
       if(count<0)
       count=25;
       delay(500);
     }
     else if(digitalRead(del) == 0)
     {
          id=count;
          deleteFingerprint(id);
          for(int i=0;i<records;i++)
          {
            if(EEPROM.read(i+10) == id)
            {
              EEPROM.write(i+10, 0xff);
              break;
            }
          }
          return;
     }
       else if(digitalRead(enroll) == 0)
     {
          return;
     }
 }
}

uint8_t getFingerprintEnroll()
{
  int p = -1;
  lcd.clear();
```



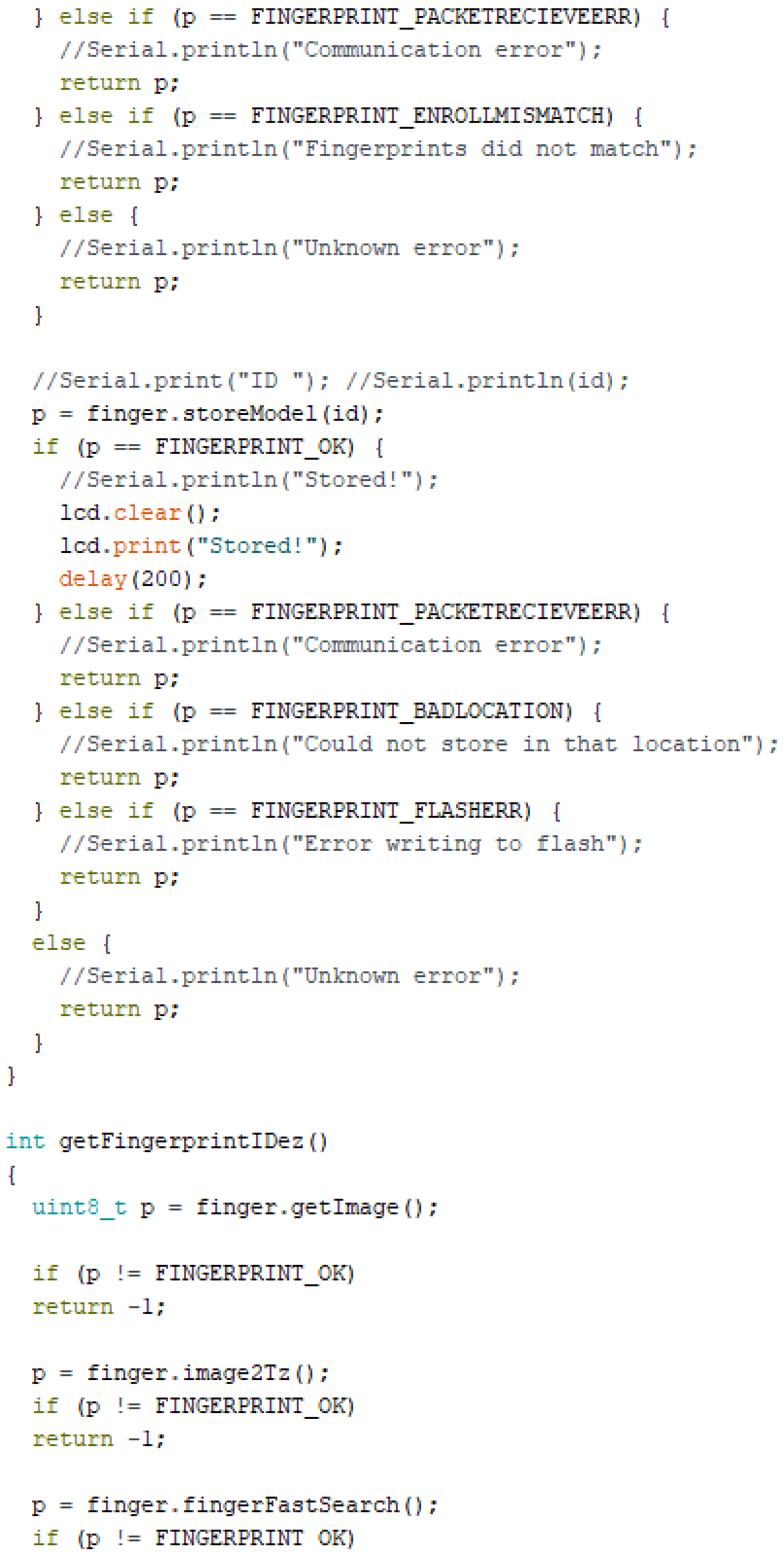
```
lcd.print(id);
lcd.setCursor(0,1);
lcd.print("Place Finger");
delay(2000);
while (p != FINGERPRINT_OK)
{
  p = finger.getImage();
  switch (p)
  {
  case FINGERPRINT_OK:
    //Serial.println("Image taken");
    lcd.clear();
    lcd.print("Image taken");
    break;
  case FINGERPRINT_NOFINGER:
    //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:
```



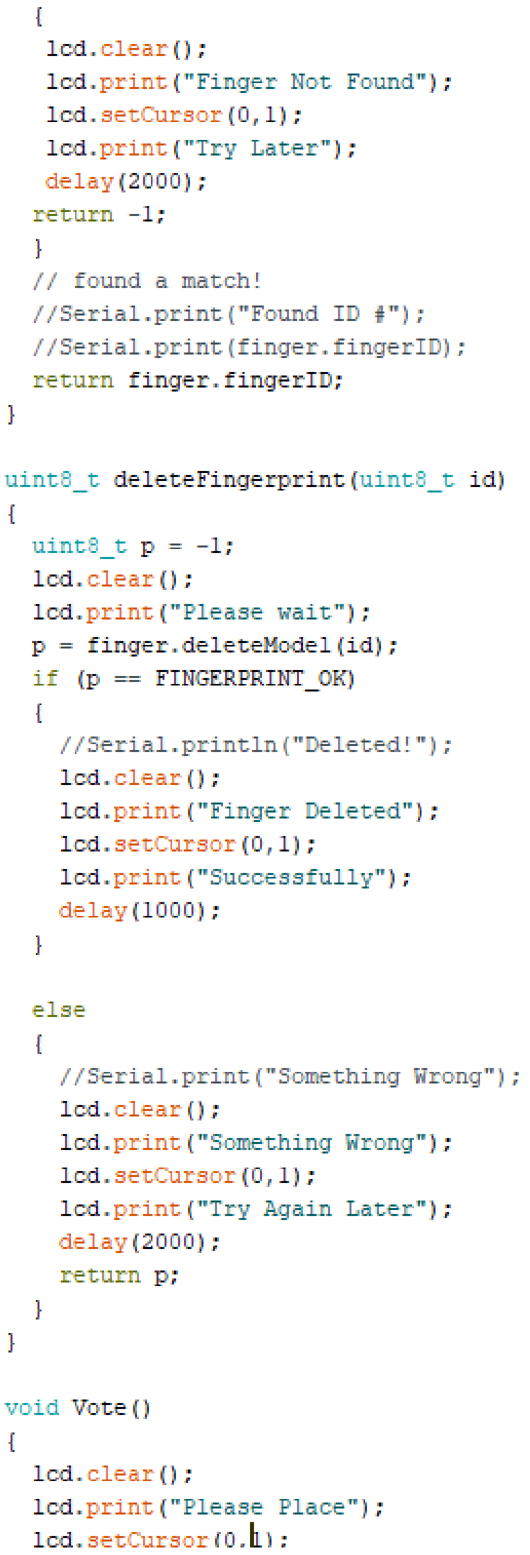
```
    //Serial.println("Image too messy");
     lcd.clear();
     lcd.print("Image too messy");
    return p;
  case FINGERPRINT_PACKETRECIEVEERR:
    //Serial.println("Communication error");
          lcd.clear();
    lcd.print("Comm Error");
    return p;
  case FINGERPRINT_FEATUREFAIL:
    //Serial.println("Could 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");
 delay(200);
 lcd.clear();
    lcd.print("Place Finger");
    lcd.setCursor(0,1);
    lcd.print("   Again");
    delay(200);
while (p != FINGERPRINT_OK) {
  p = finger.getImage();
  switch (p) {
```



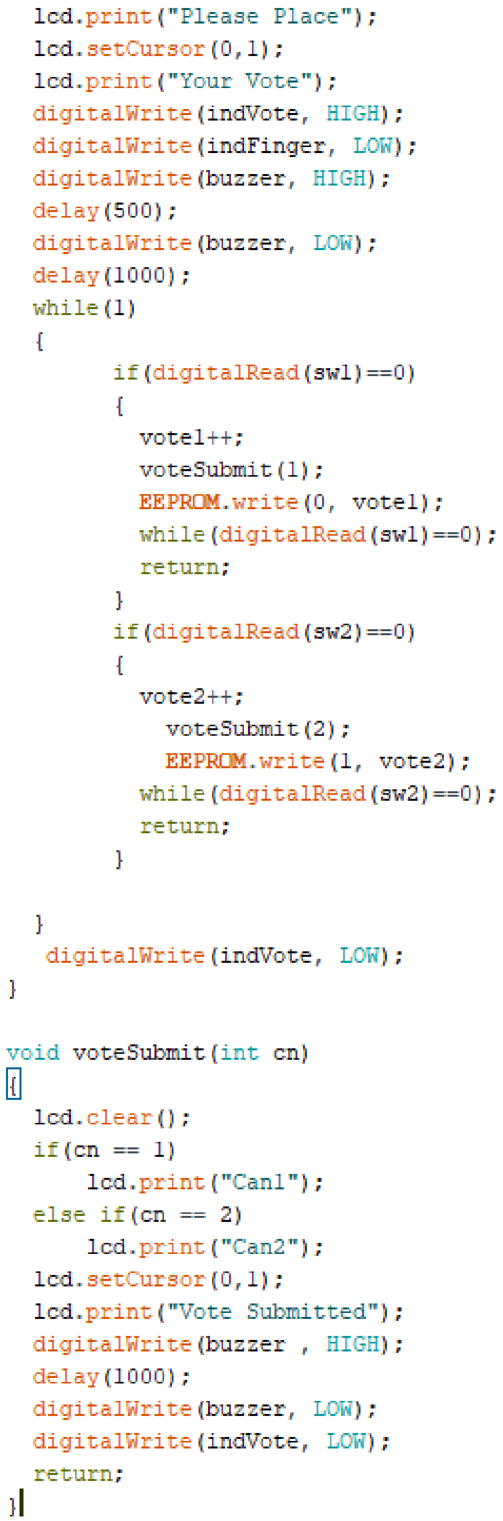
```
    //Serial.println("Image taken");
    break;
  case FINGERPRINT_NOFINGER:
    //Serial.print(".");
    break;
  case FINGERPRINT_PACKETRECIEVEERR:
    //Serial.println("Communication error");
    break;
  case FINGERPRINT_IMAGEFAIL:
    //Serial.println("Imaging error");
    break;
  default:
    //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!");
```



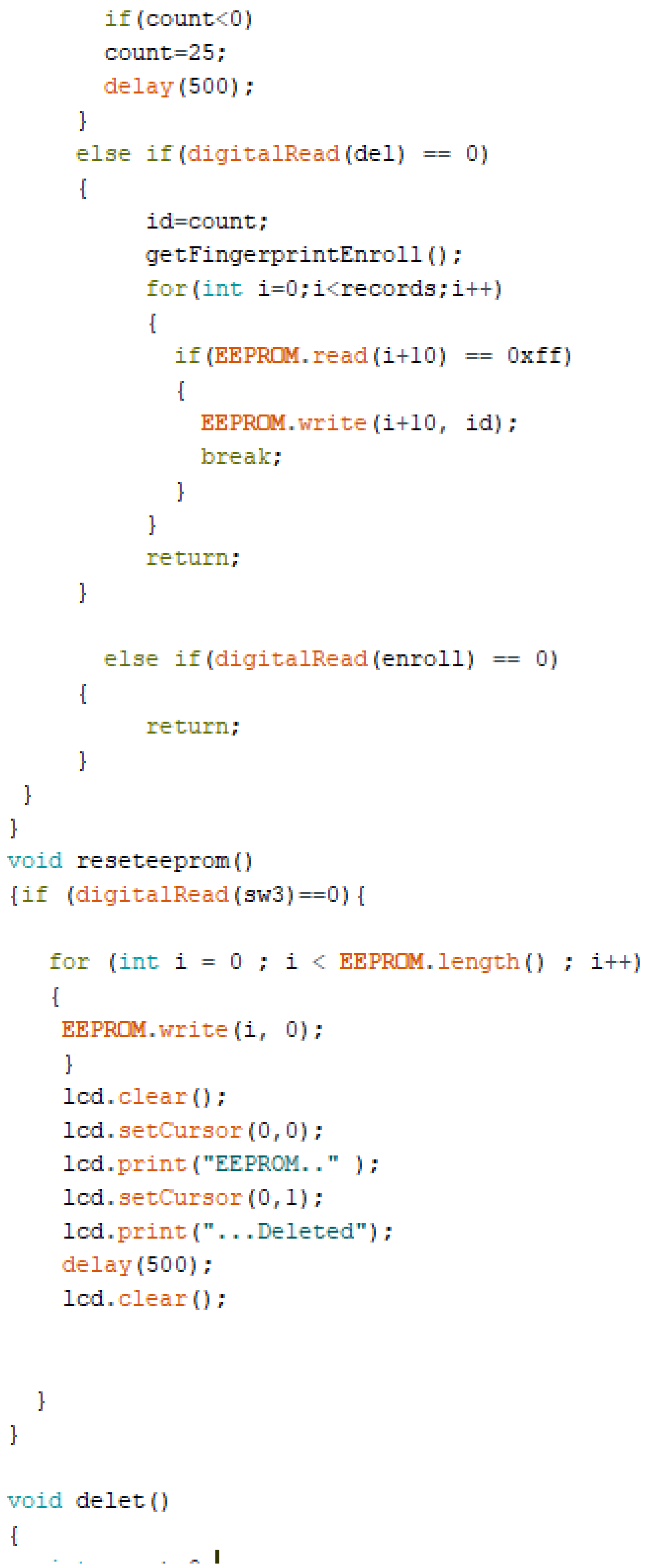
```
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    //Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_ENROLLMISMATCH) {
    //Serial.println("Fingerprints did not match");
    return p;
  } else {
    //Serial.println("Unknown error");
    return p;
  }

  //Serial.print("ID "); //Serial.println(id);
  p = finger.storeModel(id);
  if (p == FINGERPRINT_OK) {
    //Serial.println("Stored!");
    lcd.clear();
    lcd.print("Stored!");
    delay(200);
  } 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 Vote()
{
  lcd.clear();
  lcd.print("Please Place");
  lcd.setCursor(0,1);
```



```
  lcd.print("Please Place");
  lcd.setCursor(0,1);
  lcd.print("Your Vote");
  digitalWrite(indVote, HIGH);
  digitalWrite(indFinger, LOW);
  digitalWrite(buzzer, HIGH);
  delay(500);
  digitalWrite(buzzer, LOW);
  delay(1000);
  while(1)
  {
        if(digitalRead(sw1)==0)
        {
          vote1++;
          voteSubmit(1);
          EEPROM.write(0, vote1);
          while(digitalRead(sw1)==0);
          return;
        }
        if(digitalRead(sw2)==0)
        {
          vote2++;
            voteSubmit(2);
            EEPROM.write(1, vote2);
          while(digitalRead(sw2)==0);
          return;
        }

  }
   digitalWrite(indVote, LOW);
}

void voteSubmit(int cn)
{
  lcd.clear();
  if(cn == 1)
      lcd.print("Can1");
  else if(cn == 2)
      lcd.print("Can2");
  lcd.setCursor(0,1);
  lcd.print("Vote Submitted");
  digitalWrite(buzzer , HIGH);
  delay(1000);
  digitalWrite(buzzer, LOW);
  digitalWrite(indVote, LOW);
  return;
}
```



```
            if(count<0)
            count=25;
            delay(500);
          }
          else if(digitalRead(del) == 0)
          {
               id=count;
               getFingerprintEnroll();
               for(int i=0;i<records;i++)
               {
                 if(EEPROM.read(i+10) == 0xff)
                 {
                   EEPROM.write(i+10, id);
                   break;
                 }
               }
               return;
          }

            else if(digitalRead(enroll) == 0)
          {
               return;
          }
    }
  }
  void reseteeprom()
  {if (digitalRead(sw3)==0){

     for (int i = 0 ; i < EEPROM.length() ; i++)
     {
      EEPROM.write(i, 0);
      }
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("EEPROM.." );
      lcd.setCursor(0,1);
      lcd.print("...Deleted");
      delay(500);
      lcd.clear();


    }
  }

  void delet()
  {
```

Title of the Appendix

Content of the appendix is placed here.