

“Biometric Door Lock Security System”

Submitted By

MD. Golam Kayes

ID: 201016014

Joyanta Biswas

ID: 173016011

A project report submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Electrical and Electronics Engineering

Supervised By

Md. Zesun Ahmed Mia

Lecturer

Department of Electrical and Electronic Engineering



Main Campus

688 Beribadh Road

Mohammadpur

Dhaka - 1207, Bangladesh

Phone: +88-02-223328001-6, +88-02-223361301, +88-02-223361255

Fax: +88-02-223360610

Date of Submission: 23rd June, 2022

Declaration

This project report is submitted to the department of Electrical and Electronics Engineering, University of Liberal Arts Bangladesh in partial fulfillment of the requirements for the degree of Bachelor of Science. So, I hereby, declare that this project report is based on the works and researches done by me. Materials of work found by other researchers are mentioned by reference. This project report, neither in whole, nor in part, has been previously submitted for any degree.

Md. Golam Kayes

ID: 201016014

Email Address: Golam.kayes.eee@ulab.edu.bd

Joyanta Biswas

ID: 173016011

Email Address: Joyanta.biswas.eee@ulab.edu.bd

Supervisor Certification

The project report entitled “**Biometric Door Lock Security System**” is submitted to the department of Electrical and Electronics Engineering, University of Liberal Arts Bangladesh, Dhaka in partial fulfillment of the requirements for the degree of Bachelor of Science.

Dated: 23rd June, 2022

M. Mofazzal Hossain
Professor & Head
Department of Electrical and Electronic Engineering (EEE)
Department of Electronics and Telecommunication Engineering
(ETE)
University of Liberal Arts Bangladesh

Signature & Date

Md. Zesun Ahmed Mia
Lecturer
Department of Electrical and Electronic Engineering
University of Liberal Arts Bangladesh

Signature & Date

ACKNOWLEDGEMENT

The satisfaction and the successful completion of any project would be incomplete without a mention of people who made it possible and whose constant guidance and encouragement all the efforts. This is my final project and that is why this project is very special for me and I took a great interest while completing it.

I would like to thank to my honorable teacher Prof. M. Mofazzal Hossain, Head of the Department of Electrical and Electronic Engineering & Department of Electronics and Telecommunication Engineering, University of Liberal Arts Bangladesh, who arranged this project opportunity for me.

I am truly indebted and wish to express my gratitude to my supervisor Md. Zesun Ahmed Mia, Lecturer, Department of Electrical and Electronic Engineering, University of Liberal Arts Bangladesh for him inspiration, excellent supervision, continuing encouragement and giving confidence and support during every stage of this project work, without which, it would not have been possible for me to complete this project successfully. I also thank him for his insightful comments and suggestions which continually inspired me to explore new dimensions of research.

I would like to thank to my honorable teacher Dr. Mirza Rasheduzzaman, Abul Barkat Mollah Sayeed Ud Doulah, Atik Jawad, Ramit kumar Sadhukhan, Department of Electrical and Electronic Engineering, University of Liberal Arts Bangladesh for their beneficial advices, support and guidance throughout this project for support and helps me to complete my project.

Finally, thank to my parents, family and friends for their help, encouragement and guidance, who without I would not be where I am today.

ABSTRACT

Our “**Biometric Door Lock Security System**” makes our lives smarter, easier, and more secure. The proposed system being controlled by a micro controller has made the project. In simple words, how the system operates is that we take power from the main lines and feed it to an SMPS to output a regulated DC voltage that is to be used to run the Node MCU. Here we will be able to control our door with fingerprint. After recognizing the fingerprint we will be able to open this door very easily. When a person comes close to the door and scan his finger in fingerprint sensor then sensor sense fingerprint. If it is authorized finger then micro controller get a signal and this signal is sent to servo motor. Then servo motor will move and door will be open. After completing the project, we have put it through several tests and although we did face some initial difficulties while making the project but at the end, the results we came up with from the test were quite satisfactory.

TABLE OF CONTANTS

Declaration	II
Supervisor Certification	III
Acknowledgement	IV
Abstract	V
List of Figures	VIII
List of Tables	IX

CHAPTER 1	INTRODUCTION	01-03
------------------	---------------------	--------------

1.1	Introduction	01-02
1.2	Purpose of Research	02
1.3	Project Keywords	02
1.4	Problem to be solve	02
1.5	Objectives	03
1.6	Organization of Book	03
1.7	Instrument Cost Analysis	03

CHAPTER 2	LITERATURE REVIEW	04-06
------------------	--------------------------	--------------

2.1	Introduction	04
2.2	Literature Review	04-05
2.3	Summary	06

CHAPTER 3	METHODOLOGY	07-10
------------------	--------------------	--------------

3.1	Introduction	07
3.2	System Description	07
3.3	Methodology	07
3.4	Block Diagram	08

3.5	Circuit Diagram	08-09
3.6	Working Principle	09-10
3.7	Complete Project Prototype	10
CHAPTER 4	HARDWARE DETAILS	11-30
4.1	Node MCU	11-12
4.2	Fingerprint Sensor	13-19
4.3	Switch Mood Power Supply	19-20
4.4	Capacitor	20-22
4.5	5V Regulator IC	22-23
4.6	Resistor	23-24
4.7	Servo Motor	24-25
4.8	Arduino IDE	26-29
4.9	Power Inverter System	26
4.10	Proteus Software	30
CHAPTER 5	POWER INVERTER SYSTEM	31
5.1	Power Inverter System	31
CHAPTER 6	RESULT & DISCUSSION	32-33
5.1	Result	32-33
5.2	Advantages	33
5.3	Applications	34
5.4	Limitations	34
5.5	Discussion	34
CHAPTER 7	CONCLUSION	35
6.1	Future Scope of Work	35
6.2	Conclusion	35
	Reference	36-37
	Appendix	38-60

List of Figures

Figure 3.1	Block Diagram	08
Figure 3.2	Schematic Diagram	09
Figure 3.3	The Complete Project Prototype	10
Figure 4.1	Node MCU	12
Figure 4.2	Fingerprint Sensor	13
Figure 4.3	Fingerprint Sensor Module	14
Figure 4.4	Fingerprint Structure	16
Figure 4.5	Minutiae Point in a Fingerprint Image	17
Figure 4.6	SMPS Power Supply	19
Figure 4.7	Capacitor	20
Figure 4.8	The Smoothing Capacitor with Full Bridge Rectifier	22
Figure 4.9	5V Regulator IC	23
Figure 4.10	Two Common Resistor Schematic Symbol	24
Figure 4.11	Servo Motor	25
Figure 4.12	OLED Display	25
Figure 4.13	Arduino IDE	27
Figure 4.14	Proteus Software	30
Figure 5.1	Power Inverter System	31
Figure 6.1	Result database	32
Figure 6.2	Result in Excel	33

List of Tables

Table 1	Cost of Components with Price	03
---------	-------------------------------	----

Chapter One

Introduction

1.1 INTRODUCTION

The art of recognizing human face is very difficult because it recognizes different features such as expressions, age, hair transformation, expressing face change, such as security systems, credit card validation, airports, railway stations and households. Security plays an important role in identifying criminals in security. If a face is recognized, then it is familiar with the system and the system may be able to display the current face in a stored database. If the face is not found in the storage database or face the face for the first time, the face is stored in the database for further identification. Application-oriented devices are crafted from different perspectives. Some security the apparatus is concerned about the entry of an unauthorized person into the home, something to worry about gas leakage or fire protection and some are password based or Biometric Lock at the entrance.

In this article, the consideration of the security device is suggested home security and humanitarian efforts. The device does fingerprint-based calling in addition to the doorbell and door locks, indoor users can open the door from anywhere in Home through a remote controller via this device. Takes fingerprint from device person wants to enter the house through a fingerprint scanning module. The module matches that with a fingerprint database in his memory. The biometric composition is one of the best traits because of our skin, Palms and soles flow on each finger's fingertips like a yellow pattern that is unique and Irreversible.

Fingerprint recognition is one of the safest systems because of one of its fingerprints the person never meets others. So unauthorized access may be banned designing a lock that saves and unlocks fingerprints from one or more authorized users System when a match is found. There are situations that can be very frustrating when a person shut themselves down outside his home or office, he leaves his key inside or sometimes the thief just breaks Locks and steals everything. This type of situation always bothers user's Manual door lock with keys, although people use smart cards in some places, it can create a situation when a card is lost or the card is kept inside. Then into another Supervisors are available to lock houses or offices and to secure keys.

People have problems using keys or smart cards. That is when our system come to play its role facial recognition, voice message passing and fingerprint-based lock. The initial feature of the facial image is the gray intensity of each Pixel each face is then converted to a vector of rows by merging it into one row after another. Therefore, local binary pattern histogram (LBPH) has more relevant features / Removal of signature. Local binary pattern histogram (LBPH) is a simple method of data Minimize dimensions while minimizing mineral construction errors LBPH is a data context used to simplify data sets. On a small circuit board, low-cost military-based computer recycling pies control the lock and other sensors. The door locks on the door and opens when there is a face Recognized.

1.2 PURPOSE OF RECHARCH

In this project we have tried to build an automatic security system with Fingerprint Sensor. Which will help to build high security of our house in many ways. In this project the loads of the house can be power by the AC 220V and Fingerprint sensor data will be saved in a data logger system. Which will be helpful for monitor who enter or exit through this system.

1.3 PROJECT KEYWORDS

Home Automation System, Node MCU, Automation, Data Logger System.

1.4 SOLUTION

If anyone is authorized in the fingerprint system, they will not experience any delays. To enter a room. Our proposed fingerprint-based lock system is reliable and extremely secure Lock that will not only ensure a safe environment but also make the lifestyle easier. This system can Very useful in residential buildings, big offices, and universities so on. The lock will be hung on the wall with a fingerprint on it. When an unauthorized person detects, they will create a bell ring and we may be able to see if a person encounters the device, then it is up to us to allow it. Our goal is to provide multi-level security that provides high-level security System.

1.5 OBJECTIVES

We have some specific objectives for this project and they are pointed below:

- To design and construct of a **Biometric Door Lock Security System**.
- To provide a smart door lock system.
- To put the whole system into action so that we can observe its practical effects and verify our work.
- To study the system performance for future reference and improvement purposes.

1.6 ORGANIZATION OF BOOK

This project book consists of five chapter. The first chapter contains the statement of the introduction, our motivation for the project, objectives of the study, our used methodologies and the project organization. Chapter two contains literature review in details. Chapter three describes the proposed system architecture with component details and the software which we have used for our work. Chapter four deals with the circuit design, working principle and shows the complete prototype of the project that we have built. In the final chapter, we have discussed the advantages and applications of the project and also about some aspects we had to overcome while doing the project and lastly, we gave the conclusion of the book.

1.7 INSTRUMENT COST ANALYSIS

Table 1: Cost of Components with Price

In the below table we have summarized our project expenditure.

Sl. No.	Product Name	Specification	Qty.	Unit Price (Taka)	Total Price (Taka)
01.	SMPS	5V, 7 Amp	1	550	550
02.	Node MCU	ESP8266	1	264	264
03.	OLED Display	0.96"	1	486	486
04.	Fingerprint Sensor	R307	1	980	980
05.	Servo Motor	DC	1	119	119
06.	Battery	7.4V	2	240	240
07.	Others			690	690
	Total				3329

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

This chapter is arranged on Literature Review. Here's a look at some of last year's literature, like our project. By reading them, we can overcome the mistakes of the previous project and make a more effective project.

2.2 LITERATURE REVIEW

This section describes various technology surveys used in the secure door locking system. In the old days, only mechanical locks were available that were not sufficiently secure. All relevant work to complete the study, including the development of a GUI-based face recognition system and auto-switching magnetic lock mechanism with micro controller. Auto-switching magnetic door lock system based using micro controller has been developed in 2012. The author concluded that the performance and functionality of the design system can be improved by another feature that automatically captures and create a facial recognition system. Whenever a human being appears at the entrance, make an image with a sensor [1]. In (2018) automatic gate entry system uses LBPH (local binary pattern histograms) which not only increase the level of security but also improve the intelligence level on the building entry system. The performance and accuracy of the system depends entirely on the core processor and the accuracy of the face recognition process [2].

In (2017) a survey on various door lock access control was done which consider five systems for comparative analysis includes Password lock system, RFID, Bio-matric, OTP (one-time password) and cryptography-based lock system. They made a conclusion after the analysis of all five system that single system would not work perfectly therefore in high secure door lock system is required than combination of all these five mechanisms should be utilized [3]. In (2019) a proposed door lock system for critical zone in which they made a security system for high alert areas using two factors authentication with OTP (one-time password) & RFID (radio frequency identification). In their research paper they also made a secure wallet application

which help them to remember passwords and established a secure system for red zone areas which only allow authorized person [4]. A door lock security system has been observed in 2016 in which only one authentication method was present. The main loop hole/issue is the authorization is done with three types of authentications 1) RFID 2) face recognition 3) password, to improve the security of door lock system [5].

In (2017) a comparative analysis of real time face recognition system algorithms using Raspberry-Pi has been done that utilizes the three algorithms i.e., PCA (principal component analysis), LBPH (local binary pattern histogram) & Fisher face. This research concluded that the utilization of Raspberry Pi will removes the dependencies of the underlying hardware platform with respect to time and accuracy. Also, LBPH algorithm has been considered as the benchmark of all facial recognition [6]. In the proposed system, a ZigBee module incorporates digital door lock. Door lock acts as the central controller of the overall home automation system. Since entering and leaving a home, first and foremost, respectively, people can easily monitor and monitor the home environment and state of the home automation function in the digital door lock system before they enter or leave the home. In addition, it also allows users to remotely monitor the state of the home via the Internet or any other public network [7]. The door sensors open the door as soon as the homeowner sends a push notification. The user will receive this notification regardless of whether the phone is locked or unlocked, or if another app has been opened at that time. This was the main purpose of their project, which made the user feel safe. This setup can also be used in commercial offices where some areas are restricted to certain personnel, such a system will immediately notify any unauthorized personnel who try to gain access to the area. So, the expansion and application of such a system is limited by imagination [8].

In (2019) reviewing of various face recognition techniques has been done in which they concluded that PCA (principal component analysis), LDA (linear discriminant analysis), ICA (independent component analysis), A There are various techniques for facial recognition such as NN (artificial nerve). Network (SVM) Support Vector Machine - Compare all these face recognition techniques [9]. In (2019) IOT based real time face recognition door lock system using neural network in this paper face recognition using histogram were used to shown to be accurate and fast. When BPNN techniques combined with PCA the outcome result would be very fast. Captured face would be send to message cloud where the extracted faces deleted after that known and unknown faces would authorize or not [10].

2.3 SUMMARY

We try to do this project by reading the above literature, and we have been able to make our project successful by reducing the mistakes of last year's project.

CHAPTER THREE

METHODOOGY

3.1 INTRODUCTION

In this chapter we describe our project block diagram, circuit diagram, project working principle and final project view.

3.2 SYSTEM DISCRIPTION

In this project, we have used a SMPS which has a Transformer to step down the input ac supply to get our desired value and then using the rectifier circuit and filter inside we get a dc output. Then the regulator IC has been used to output a regulated 5V so that we can use it to run the Node MCU. The readings received from the fingerprint sensor helped the micro controller to make necessary decisions and which in turns helped the Servo to work accordingly. Here also use SMPS, Fingerprint Sensor, Servo motor etc.

3.3 METHODOLOGY

- Creating an idea for design and construction of a **Biometric Door Lock Security System** with the combination of Node MCU micro controller. And designing a block diagram & circuit diagram to know which components we need to construct it.
- Collecting all the components and programming the micro controller to control the whole system.
- Setting up all the components in a PCB board & then soldering. Lastly, assembling all the blocks in a board and to run the system & for checking purposes.

3.4 BLOCK DIAGRAM

In our project we have set up a **Biometric Door Lock Security System**. Here the current from the AC source is entering the DC output circuit with the help of SMPS. In this circuit we have used one Node MCU micro controller for master controller unit. Here we also use a Fingerprint Sensor, Servo Motor etc.

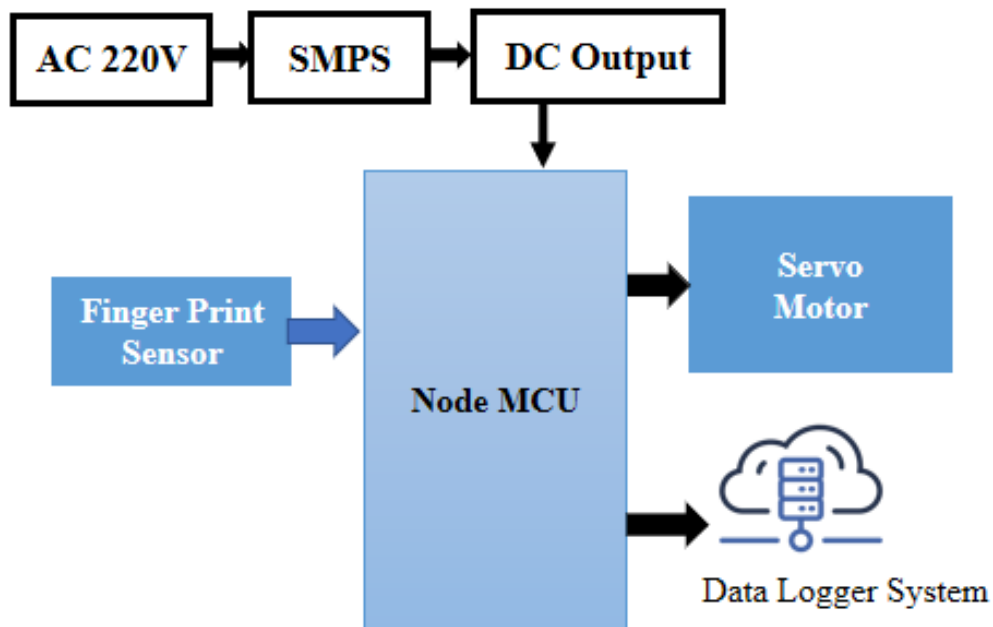


Figure 3.1: Block Diagram of **Biometric Door Lock Security System**

3.5 CIRCUIT DIAGRAM

The schematic diagram here is representing the electrical circuit and the components of the **Biometric Door Lock Security System**. Here we have used standardized symbols and lines.

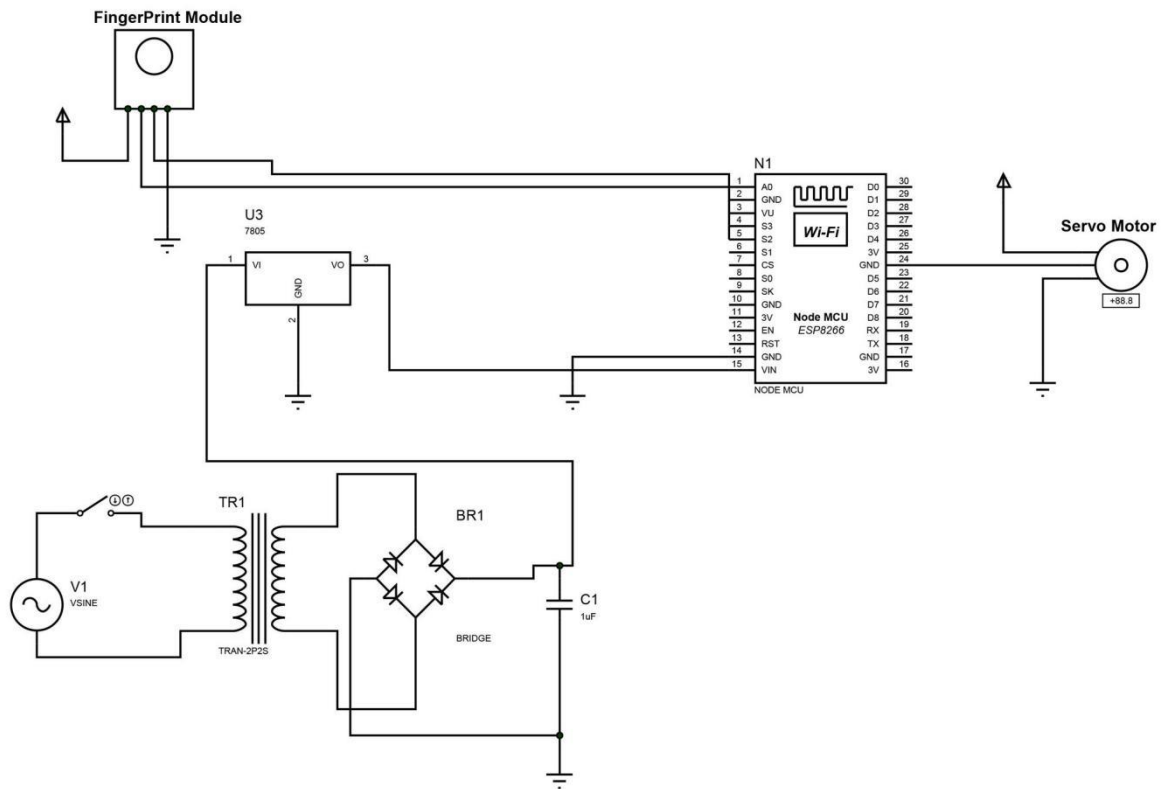


Figure 3.2: Circuit Diagram of Biometric Door Lock Security System.

3.6 WORKING PRINCIPLE

The way **Biometric Door Lock Security System** works is that i take the 220V R.M.S from the supply voltage and then give it to the SMPS, the SMPS has a step-down transformer inside that stepped down that voltage down to 12V. This 12V supply is still alternating in nature so now, we use a Full Wave Bridge Rectifier to rectify that output. Now we use Capacitors to smooth the curve and turn the 12V pulsating current to pure 12V direct current. But this 12V dc is still much high for our Node MCU so we use a 7805 5V Regulator IC to make that voltage come down to 5V. All of this happens inside the SMPS. We feed this voltage to the micro controller for it to run. Here we also use Fingerprint sensor, servo motor or magnetic door lock etc.

The micro controller controls the whole system and also takes command from the user through an app as well. Here our door will be able to control with the help of fingerprint. Recognize Fingerprint will be able to open this door very easily. When a human scan finger then it will detect the fingerprint from sensor. If the fingerprint is recognized then servo motor will move

and door will be open. All finger scan entrance/exit data will be saved in data logger system. In brief, this is how the system works.

3.7 COMPLETE PROJECT PROTOTYPE

The complete prototype of our project is shown below:

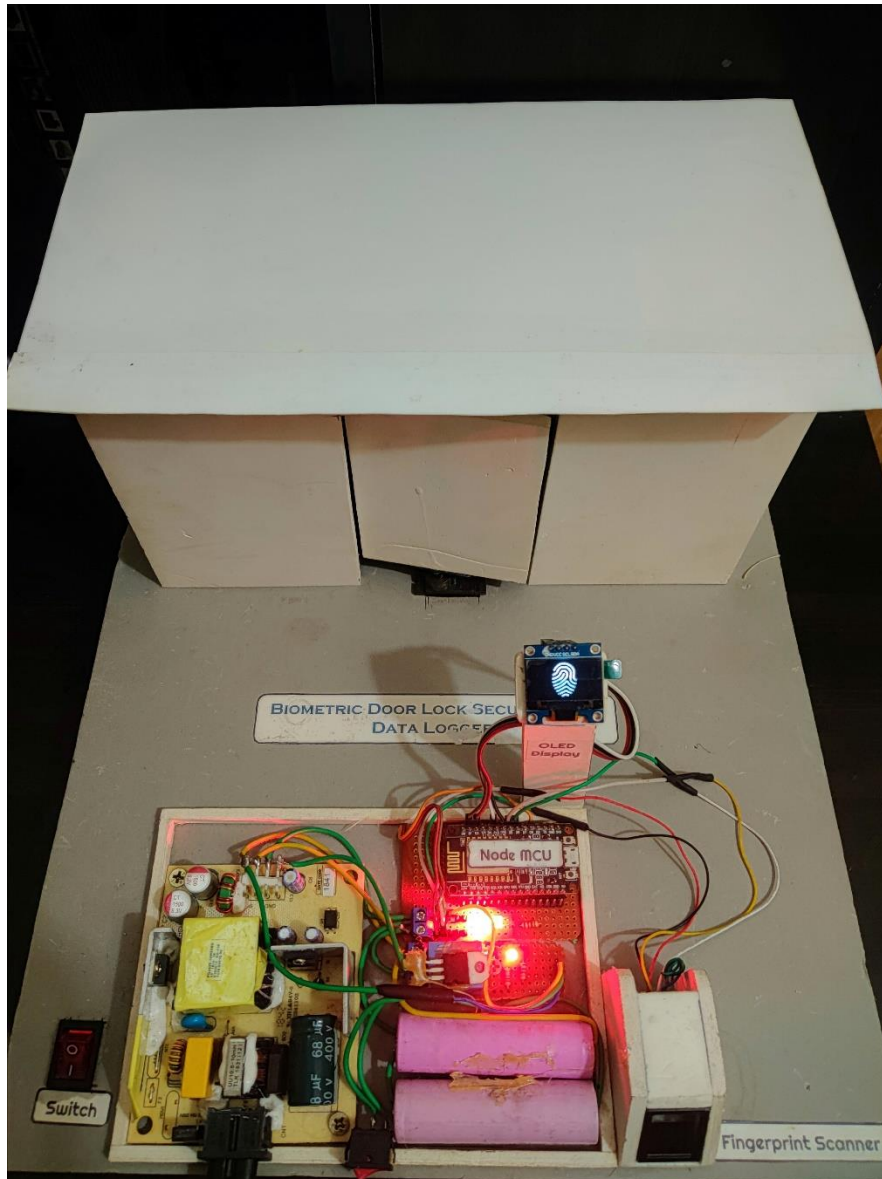


Figure 3.3: The Complete Prototype of the Project (Top View)

CHAPTER FOUR

HARDWARE COMPONENTS

4.1 NodeMCU

NodeMCU is an open-source firmware for which open-source prototyping board designs are available. The name "NodeMCU" combines "node" and "MCU" (micro-controller unit). The term "NodeMCU" strictly speaking refers to the firmware rather than the associated development kits. Both the firmware and prototyping board designs are open source.

The firmware uses the Lua scripting language. The firmware is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open-source projects, such as lua-cjson and SPIFFS. Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit ESP32 has also been implemented.

The prototyping hardware typically used is a circuit board functioning as a dual in-line package (DIP) which integrates a USB controller with a smaller surface-mounted board containing the MCU and antenna. The choice of the DIP format allows for easy prototyping on breadboards. The design was initially was based on the ESP-12 module of the ESP8266, which is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications.

Features:

Wi-Fi Module – ESP-12E module similar to ESP-12 module but with 6 extra GPIOs. USB – micro-USB port for power, programming and debugging Headers – 2x 2.54mm 15-pin header with access to GPIOs, SPI, UART, ADC, and power pins Misc. – Reset and Flash buttons Power – 5V via micro USB port Dimensions – 49 x 24.5 x 13mm Node MCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects). Node MCU started on 13 Oct 2014, when Hong committed the first file of NodeMCU-firmware to GitHub. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported MQTT client library from Contac to the ESP8266 SoC platform, and committed to

Node MCU project, then Node MCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to Node MCU project, enabling Node MCU to easily drive LCD, Screen, OLED, even VGA displays. In summer 2015 the creators abandoned the firmware project and a group of independent contributors took over. By summer 2016 the Node MCU included more than 40 different modules. Due to resource constraints users need to select the modules relevant for their project and build a firmware tailored to their needs.

ESP8266 Arduino Core:

As Arduino.cc began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to support alternate toolchains to allow Arduino C/C++ to be compiled for these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 WiFi SoC, popularly called the "ESP8266 Core for the Arduino IDE". This has become a leading software development platform for the various ESP8266-based modules and development boards, including Node MCUs.

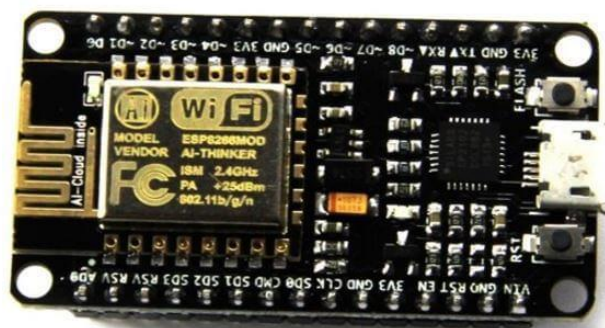


Figure 4.1: Node MCU

4.2 Fingerprint Sensor

R305 fingerprint module is fingerprint sensor with TTL UART interface for direct connections to micro controller UART or to PC through MAX232 / USB-Serial adapter. The user can store the fingerprint data in the module and can configure it in 1:1 or 1: N mode for identifying the person. The FP module can directly interface with 3v3 or 5v Micro controller. A level converter (like MAX232) is required for interfacing with PC serial port. Optical biometric fingerprint reader with great features and can be embedded into a variety of end products, such as access control, attendance, safety deposit box, car door locks.



Figure 4.2: Finger print Sensor.

Features

- Integrated image collecting and algorithm chip together, All-in-one
- Fingerprint reader can conduct secondary development, can be 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.

Specifications of R305 Finger Print Module:

- Fingerprint sensor type: Optical.
- Sensor Life: 100 million times.
- Static indicators: 15KV Backlight: bright green.
- RS232 communication baud rate: 4800 BPS~115200BPS changeable.
- Dimension: 55*32*21.5mm.
- Image Capture Surface 15—18(mm).
- Verification Speed: 0.3 sec.
- Scanning Speed: 0.5 sec.
- Character file size: 256 bytes.
- Template size: 512 bytes.
- Storage capacity: 250.
- Security level: 5 (1, 2, 3, 4, 5(highest)).
- False Acceptance Rate (FAR):0.0001%.
- False Rejection Rate (FRR): 0.1%.
- Resolution 500 DPI.
- Voltage: 3.6-6.0 VDC.



Figure 4.3: Finger Print Sensor

Fingerprint sensor working principle

Fingerprint sensors in the fingerprint recognition systems serve as an important component of the data acquisition subsystem. A fingerprint recognition system needs a highly precise digital image of user fingerprints and fingerprint sensors to help them acquire that. Fingerprint sensors may have different sensing mechanisms such as optical, capacitive, ultrasonic, etc., which leverages different technologies to capture a fingerprint image. Despite the technological differences in sensing and creating the fingerprint image, all fingerprint sensors have a common objective: to capture a high-quality fingerprint image.

Optical image sensors are more or less like image sensors used in digital photography. They are divided into several pixels that capture information of an image using visible light. The image can be focused on the sensor with the help of lenses. Capacitive sensors, on the other hand, make use of human skin conductivity to create a fingerprint image. A grid or array of several tiny capacitors is used in these sensors. They are so small that they can act as good as a single pixel of an optical scanner. Skin conductivity differs in finger ridges and valleys and this difference is captured by the grid of tiny capacitors, creating a fingerprint image depending on the internal programming.

Ultrasonic sensors make use of ultrasonic sound waves to create a fingerprint image. The ultrasonic fingerprint sensor working principle is based on the detection of ultrasonic sound waves that are reflected from the skin surface. The ultrasonic sensor can create a 3D image of user fingerprints and is considered highly secure than other fingerprint sensor technologies. Regardless of the difference in sensing technology and working mechanism, the fingerprint sensor working principle focuses on high-quality fingerprint image capture.

Fingerprint matching

The fingerprint matching can be carried out using different methods. Transform-based methods typically use two-dimensional Fourier Transforms and Howe Transforms applied to the pixel array representing the fingerprint.

The idea is to mathematically transform the image in some way and then compare the coefficients of the transformed images. In this context, the features of fingerprints are

conversion factors. A standard has been developed for the transmission and storage of fingerprints using a transformation-based method. Correlation-based methods take into account that fingerprints and their representative matrices obtained from a scanner cannot simply be superimposed due to all the differences. However, small areas of two fingerprints may be correlated when superimposed. If the geometric relationships between the centers of small areas remain roughly the same when superimposed to maximize the correlation between the two images, the images may be related to the same papillary ridges of the finger.

Minutiae Algorithm

The matching of two fingerprint images is based on the numbering of minutiae and bifurcation points. Minutia-based methods tend to emulate what forensic experts do. Minutiae-based algorithms only extract local features of a fingerprint image. A fingerprint image contains minutiae points, core points, ridges, valleys, background area, foreground area, local features as well as global features. Ridges are represented by black lines while valleys are seen as white lines in a typical fingerprint image. The core point is the central area of a fingerprint image, while minutiae points are seen as small lines.

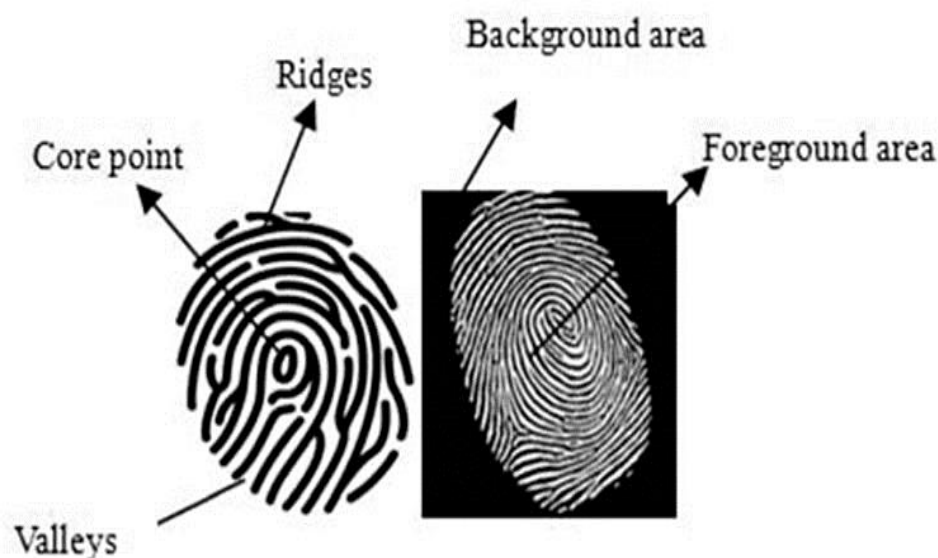


Figure 4.4: Fingerprint Structure

There are mainly eight types of minutiae points in a fingerprint image which are:

- Ridge ending
- Bridge
- Crossover
- Dot
- Bifurcation
- Hook
- Delta
- Enclosure

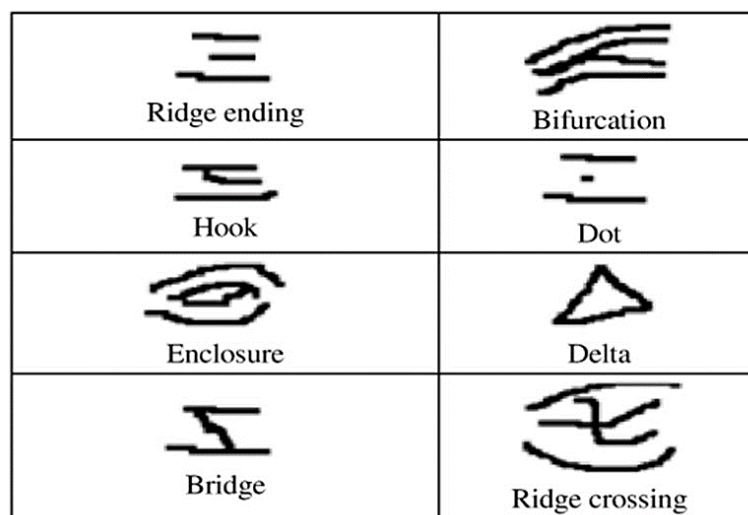


Figure 4.5: Minutiae points in a fingerprint image.

Minutia points also have a direction associated with the ridge at the point of their origin. A mathematical algorithm moves through the image looking for ridges where they separate or end, and maps the minutiae.

When comparing two fingerprints, minutia cards are placed one above the other and rotated/moved relative to one another. In this case, if a certain number of minutiae are obtained that coincide in position and direction, it is considered a coincidence. print identification includes local features of the fingerprint image. The two most prominent local ridge characteristics are ridge ending and ridge bifurcation. A ridge ending fingerprint is the one in which a ridge ends abruptly. When a ridge forks or diverges into branch ridges, it is called ridge bifurcation. Collectively they are called minutiae. Minutiae-based fingerprint identification algorithm recognizes fingerprints with the help of valid bifurcation points. The

valid bifurcation points are extracted using the cross-numbering method and spurious minutiae are removed by applying false minutiae removal techniques.

Image enhancement

For enhancing a fingerprint image intensity adjustment is on the basis of the threshold value. First, obtain the threshold of an image and then based on the threshold, decide whether to adjust the intensity or not. If the intensity is to be adjusted, how much it should be.

Binarization

Here binarization is done with NOT operation which changes the value and color of ridges and furrows. Here ridges are represented by white or 1's and furrows are represented by black or 0's.

Extracting ROI and ridge thinning in fingerprint recognition

OPEN and CLOSE operation is done to extract regions of interest from binary images. Open operations remove noise in the background and close operation shrinks images to remove cavities. Ridge thinning makes the ridges just one pixel wide without changing their basic structure. It helps in removing the redundant pixels.

Minutiae extraction

A 3×3-pixel window is used to extract the minutiae points. The window contains 0s and 1s values called pixels. If the center pixel having value 1 corresponds to its three neighbors having value 1 then it is called bifurcation point otherwise it is a ridge ending point. A ridge ending fingerprint is the one in which a ridge ends abruptly. This method of minutiae extraction is relatively simple, which may be prone to extracting false bifurcation points. After all the bifurcation points are examined, we can apply the false point removal technique.

If any of these points satisfy the bifurcation condition (3 neighboring 1s) that the point is removed and the process is repeated for all bifurcation points. This is achieved by examining four neighboring pixels – one above, one below, one on the right, and one on the left of a bifurcation point. After removing these false minutiae points, valid bifurcations were deleted. So to avoid it, the neighboring bifurcation points are deleted, which are just 1 pixel close to each other i.e. there are n 1-pixel close bifurcation points then delete n-1 points. If a 4×4 pixel

window is taken as shown in the table above, then extra neighbors are detected with value one, which creates false minutiae points.

Removal of false minutiae

False minutiae are the points that occur because of breaks, noise, gaps, etc. Sometimes, fingerprint image enhancement process enables removal of false minutiae, so the false minutiae points are removed during the post-processing of the fingerprint image.

False minutiae points are removed in the following steps:

1. First, calculate the average distance D between two neighbor's ridges.
2. Then calculate the distance between two minutiae points. Let's call this distance d .
3. If d is less than D and the two minutiae points are in the same ridge then remove both of them.
4. After the removal of false minutiae points, mark the valid minutiae points and the image is matched on the basis of these minutiae points.

4.3 Switch Mode Power Supply (SMPS)

A switched-mode power supply is an electronic power supply that incorporates a switching regulator to convert electrical power efficiently. Like other power supplies, an SMPS transfers power from a DC or AC source to DC loads, such as a personal computer, while converting voltage and current characteristics.



Figure 4.6: SMPS Power Supply

A switched-mode power supply (switching-mode power supply, switch-mode power supply, switched power supply, SMPS, or switcher) is an electronic power supply that incorporates a switching regulator to convert electrical power efficiently. Like other power supplies, an SMPS transfers power from a DC or AC source (often mains power) to DC loads, such as a personal computer, while converting voltage and current characteristics. Unlike a linear power supply, the pass transistor of a switching-mode supply continually switches between low-dissipation, full-on and full-off states, and spends very little time in the high dissipation transitions, which minimizes wasted energy. Ideally, a switched-mode power supply dissipates no power. Voltage regulation is achieved by varying the ratio of on-to-off time. In contrast, a linear power supply regulates the output voltage by continually dissipating power in the pass transistor. This higher power conversion efficiency is an important advantage of a switched-mode power supply. Switched-mode power supplies may also be substantially smaller and lighter than a linear supply due to the smaller transformer size and weight.

4.4 Capacitor

When there is a potential difference across the conductors (e.g., when a capacitor is attached across a battery), an electric field develops across the dielectric, causing positive charge (+Q) to collect on one plate and negative charge (-Q) to collect on the other plate. If a battery has been attached to a capacitor for a sufficient amount of time, no current can flow through the capacitor. However, if an accelerating or alternating voltage is applied across the leads of the capacitor, a displacement current can flow.



Figure 4.7: Capacitor.

An ideal capacitor is characterized by a single constant value for its capacitance. Capacitance is expressed as the ratio of the electric charge (Q) on each conductor to the potential Difference (V). The SI unit of capacitance is the farad (F), which is equal to one coulomb per volt (1 C/V). Typical capacitance values range from about 1 pF (10^{-12} F) to about 1 mF (10^{-3} F). The capacitance is greater when there is a narrower separation between conductors and when the conductors have a larger surface area.

In practice, the dielectric between the plates passes a small amount of leakage current and also has an electric field strength limit, known as the breakdown voltage. The conductors and leads introduce an undesired inductance and resistance. Capacitors are widely used in electronic circuits for blocking direct current while allowing alternating current to pass. In analog filter networks, they smooth the output of power supplies. In resonant circuits they tune radios to particular frequencies. In electric power transmission systems, they stabilize voltage and power flow.

The full-wave bridge rectifier however, gives us a greater mean DC value ($0.637\text{ V}_{\text{max}}$) with less superimposed ripple while the output waveform is twice that of the frequency of the input supply frequency. We can improve the average DC output of the rectifier while at the same time reducing the AC variation of the rectified output by using smoothing capacitors to filter the output waveform. Smoothing or reservoir capacitors connected in parallel with the load across the output of the full wave bridge rectifier circuit increases the average DC output level even higher as the capacitor acts like a storage device as shown below. Too low a capacitance value and the capacitor has little effect on the output waveform. But if the smoothing capacitor is sufficiently large enough (parallel capacitors can be used) and the load current is not too large, the output voltage will be almost as smooth as pure DC.

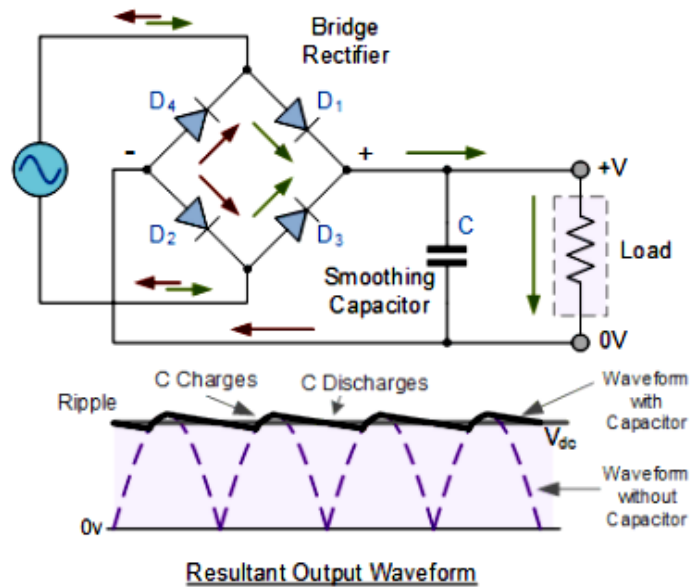


Figure 4.8: The Smoothing Capacitor with Full Bridge Rectifier.

4.5 5V Regulator IC

Voltage sources in a circuit may have fluctuations resulting in not providing fixed voltage outputs. A voltage regulator IC maintains the output voltage at a constant value. 7805 IC, a member of 78xx series of fixed linear voltage regulators used to maintain such fluctuations, is a popular voltage regulator integrated circuit (IC). The xx in 78xx indicates the output voltage it provides. 7805 IC provides +5 volts regulated power supply with provisions to add a heat sink.

7805 IC Rating

- Input voltage range 7V- 35V
- Current rating $I_c = 1A$
- Output voltage range $V_{Max}=5.2V$, $V_{Min}=4.8V$

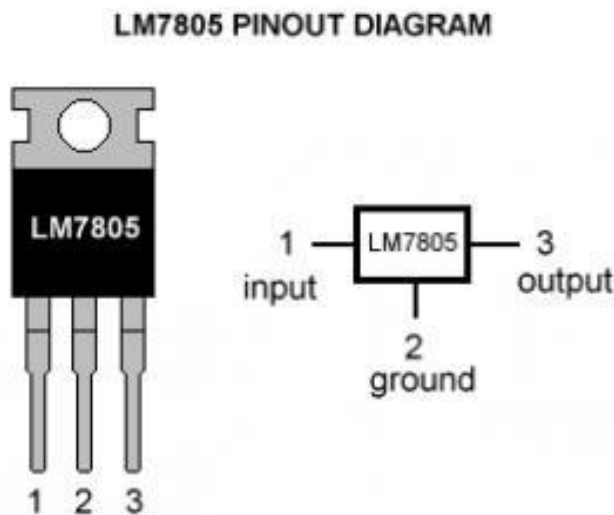


Figure 4.9: 5V Regulator IC

4.6 Resistor

Resistors are electronic components which have a specific, never-changing electrical resistance. The resistor's resistance limits the flow of electrons through a circuit. They are passive components, meaning they only consume power (and can't generate it). Resistors are usually added to circuits where they complement active components like op-amps, microcontroller, and other integrated circuits. Commonly resistors are used to limit current, divide voltages, and pull-up I/O lines.

The electrical resistance of a resistor is measured in ohms. The symbol for an ohm is the greek capital-omega: Ω . The (somewhat roundabout) definition of 1Ω is the resistance between two points where 1 volt (1V) of applied potential energy will push 1 ampere (1A) of current. As SI units go, larger or smaller values of ohms can be matched with a prefix like kilo-, mega-, or giga-, to make large values easier to read. It's very common to see resistors in the kilo ohm ($k\Omega$) and mega ohm ($M\Omega$) range (much less common to see milliohm ($m\Omega$) resistors). For example, a $4,700\Omega$ resistor is equivalent to a $4.7k\Omega$ resistor, and a $5,600,000\Omega$ resistor can be written as $5,600k\Omega$ or (more commonly as) $5.6M\Omega$. All resistors have two terminals, one connection on each end of the resistor. When modeled on a schematic, a resistor will show up as one of these two symbols:

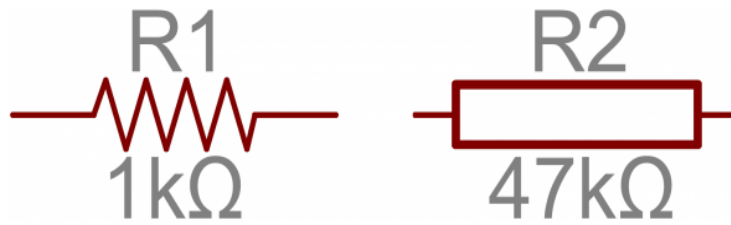


Figure 4.10: Two common resistor schematic symbols

4.7 Servo Motor

A servomotor is a closed-loop servomechanism that uses position feedback to control its motion and final position. The input to its control is a signal (either analogue or digital) representing the position commanded for the output shaft.

The motor is paired with some type of position encoder to provide position and speed feedback. In the simplest case, only the position is measured. The measured position of the output is compared to the command position, the external input to the controller. If the output position differs from that required, an error signal is generated which then causes the motor to rotate in either direction, as needed to bring the output shaft to the appropriate position. As the positions approach, the error signal reduces to zero and the motor stops.

The very simplest servomotors use position-only sensing via a potentiometer and bang-bang control of their motor; the motor always rotates at full speed (or is stopped). This type of servomotor is not widely used in industrial motion control, but it forms the basis of the simple and cheap servos used for radio-controlled models.

More sophisticated servomotors use optical rotary encoders to measure the speed of the output shaft and a variable-speed drive to control the motor speed. Both of these enhancements, usually in combination with a PID control algorithm, allow the servomotor to be brought to its commanded position more quickly and more precisely, with less overshooting.



Figure 4.11: Servo Motor

4.8 OLED Display

These displays are small, only about 1.3" diagonal, but very readable due to the high contrast of an OLED display. This display is made of 128x64 individual white OLED pixels, each one is turned on or off by the controller chip. Because the display makes its own light, no backlight is required. This reduces the power required to run the OLED and is why the display has such high contrast; we really like this miniature display for its crispness.



Figure 4.12: OLED Display

Details:

- Diagonal Screen Size : 1.30"
- Number of Pixels : 128×64

- Color Depth: Monochrome (White)
- Module Construction: COG
- Module Size (mm) : 34.50 x 35.00
- Panel Size (mm) : 34.50 x 23.00 x 1.45
- Active Area (mm) : 29.420 x 14.70
- Pixel Pitch (mm) : 0.23 x 0.23
- Pixel Size (mm) : 0.21 x 0.21
- Weight (g) : 2.18
- Duty : 1/64
- Brightness (cd/m2) : 100 (Type) @ 12V

4.9 Arduino IDE

The digital micro controller unit named as Arduino Nano can be programmed with the Arduino software IDE. There is no any requirement for installing other software rather than Arduino. Firstly, Select "Arduino Nano from the Tools, Board menu (according to the micro controller on our board). The IC used named as ATmega328 on the Arduino Nano comes pre burned with a boot loader that allows us to upload new code to it without the use of an external hardware programmer.

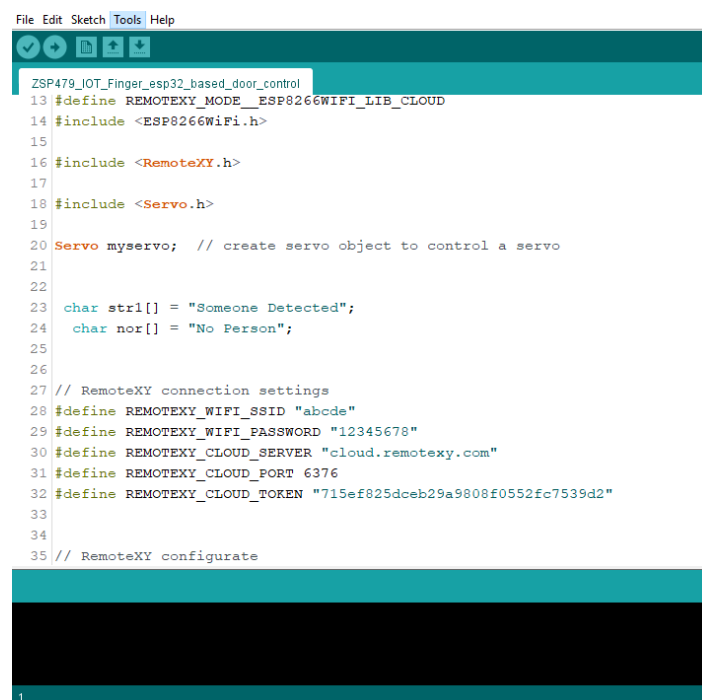
Communication is using the original STK500 protocol (reference, C header files). We can also bypass the boot loader and programs the micro controller through the ICSP (In Circuit Serial Programming) header. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available. The ATmega16U2/8U2 is loaded with a DFU boot loader, which can be activated by:

On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

The Arduino Nano is one of the latest digital microcontroller units and has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The

ATmega328 provides UART TTL at (5V) with serial communication, which is available on digital pins 0 -(RX) for receive the data and pin no.1 (TX) for transmit the data. An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an .in file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board.

The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial Communication on pins 0 and 1). A Software Serial library allows for serial communication on any of the Nano's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus. Arduino programs are written in C or C++ and the program code written for Arduino is called sketch. The Arduino IDE uses the GNU tool chain and AVR Lab to compile programs, and for uploading the programs it uses argued. As the Arduino platform uses Atmel microcontrollers, Atmel's development environment, AVR Studio or the newer Atmel Studio, may also be used to develop software for the Arduino.



```

File Edit Sketch Tools Help
ZSP479_IOT_Finger_esp32_based_door_control
13 #define REMOTEXY_MODE_ESP8266WIFI_LIB_CLOUD
14 #include <ESP8266WiFi.h>
15
16 #include <RemoteXY.h>
17
18 #include <Servo.h>
19
20 Servo myservo; // create servo object to control a servo
21
22
23 char str1[] = "Someone Detected";
24 char nor[] = "No Person";
25
26
27 // RemoteXY connection settings
28 #define REMOTEXY_WIFI_SSID "abcde"
29 #define REMOTEXY_WIFI_PASSWORD "12345678"
30 #define REMOTEXY_CLOUD_SERVER "cloud.remotexy.com"
31 #define REMOTEXY_CLOUD_PORT 6376
32 #define REMOTEXY_CLOUD_TOKEN "715ef825dceb29a9808f0552fc7539d2"
33
34
35 // RemoteXY configurate
  
```

Figure 4.13: Arduino Software Interface IDE

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

Writing Sketches

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension. `.ino`. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

Uploading

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Port menus. The boards are described below. On the Mac, the serial port is probably something like `/dev/tty.usbmodem241` (for an Uno or Mega2560 or Leonardo) or `/dev/tty.usbserial-1B1` (for a Duemilanove or earlier USB board), or `/dev/tty.USA19QW1b1P1.1` (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be `/dev/ttyACMx`, `/dev/ttyUSBx` or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the Sketch menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino bootloader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will

blink the on-board (pin 13) LED when it starts (i.e., when the board resets).

Libraries

Libraries provide extra functionality for use in sketches, e.g., working with hardware or manipulating data. To use a library in a sketch, select it from the Sketch > Import Library menu. This will insert one or more `#include` statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its `#include` statements from the top of your code.

There is a list of libraries in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch. See these instructions for installing a third-party library.

Third-Party Hardware

Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its own sub-directory. (Don't use "Arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

Serial Monitor

This displays serial sent from the Arduino board over USB or serial connector. To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down menu that matches the rate passed to Serial. Begin in your sketch. Note that on Windows, Mac or Linux the board will reset (it will rerun your sketch) when you connect with the serial monitor. Please note that the Serial Monitor does not process control characters; if your sketch needs a complete management of the serial communication with control characters, you can use an external terminal program and connect it to the COM port assigned to your Arduino board. You can also talk to the board from Processing, Flash, MaxMSP, etc (see the interfacing page for details).

4.10 Proteus Software

The Proteus Design Suite is a proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronics design engineers and technicians to create schematics and electronics prints for manufacturing printed circuit boards.

The first version of what is now the Proteus Design Suite was called PC-B and was written by the company chairman, John Jameson, for DOS in 1988. Schematic Capture support followed in 1990 with a port to the Windows environment shortly thereafter. Mixed mode SPICE Simulation was first integrated into Proteus in 1996 and microcontroller simulation then arrived in Proteus in 1998. Shape based auto routing was added in 2002 and 2006 saw another major product update with 3D Board Visualization. More recently, a dedicated IDE for simulation was added in 2011 and MCAD import/export was included in 2015. Support for high-speed design was added in 2017. Feature led product releases are typically biannual, while maintenance-based service packs are released as required.

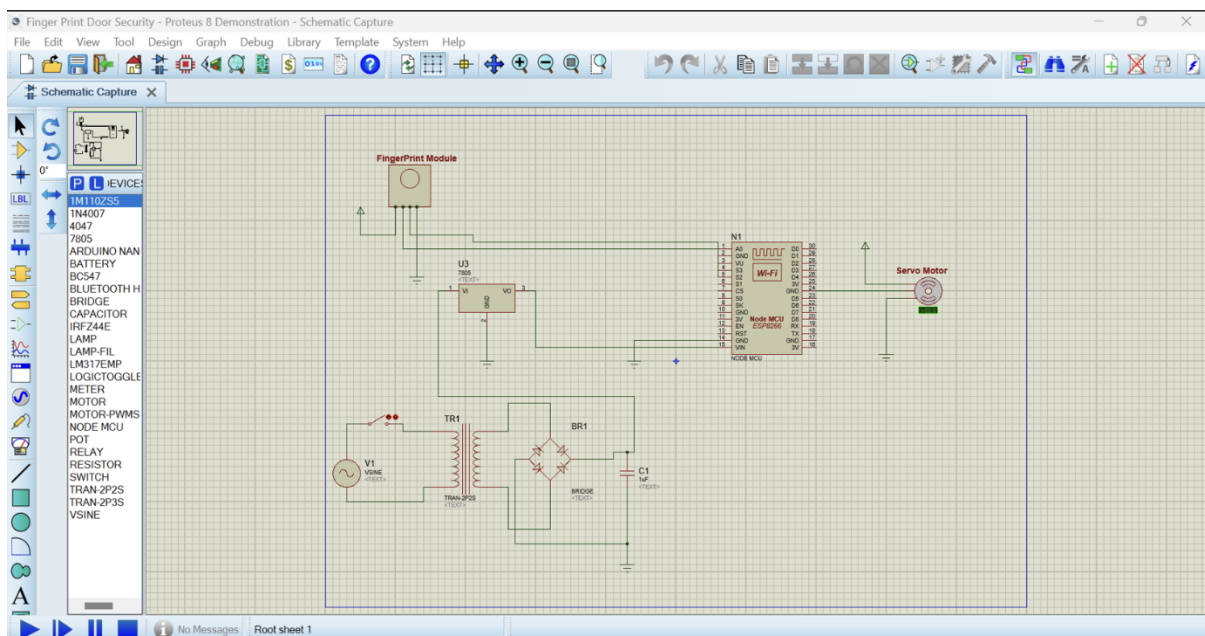


Figure 4.14: Our Project Design in Proteus Software

CHAPTER FIVE

POWER INVERTER SYSTEM

There are two batteries. Each battery has 3.7 volts. The two batteries together are 7.4 volts. Battery output is converted to 5 volts by 7.4-volt battery relay. Because the rest of the component needs 5 volts to run. This relay is fitted because the current will run on the battery automatically when the current goes out and the current is actually connected to the battery and the current output. For the relay has two ends. Normally and close. Normally it is always open and closed. And what is normally closed is always in a short state.

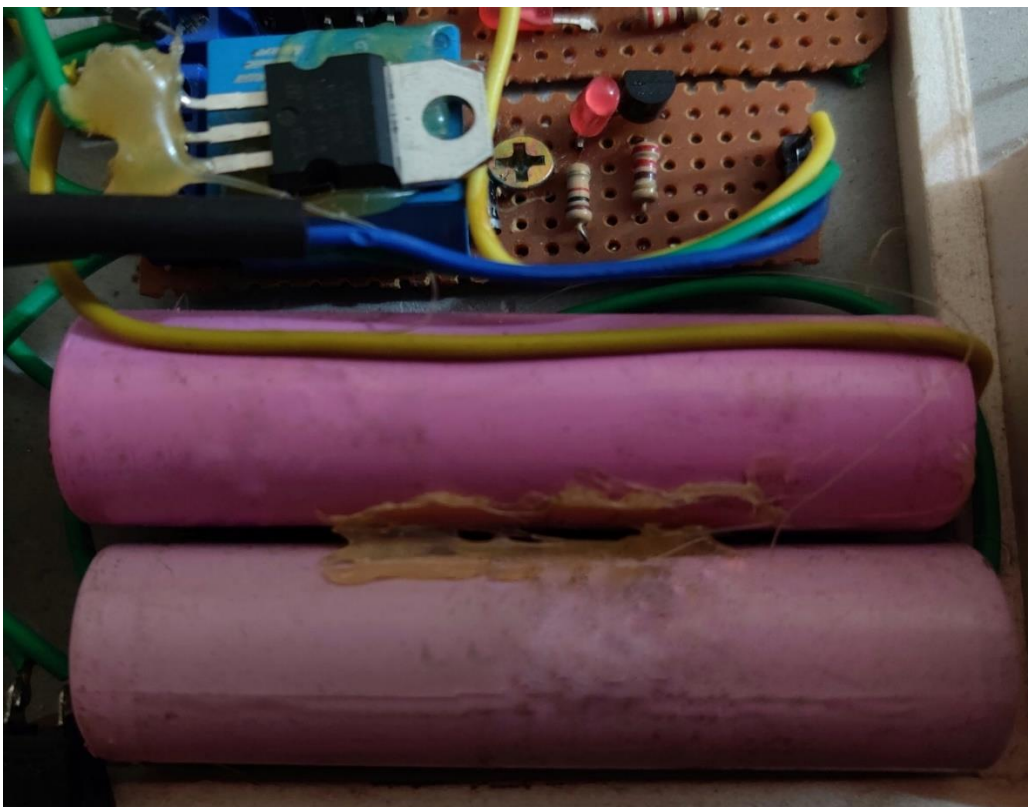


Figure 5.1: Power Inverter System

CHAPTER SIX

RESULT & DISCUSSION

5.1 Results

After making our project we observe it very careful. It works as we desire. Our project give output perfectly and all equipment are work perfectly. We check how much it works and we get perfect output from this project.

- First way is the fingerprint. Only authorized people can enter through this system. When an authorized person scan finger then the fingerprint sensor module recognized the finger print then door will be open by the command of micro controller.

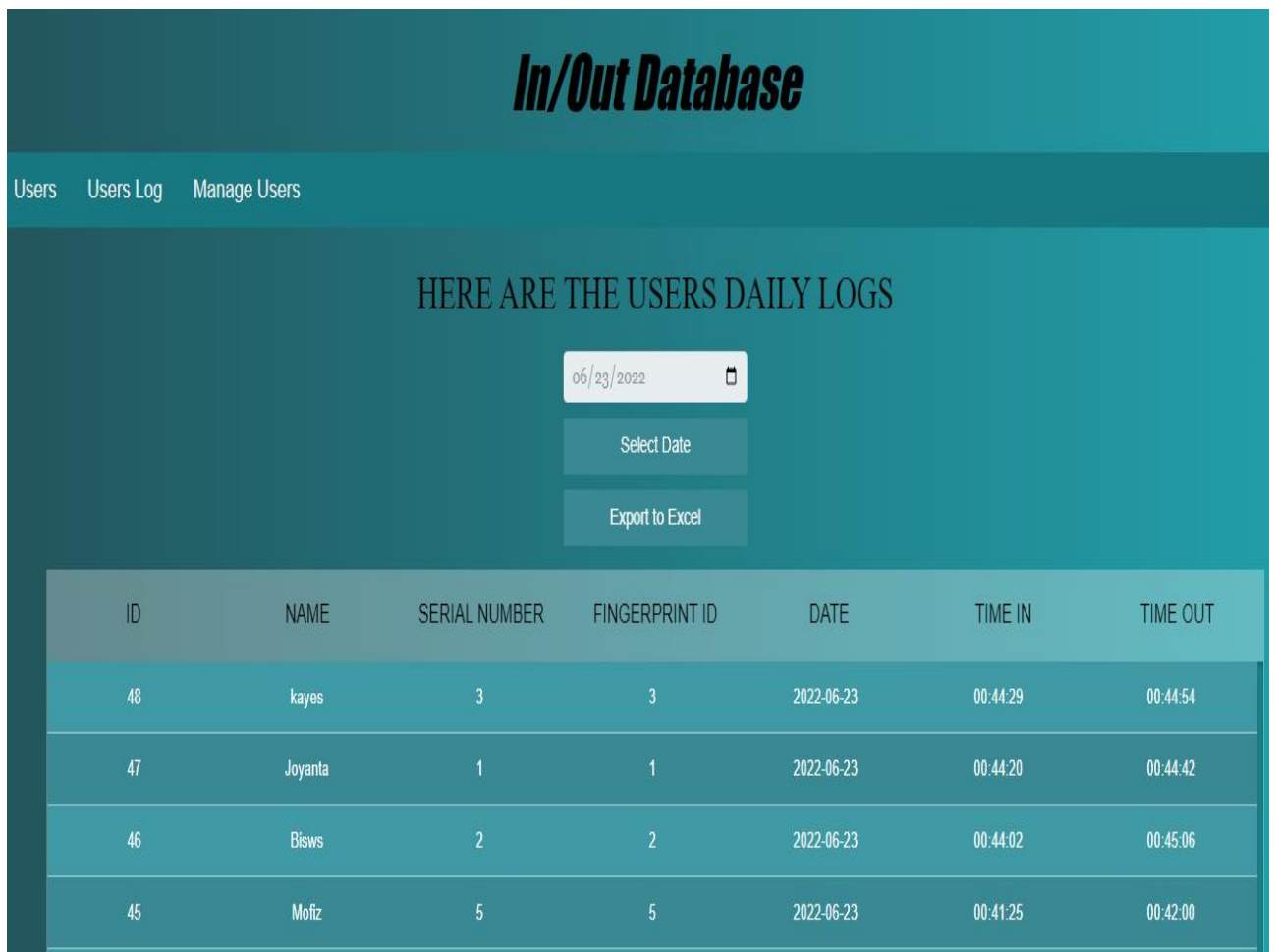


Figure: 6.1: In and Out Database

	A	B	C	D	E	F	G
1	Entry	Name	Serial	Fingerprint ID	Date log	Time In	Time Out
2	Exit		Number				
2	48	kayes	3	3	6/23/2022	0:44:29	0:44:54
3	47	Joyanta	1	1	6/23/2022	0:44:20	0:44:42
4	46	Bisws	2	2	6/23/2022	0:44:02	0:45:06
5	45	Mofiz	5	5	6/23/2022	0:41:25	0:42:00
6	44	Joyanta	1	1	6/23/2022	0:41:06	0:42:13
7	43	Mofiz	5	5	6/23/2022	0:40:46	0:40:55
8	42	Biswas	6	6	6/23/2022	0:33:46	0:34:24
9	41	Joyanta	1	1	6/23/2022	0:33:30	0:34:35
10	40	Biswas	6	6	6/23/2022	0:32:59	0:33:21
11	39	Biswas	6	6	6/23/2022	0:32:46	0:32:54
12	38	Joyanta	1	1	6/23/2022	0:28:22	0:28:34
13	37	Mofiz	5	5	6/23/2022	0:28:00	0:28:09
14	36	Joyanta	1	1	6/23/2022	0:20:47	0:21:50
15	35	kayes	3	3	6/23/2022	0:20:33	0:22:06
16	34	Joyanta	1	1	6/23/2022	0:12:35	0:13:20
17	33	kayes	3	3	6/23/2022	0:05:20	0:20:18
18	32	Joyanta	1	1	6/23/2022	0:03:25	0:04:58
19	31	Joyanta	1	1	6/23/2022	0:02:12	0:03:08
20	30	Joyanta	1	1	6/23/2022	0:00:44	0:01:15
User_Log2022-06-23							
Ready Accessibility: Unavailable							

Figure: 6.2: In and Out Datasheet in Excel

5.2 Advantages

There are many advantages of our project because of its accuracy. Some of the advantages are pointed out below:

- This project is easy to use.
- Fingerprint based home automation.
- Data Logger System.
- Cost effective
- No foul play takes place
- User-friendly
- Automatic and requires less human interaction
- Good way to reduce energy wastage
- Can be operated virtually from anywhere in the world

5.3 Applications

The application areas for this project in this modern and practical world are huge and some of these are given below:

- It can be used in any educational institutions
- Industry or corporate offices.
- Hospitals
- To identify authentic or fake person
- Residential areas

5.4 Limitations

No project is without its limitations and ours is no exception from that but the final output we received from our project is quite satisfactory. Listed below, are some of the limitations we have:

- It may take time to send data in data logger system.
- Must be connect with internet.
- No extra power sources.

5.5 Discussion

We built this system to secure our home or office. Using of fingerprint sensor we may preselect the person, only this person can enter to this door. Although we had some difficulty in building this system, due to the perseverance and valuable guidance of the supervisor sir, we had to complete it in a beautiful way.

CHAPTER SEVEN

CONCLUSION

6.1 Future Scope of Work

As we have already discussed about the limitations of our project so definitely there's room for improvement and thus, we have lots of future scope of work available to us for this project. Some of these are listed below:

- In future, we are thinking about adding record tracking feature to the project.
- In future, we are thinking about adding voice conversation options in the system.
- In future, we are thinking about adding a feature to our system which will notify the police if something goes wrong.
- Used cheap Chinese products for the prototype so there's some processing delay present in the circuit

6.2 Conclusion

We are now in the digital era and with the improvements of technology, our energy consumption rate has gone very high as well. We also don't feel that much safe as well due to having too many people around with ill intentions. That's we have built this project to tackle both of the issues. This project operates door condition with the help of finger print sensor. This kind of project is also available in outside countries but in ours we don't have much but as we can realize now, it has become really necessary for us. The project doesn't cost that much to build and it's very easy to use. It was built with the in-tension that everyone should be able to use it for their safety and home automation. Our project is in its pilot phase and now we doing practical tests with it to validate our work and taking notes for future improvements as well. At last, we can say that this project is necessary for our country and we hope we can succeed on our mission of making our homes and offices more security.

.

References

- [1] H. Hassan, R. Bakar, and A. Thaqib Fawwaz Mokhtar, “Face recognition based on auto-switching magnetic door lock system using microcontroller,” (2012)
- [2] K. P. Bhattarai, B. P. Gautam, and K. Sato, “Authentic Gate Entry System (AuthGES) by Using LBPH for Smart Home Security,” in 2018 International Conference on Networking and Network Applications (NaNA), (2018)
- [3] R. S. Divya and M. Mathew, “Survey on various door lock access control mechanisms,” in 2017 International Conference on Circuit ,Power and Computing Technologies (ICCPCT), (2017)
- [4] “Super secure door lock system for critical zones - IEEE Conference Publication.” [Online]. Available: <https://ieeexplore.ieee.org/document/8076773>. [Accessed: 15-Jun] (2019)
- [5] P. R. Nehete, J. P. Chaudhari, S. R. Pachpande, and K. P. Rane, “Literature Survey on Door Lock Security Systems,” *Int. J. Comput. Appl.*, vol. 153, no. 2, pp. 13–18, Nov. (2016)
- [6] U. Laila, M. Ahmad Khan, M. Kashif Shaikh, S. Annas bin Mazhar, and K. Mahboob, “Comparative analysis for a real time face recognition system using raspberry Pi,” (2017)
- [7] Y. Tae Park, P. Sthapit, and J.-Y. Pyun, “Smart digital door lock for the home automation,” (2009)
- [8] A. Anitha, “Home security system using internet of things,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 263, p. 042026, Nov. (2017) IX. M. Lal, K. Kumar, R. H. Arain, A. Maitlo, S. A. Ruk, and H. Shaikh, “Study of Face Recognition Techniques: A Survey,” *Int. J. Adv. Comput. Sci. Appl. IJACSA*, vol. 9, no. 6, (2018)

- [9] Meenakshi N, Monish M, Dikshit KJ, Bharath S. Arduino Based Smart Fingerprint Authentication System. In 2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT) 2019 Apr 25 (pp. 1-7). IEEE.
- [10] Baidya J, Saha T, Moyashir R, Palit R. Design and implementation of a fingerprint based lock system for shared access. In 2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC) 2017 Jan 9 (pp. 1-6). IEEE.
- [11] Anu, Bhatia D. A smart door access system using finger print biometric system. International Journal of Medical Engineering and Informatics 2. 2014 Jan 1;6(3):274-80.
- [12] Afolabi A, Alice O. On Securing a door with finger print biometric technique. Transactions on Machine Learning and Artificial Intelligence. 2014 Apr 11;2:86- 96.

Appendix

Micro Controller Code:

<pre> ----- -----*/ //*****lib aries***** * #include <Servo.h> #include <SPI.h> #include <Wire.h> #include <ESP8266WiFi.h> #include <SoftwareSerial.h> #include <ESP8266WebServer.h> #include <ESP8266HTTPClient.h> #include <Adafruit_GFX.h> //https://github.com/adafruit/Adafruit- GFX-Library #include <Adafruit_SSD1306.h> //https://github.com/adafruit/Adafruit_SS D1306 #include <Adafruit_Fingerprint.h> //https://github.com/adafruit/Adafruit- Fingerprint-Sensor-Library #include <WiFiClient.h> WiFiClient wifiClient; Servo servo; </pre>	<pre> //----- finger.getTemplateCount(); Serial.print("Sensor contains "); Serial.print(finger.templateCount); Serial.println(" templates"); Serial.println("Waiting for valid finger..."); //-----*test the connection*----- //SendFingerprintID(FingerID); } //***** ***** void loop() { //check if there's a connection to WiFi or not if(WiFi.status() != WL_CONNECTED){ connectToWiFi(); } //----- //If there no fingerprint has been scanned return -1 or -2 if there an error or 0 if there nothing, The ID start form 1 to 127 FingerID = getFingerprintID(); // Get the Fingerprint ID from the Scanner </pre>
---	--

<pre>//Fingerprint scanner Pins #define Finger_Rx 14 //D5 #define Finger_Tx 12 //D6 // Declaration for SSD1306 display connected using software I2C #define SCREEN_WIDTH 128 // OLED display width, in pixels #define SCREEN_HEIGHT 64 // OLED display height, in pixels #define OLED_RESET 0 // Reset pin # (or -1 if sharing Arduino reset pin) Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET); //***** ***** *** SoftwareSerial mySerial(Finger_Rx, Finger_Tx); Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial); //***** ***** *** /* Set these to your desired credentials. */ const char *ssid = "abcde"; //ENTER YOUR WIFI SETTINGS const char *password = "1234567890"; //***** ***** ***</pre>	<pre>delay(50); //don't need to run this at full speed. //----- DisplayFingerprintID(); //----- ChecktoAddID(); //----- ChecktoDeleteID(); //----- } //*****Display the fingerprint ID state on the OLED***** void DisplayFingerprintID(){ //Fingerprint has been detected if (FingerID > 0){ display.clearDisplay(); display.drawBitmap(34, 0, FinPr_valid_bits, FinPr_valid_width, FinPr_valid_height, WHITE); display.display(); SendFingerprintID(FingerID); // Send the Fingerprint ID to the website.</pre>
--	---

```
String postData ; // post array that will be
send to the website
String apiGetData =
"http://192.168.122.241
/biometricattendance/getdata.php";
//computer IP or the server domain
int FingerID = 0; // The Fingerprint ID
from the scanner
uint8_t id;
int data1=D7;

int temp=0,i=0;
char str[15];

//*****Biometric
Icons*****
**
#define Wifi_start_width 54
#define Wifi_start_height 49
const uint8_t PROGMEM Wifi_start_bits[]
= {
    0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x1f,0xf0,0x00,0x00,0x00
,0x00,0x03,0xff,0xff,0x80,0x00,0x00
,0x00,0x1f,0xf0,0x1f,0xf0,0x00,0x00
,0x00,0x7e,0x00,0x00,0xfc,0x00,0x00
,0x01,0xf0,0x00,0x00,0x1f,0x00,0x00
```

```
}
//-----
//No finger detected
else if (FingerID == 0){
    display.clearDisplay();
    display.drawBitmap( 32, 0, FinPr_start_bits,
FinPr_start_width, FinPr_start_height,
WHITE);
    display.display();
}
//-----
//Didn't find a match
else if (FingerID == -1){
    display.clearDisplay();
    display.drawBitmap( 34, 0,
FinPr_invalid_bits, FinPr_invalid_width,
FinPr_invalid_height, WHITE);
    display.display();
}
//-----
//Didn't find the scanner or there an error
else if (FingerID == -2){
    display.clearDisplay();
    display.drawBitmap( 32, 0,
FinPr_failed_bits, FinPr_failed_width,
FinPr_failed_height, WHITE);
    display.display();
}
}
//*****send the fingerprint ID to the
website*****
```

```
,0x03,0xc0,0x00,0x00,0x07,0xc0,0x00
,0x0f,0x00,0x00,0x00,0x01,0xe0,0x00
,0x1c,0x00,0x00,0x00,0x00,0x70,0x00
,0x38,0x00,0x07,0xc0,0x00,0x38,0x00
,0x70,0x00,0xff,0xfe,0x00,0x1e,0x00
,0xe0,0x03,0xfc,0x7f,0xc0,0x0e,0x00
,0x00,0x1f,0x80,0x03,0xf0,0x00,0x00
,0x00,0x3c,0x00,0x00,0x78,0x00,0x00
,0x00,0xf0,0x00,0x00,0x1c,0x00,0x00
,0x01,0xe0,0x00,0x00,0x0c,0x00,0x00
,0x03,0x80,0x00,0x00,0x00,0x00,0x00
,0x03,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x3f,0xf8,0x07,0x1e,0x00
,0x00,0x00,0xff,0xfe,0x1f,0xbf,0x80
,0x00,0x03,0xe0,0x04,0x7f,0xff,0xc0
,0x00,0x07,0x80,0x00,0xff,0xff,0xe0
,0x00,0x0e,0x00,0x00,0xff,0xff,0xe0
,0x00,0x0c,0x00,0x00,0x7f,0xff,0xc0
,0x00,0x00,0x00,0x00,0xfe,0x07,0xe0
,0x00,0x00,0x00,0x03,0xf8,0x03,0xf8
,0x00,0x00,0x07,0xe7,0xf9,0xf1,0xfc
,0x00,0x00,0x1f,0xe7,0xf1,0xf9,0xfc
,0x00,0x00,0x1f,0xe7,0xf3,0xf9,0xfc
,0x00,0x00,0x3f,0xe7,0xf3,0xf9,0xfc
,0x00,0x00,0x3f,0xe7,0xf1,0xf1,0xfc
,0x00,0x00,0x3f,0xe3,0xf8,0xe3,0xfc
,0x00,0x00,0x3f,0xf3,0xfc,0x07,0xf8
,0x00,0x00,0x1f,0xf0,0x7f,0x0f,0xc0
,0x00,0x00,0x0f,0xe0,0x7f,0xff,0xe0
,0x00,0x00,0x07,0xc0,0xff,0xff,0xe0
,0x00,0x00,0x00,0x00,0x7f,0xff,0xe0
```

```
void SendFingerprintID( int finger ){

    HTTPClient http; //Declare object of class
    HTTPClient
    //Post Data
    postData = "FingerID=" + String(finger); //
    Add the Fingerprint ID to the Post array in
    order to send it
    // Post methode

    http.begin(wifiClient, apiGetData);// get the
    result (**the error code**)
    http.addHeader("Content-Type",
    "application/x-www-form-urlencoded");
    //Specify content-type header
    digitalWrite(D7,1);
    delay(1000);
    digitalWrite(D7,0);
    int httpCode = http.POST(postData); //Send
    the request

    String payload = http.getString(); //Get the
    response payload

    Serial.println(httpCode); //Print HTTP return
    code
    Serial.println(payload); //Print request
    response payload
    Serial.println(postData); //Post Data
    Serial.println(finger); //Print fingerprint ID
    //////////////////////////////////////
    //////////////////////////////////////
```

<pre> ,0x00,0x00,0x00,0x00,0x3f,0xff,0x80 ,0x00,0x00,0x00,0x00,0x1f,0xbf,0x00 ,0x00,0x00,0x00,0x00,0x03,0x18,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00 }; #define Wifi_connected_width 63 #define Wifi_connected_height 49 const uint8_t PROGMEM Wifi_connected_bits[] = { 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x03,0xff,0xff,0x80,0x00,0x00 ,0x00,0x00,0x3f,0xff,0xff,0xf8,0x00,0x00 ,0x00,0x01,0xff,0xff,0xff,0xff,0x00,0x00 ,0x00,0x0f,0xff,0xff,0xff,0xff,0xe0,0x00 ,0x00,0x3f,0xff,0xc0,0x07,0xff,0xf8,0x00 ,0x00,0xff,0xf8,0x00,0x00,0x3f,0xfe,0x00 ,0x03,0xff,0x80,0x00,0x00,0x03,0xff,0x80 ,0x07,0xfe,0x00,0x00,0x00,0x00,0xff,0xc0 ,0x1f,0xf8,0x00,0x00,0x00,0x00,0x3f,0xf0 ,0x3f,0xe0,0x01,0xff,0xff,0x00,0x0f,0xf8 </pre>	<pre> /// ////////// if (FingerID == 1){ Serial.println("Low"); servo.write(0); delay(6000); servo.write(130); } if (FingerID == 2){ Serial.println("Low"); servo.write(0); delay(6000); servo.write(130); } if (FingerID == 3){ Serial.println("Low"); servo.write(0); delay(6000); servo.write(130); } if (payload.substring(0, 5) == "login") { String user_name = payload.substring(5); // Serial.println(user_name); </pre>
---	---

<pre> ,0x7f,0x80,0x0f,0xff,0xff,0xe0,0x03,0xfc ,0xff,0x00,0x7f,0xff,0xff,0xfc,0x01,0xfe ,0xfc,0x01,0xff,0xff,0xff,0xff,0x00,0x7e ,0x78,0x07,0xff,0xc0,0x07,0xff,0xc0,0x3c ,0x00,0x0f,0xfc,0x00,0x00,0x7f,0xe0,0x00 ,0x00,0x1f,0xf0,0x00,0x00,0x1f,0xf0,0x00 ,0x00,0x3f,0xc0,0x00,0x00,0x07,0xf8,0x00 ,0x00,0x7f,0x00,0x01,0x00,0x01,0xfc,0x00 ,0x00,0x7e,0x00,0x7f,0xfc,0x00,0xfc,0x00 ,0x00,0x3c,0x03,0xff,0xff,0x80,0x78,0x00 ,0x00,0x00,0x07,0xff,0xff,0xc0,0x00,0x00 ,0x00,0x00,0x1f,0xff,0xff,0xf0,0x00,0x00 ,0x00,0x00,0x3f,0xf0,0x1f,0xf8,0x00,0x00 ,0x00,0x00,0x3f,0x80,0x03,0xf8,0x00,0x00 ,0x00,0x00,0x3f,0x00,0x01,0xf8,0x00,0x00 ,0x00,0x00,0x1c,0x00,0x00,0x70,0x00,0x00 ,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x0f,0xe0,0x00,0x00,0x00 ,0x00,0x00,0x00,0x1f,0xf0,0x00,0x00,0x00 ,0x00,0x00,0x00,0x3f,0xf8,0x00,0x00,0x00 ,0x00,0x00,0x00,0x3f,0xf8,0x00,0x00,0x00 ,0x00,0x00,0x00,0x3f,0xf8,0x00,0x00,0x00 ,0x00,0x00,0x00,0x3f,0xf8,0x00,0x00,0x00 ,0x00,0x00,0x00,0x1f,0xf0,0x00,0x00,0x00 ,0x00,0x00,0x00,0x0f,0xe0,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 </pre>	<pre> display.clearDisplay(); display.setTextSize(2); // Normal 2:2 pixel scale display.setTextColor(WHITE); // Draw white text display.setCursor(15,0); // Start at top-left corner display.print(F("Welcome")); display.setCursor(0,20); display.print(user_name); display.display(); } else if (payload.substring(0, 6) == "logout") { String user_name = payload.substring(6); // Serial.println(user_name); display.clearDisplay(); display.setTextSize(2); // Normal 2:2 pixel scale display.setTextColor(WHITE); // Draw white text display.setCursor(10,0); // Start at top-left corner display.print(F("Good Bye")); display.setCursor(0,20); display.print(user_name); display.display(); } delay(1000); postData = ""; </pre>
---	---

```

,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
};
#define FinPr_start_width 64
#define FinPr_start_height 64
const uint8_t PROGMEM FinPr_start_bits[]
= {
    0x00,0x00,0x00,0x1f,0xe0,0x00,0x00,0x00
,0x00,0x00,0x01,0xff,0xfe,0x00,0x00,0x00
,0x00,0x00,0x03,0xff,0xff,0x80,0x00,0x00
,0x00,0x00,0x0f,0xc0,0x0f,0xe0,0x00,0x00
,0x00,0x00,0x1f,0x00,0x01,0xf8,0x00,0x00
,0x00,0x00,0x3c,0x00,0x00,0x7c,0x00,0x00
,0x00,0x00,0x78,0x00,0x00,0x3e,0x00,0x00
,0x00,0x00,0xf0,0x3f,0xf8,0x0f,0x00,0x00
,0x00,0x01,0xe0,0xff,0xfe,0x07,0x80,0x00
,0x00,0x03,0xc3,0xff,0xff,0x03,0x80,0x00
,0x00,0x03,0x87,0xc0,0x07,0xc3,0xc0,0x00
,0x00,0x07,0x0f,0x00,0x03,0xe1,0xc0,0x00
,0x00,0x0f,0x0e,0x00,0x00,0xe0,0xe0,0x00
,0x00,0x0e,0x1c,0x00,0x00,0xf0,0xe0,0x00
,0x00,0x0c,0x3c,0x1f,0xe0,0x70,0xe0,0x00
,0x00,0x00,0x38,0x3f,0xf0,0x38,0x70,0x00
,0x00,0x00,0x78,0x78,0xf8,0x38,0x70,0x00
,0x00,0x00,0x70,0x70,0x3c,0x18,0x70,0x00
,0x00,0x00,0xe0,0xe0,0x1e,0x1c,0x70,0x00
,0x00,0x03,0xe1,0xe0,0x0e,0x1c,0x70,0x00
,0x00,0x0f,0xc1,0xc3,0x0e,0x1c,0x70,0x00
,0x00,0x3f,0x03,0xc3,0x8e,0x1c,0x70,0x00
,0x00,0x3e,0x03,0x87,0x0e,0x1c,0x70,0x00
,0x00,0x30,0x07,0x07,0x0e,0x18,0xe0,0x00
,0x00,0x00,0x0e,0x0e,0x0e,0x38,0xe0,0x00

```

```

http.end(); //Close connection
}
//*****Get the Fingerprint
ID*****
int getFingerprintID() {
    uint8_t p = finger.getImage();
    switch (p) {
        case FINGERPRINT_OK:
            //Serial.println("Image taken");
            break;
        case FINGERPRINT_NOFINGER:
            //Serial.println("No finger detected");
            return 0;
        case FINGERPRINT_PACKETRECEIVEERR:
            //Serial.println("Communication error");
            return -2;
        case FINGERPRINT_IMAGEFAIL:
            //Serial.println("Imaging error");
            return -2;
        default:
            //Serial.println("Unknown error");
            return -2;
    }
    // OK success!
    p = finger.image2Tz();
    switch (p) {
        case FINGERPRINT_OK:
            //Serial.println("Image converted");
            break;
        case FINGERPRINT_IMAGEMESS:
            //Serial.println("Image too messy");

```

```
,0x00,0x00,0x3e,0x1e,0x1e,0x38,0xe0,0x00
,0x00,0x00,0xf8,0x1c,0x1c,0x38,0xe0,0x00
,0x00,0x03,0xf0,0x38,0x3c,0x38,0xe0,0x00
,0x00,0x3f,0xc0,0xf8,0x78,0x38,0xe0,0x00
,0x00,0x7f,0x01,0xf0,0x70,0x38,0xf0,0x00
,0x00,0x78,0x03,0xe0,0xe0,0x38,0x70,0x00
,0x00,0x00,0x0f,0x81,0xe0,0x38,0x7c,0x00
,0x00,0x00,0x3f,0x03,0xc0,0x38,0x3e,0x00
,0x00,0x00,0xfc,0x0f,0x80,0x38,0x1e,0x00
,0x00,0x07,0xf0,0x1f,0x1c,0x1c,0x04,0x00
,0x00,0x3f,0xc0,0x3e,0x3f,0x1e,0x00,0x00
,0x00,0x7f,0x00,0xf8,0x7f,0x0f,0x00,0x00
,0x00,0x38,0x01,0xf0,0xf7,0x07,0xc0,0x00
,0x00,0x00,0x07,0xe1,0xe3,0x83,0xf8,0x00
,0x00,0x00,0x3f,0x87,0xc3,0xc0,0xfc,0x00
,0x00,0x01,0xfe,0x0f,0x81,0xe0,0x3c,0x00
,0x00,0x0f,0xf8,0x1f,0x00,0xf0,0x00,0x00
,0x00,0x1f,0xc0,0x7c,0x00,0x7c,0x00,0x00
,0x00,0x1e,0x01,0xf8,0x00,0x3f,0x00,0x00
,0x00,0x00,0x07,0xe0,0x78,0x0f,0xc0,0x00
,0x00,0x00,0x3f,0x81,0xfe,0x07,0xf0,0x00
,0x00,0x01,0xfe,0x07,0xff,0x01,0xf0,0x00
,0x00,0x07,0xf8,0x0f,0x87,0x80,0x30,0x00
,0x00,0x07,0xc0,0x3f,0x03,0xe0,0x00,0x00
,0x00,0x06,0x00,0xfc,0x01,0xf8,0x00,0x00
,0x00,0x00,0x03,0xf0,0x00,0x7e,0x00,0x00
,0x00,0x00,0x0f,0xc0,0x00,0x3f,0x80,0x00
,0x00,0x00,0x7f,0x00,0xf8,0x0f,0x80,0x00
,0x00,0x00,0xfc,0x03,0xfe,0x01,0x80,0x00
,0x00,0x00,0xf0,0x1f,0xff,0x80,0x00,0x00
,0x00,0x00,0x00,0x7f,0x07,0xe0,0x00,0x00
```

```
return -1;
case FINGERPRINT_PACKETRECIEVEERR:
    //Serial.println("Communication error");
    return -2;
case FINGERPRINT_FEATUREFAIL:
    //Serial.println("Could not find fingerprint
features");
    return -2;
case FINGERPRINT_INVALIDIMAGE:
    //Serial.println("Could not find fingerprint
features");
    return -2;
default:
    //Serial.println("Unknown error");
    return -2;
}
// OK converted!
p = finger.fingerFastSearch();
if (p == FINGERPRINT_OK) {
    //Serial.println("Found a print match!");
} else if (p ==
FINGERPRINT_PACKETRECIEVEERR) {
    //Serial.println("Communication error");
    return -2;
} else if (p == FINGERPRINT_NOTFOUND) {
    //Serial.println("Did not find a match");
    return -1;
} else {
    //Serial.println("Unknown error");
    return -2;
}
```

```

,0x00,0x00,0x00,0xfc,0x03,0xf8,0x00,0x00
,0x00,0x00,0x03,0xf0,0x00,0x78,0x00,0x00
,0x00,0x00,0x0f,0xc0,0x00,0x18,0x00,0x00
,0x00,0x00,0x0f,0x01,0xf8,0x00,0x00,0x00
,0x00,0x00,0x00,0x07,0xfe,0x00,0x00,0x00
,0x00,0x00,0x00,0x1f,0xfe,0x00,0x00,0x00
,0x00,0x00,0x00,0x1e,0x0e,0x00,0x00,0x00
,0x00,0x00,0x00,0x18,0x00,0x00,0x00,0x00
};
//-----
-----

#define FinPr_valid_width 64
#define FinPr_valid_height 64
const uint8_t PROGMEM FinPr_valid_bits[]
= {
    0x00,0x00,0x03,0xfe,0x00,0x00,0x00,0x00
,0x00,0x00,0x1f,0xff,0xe0,0x00,0x00,0x00
,0x00,0x00,0x7f,0xff,0xf8,0x00,0x00,0x00
,0x00,0x00,0xfc,0x00,0xfe,0x00,0x00,0x00
,0x00,0x03,0xe0,0x00,0x1f,0x00,0x00,0x00
,0x00,0x07,0xc0,0x00,0x07,0x80,0x00,0x00
,0x00,0x0f,0x80,0x00,0x03,0xe0,0x00,0x00
,0x00,0x0e,0x03,0xff,0x01,0xe0,0x00,0x00
,0x00,0x1c,0x1f,0xff,0xe0,0xf0,0x00,0x00
,0x00,0x3c,0x3f,0xff,0xf0,0x78,0x00,0x00
,0x00,0x78,0x7c,0x00,0xf8,0x3c,0x00,0x00
,0x00,0x70,0xf0,0x00,0x3c,0x1c,0x00,0x00
,0x00,0xe1,0xe0,0x00,0x1e,0x1c,0x00,0x00
,0x00,0xe1,0xc0,0x00,0x0f,0x0e,0x00,0x00
,0x00,0xc3,0x81,0xfc,0x07,0x0e,0x00,0x00
,0x00,0x03,0x83,0xff,0x07,0x8e,0x00,0x00

```

```

// found a match!
//Serial.print("Found ID #");
Serial.print(finger.fingerID);
//Serial.print(" with confidence of ");
Serial.println(finger.confidence);

return finger.fingerID;
}
//*****Check if there a
Fingerprint ID to delete*****
void CheckToDeleteID(){

    HTTPClient http; //Declare object of class
    HTTPClient

    //Post Data
    postData = "DeleteID=check"; // Add the
    Fingerprint ID to the Post array in order to
    send it
    // Post methode

    http.begin(wifiClient, apiGetData);// get the
    result (**the error code**)
    http.addHeader("Content-Type",
    "application/x-www-form-urlencoded");
    //Specify content-type header

    int httpCode = http.POST(postData); //Send
    the request

    String payload = http.getString(); //Get the
    response payload

```

```
,0x00,0x07,0x07,0x8f,0x83,0x87,0x00,0x00
,0x00,0x0f,0x0f,0x03,0xc3,0x87,0x00,0x00
,0x00,0x1e,0x0e,0x01,0xc3,0x87,0x00,0x00
,0x00,0x3c,0x1c,0x00,0xe1,0x87,0x00,0x00
,0x00,0xf8,0x1c,0x30,0xe1,0x87,0x00,0x00
,0x07,0xf0,0x38,0x70,0xe1,0x86,0x00,0x00
,0x07,0xc0,0x78,0x70,0xe3,0x8e,0x00,0x00
,0x02,0x00,0xf0,0xf0,0xe3,0x8e,0x00,0x00
,0x00,0x01,0xe0,0xe0,0xe3,0x8e,0x00,0x00
,0x00,0x03,0xc1,0xe1,0xc3,0x8e,0x00,0x00
,0x00,0x0f,0x83,0xc3,0xc3,0x8e,0x00,0x00
,0x00,0x7f,0x07,0x83,0x83,0x0e,0x00,0x00
,0x07,0xfc,0x0f,0x07,0x83,0x0e,0x00,0x00
,0x07,0xf0,0x1e,0x0f,0x03,0x0e,0x00,0x00
,0x07,0x80,0x7c,0x1e,0x03,0x07,0x00,0x00
,0x00,0x00,0xf8,0x3c,0x03,0x87,0x80,0x00
,0x00,0x03,0xf0,0x78,0x03,0x83,0xc0,0x00
,0x00,0x1f,0xc0,0xf0,0x02,0x00,0x00,0x00
,0x00,0xff,0x01,0xe1,0xc0,0x0c,0x00,0x00
,0x07,0xfc,0x03,0xc3,0xe1,0xff,0xc0,0x00
,0x07,0xe0,0x0f,0x87,0xc7,0xff,0xf0,0x00
,0x07,0x00,0x3f,0x0f,0x0f,0xff,0xfc,0x00
,0x00,0x00,0x7c,0x3e,0x3f,0xff,0xfe,0x00
,0x00,0x03,0xf8,0x7c,0x3f,0xff,0xff,0x00
,0x00,0x1f,0xe0,0xf0,0x7f,0xff,0xff,0x80
,0x00,0xff,0x83,0xe0,0xff,0xff,0xff,0x80
,0x01,0xfc,0x07,0xc1,0xff,0xff,0xe3,0xc0
,0x01,0xe0,0x1f,0x01,0xff,0xff,0xc3,0xc0
,0x00,0x00,0xfe,0x01,0xff,0xff,0x87,0xe0
,0x00,0x03,0xf8,0x13,0xff,0xff,0x0f,0xe0
,0x00,0x1f,0xe0,0x73,0xff,0xfe,0x1f,0xe0
```

```
if (payload.substring(0, 6) == "del-id") {
    String del_id = payload.substring(6);
    Serial.println(del_id);
    deleteFingerprint( del_id.toInt() );
}

http.end(); //Close connection
}

//*****Delete Fingerprint
ID*****

uint8_t deleteFingerprint( int id) {
    uint8_t p = -1;

    p = finger.deleteModel(id);

    if (p == FINGERPRINT_OK) {
        //Serial.println("Deleted!");
        display.clearDisplay();
        display.setTextSize(2);        // Normal 2:2
        pixel scale
        display.setTextColor(WHITE);    // Draw
        white text
        display.setCursor(0,0);        // Start at top-
        left corner
        display.print(F("Deleted!\n"));
        display.display();
    } else if (p ==
FINGERPRINT_PACKETRECEIVEERR) {
        //Serial.println("Communication error");
        display.clearDisplay();
    }
}
```

<pre> ,0x00,0x7f,0x81,0xf3,0xff,0xfc,0x1f,0xe0 ,0x00,0xfc,0x03,0xe3,0xef,0xf8,0x3f,0xe0 ,0x00,0x60,0x0f,0xc3,0xc7,0xf0,0x7f,0xe0 ,0x00,0x00,0x3f,0x03,0xc3,0xe0,0xff,0xe0 ,0x00,0x00,0xfc,0x03,0xc1,0xc1,0xff,0xe0 ,0x00,0x07,0xf0,0x13,0xe0,0x83,0xff,0xe0 ,0x00,0x0f,0xc0,0x7b,0xf8,0x07,0xff,0xe0 ,0x00,0x0f,0x01,0xf9,0xfc,0x0f,0xff,0xc0 ,0x00,0x00,0x07,0xf1,0xfe,0x1f,0xff,0xc0 ,0x00,0x00,0x1f,0xc0,0xff,0x3f,0xff,0x80 ,0x00,0x00,0x7e,0x00,0xff,0xff,0xff,0x80 ,0x00,0x00,0xfc,0x00,0x7f,0xff,0xff,0x00 ,0x00,0x00,0xf0,0x1f,0x3f,0xff,0xfe,0x00 ,0x00,0x00,0x00,0x7f,0x1f,0xff,0xfc,0x00 ,0x00,0x00,0x01,0xff,0x8f,0xff,0xf8,0x00 ,0x00,0x00,0x03,0xe0,0xe3,0xff,0xe0,0x00 ,0x00,0x00,0x01,0x80,0x00,0x7f,0x00,0x00 }; //----- ----- #define FinPr_invalid_width 64 #define FinPr_invalid_height 64 const uint8_t PROGMEM FinPr_invalid_bits[] = { 0x00,0x00,0x03,0xfe,0x00,0x00,0x00,0x00 ,0x00,0x00,0x1f,0xff,0xe0,0x00,0x00,0x00 ,0x00,0x00,0x7f,0xff,0xf8,0x00,0x00,0x00 ,0x00,0x00,0xfc,0x00,0xfe,0x00,0x00,0x00 ,0x00,0x03,0xe0,0x00,0x1f,0x00,0x00,0x00 ,0x00,0x07,0xc0,0x00,0x07,0x80,0x00,0x00 ,0x00,0x0f,0x80,0x00,0x03,0xe0,0x00,0x00 </pre>	<pre> display.setTextSize(1); // Normal 1:1 pixel scale display.setTextColor(WHITE); // Draw white text display.setCursor(0,0); // Start at top- left corner display.print(F("Communication error!\n")); display.display(); return p; } else if (p == FINGERPRINT_BADLOCATION) { //Serial.println("Could not delete in that location"); display.clearDisplay(); display.setTextSize(1); // Normal 1:1 pixel scale display.setTextColor(WHITE); // Draw white text display.setCursor(0,0); // Start at top- left corner display.print(F("Could not delete in that location!\n")); display.display(); return p; } else if (p == FINGERPRINT_FLASHERR) { //Serial.println("Error writing to flash"); display.clearDisplay(); display.setTextSize(1); // Normal 1:1 pixel scale display.setTextColor(WHITE); // Draw white text </pre>
--	--

,0x00,0x0e,0x03,0xff,0x01,0xe0,0x00,0x00 ,0x00,0x1c,0x1f,0xff,0xe0,0xf0,0x00,0x00 ,0x00,0x3c,0x3f,0xff,0xf0,0x78,0x00,0x00 ,0x00,0x78,0x7c,0x00,0xf8,0x3c,0x00,0x00 ,0x00,0x70,0xf0,0x00,0x3c,0x1c,0x00,0x00 ,0x00,0xe1,0xe0,0x00,0x1e,0x1c,0x00,0x00 ,0x00,0xe1,0xc0,0x00,0x0f,0x0e,0x00,0x00 ,0x00,0xc3,0x81,0xfc,0x07,0x0e,0x00,0x00 ,0x00,0x03,0x83,0xff,0x07,0x8e,0x00,0x00 ,0x00,0x07,0x07,0x8f,0x83,0x87,0x00,0x00 ,0x00,0x0f,0x0f,0x03,0xc3,0x87,0x00,0x00 ,0x00,0x1e,0x0e,0x01,0xc3,0x87,0x00,0x00 ,0x00,0x3c,0x1c,0x00,0xe1,0x87,0x00,0x00 ,0x00,0xf8,0x1c,0x30,0xe1,0x87,0x00,0x00 ,0x07,0xf0,0x38,0x70,0xe1,0x86,0x00,0x00 ,0x07,0xc0,0x78,0x70,0xe3,0x8e,0x00,0x00 ,0x02,0x00,0xf0,0xf0,0xe3,0x8e,0x00,0x00 ,0x00,0x01,0xe0,0xe0,0xe3,0x8e,0x00,0x00 ,0x00,0x03,0xc1,0xe1,0xc3,0x8e,0x00,0x00 ,0x00,0x0f,0x83,0xc3,0xc3,0x8e,0x00,0x00 ,0x00,0x7f,0x07,0x83,0x83,0x0e,0x00,0x00 ,0x07,0xfc,0x0f,0x07,0x83,0x0e,0x00,0x00 ,0x07,0xf0,0x1e,0x0f,0x03,0x0e,0x00,0x00 ,0x07,0x80,0x7c,0x1e,0x03,0x07,0x00,0x00 ,0x00,0x00,0xf8,0x3c,0x03,0x87,0x80,0x00 ,0x00,0x03,0xf0,0x78,0x03,0x83,0xc0,0x00 ,0x00,0x1f,0xc0,0xf0,0x02,0x00,0x00,0x00 ,0x00,0xff,0x01,0xe1,0xc0,0x00,0x00,0x00 ,0x07,0xfc,0x03,0xc3,0xe1,0xff,0xc0,0x00 ,0x07,0xe0,0x0f,0x87,0xc7,0xff,0xf0,0x00 ,0x07,0x00,0x3f,0x0f,0x0f,0xff,0xf8,0x00	display.setCursor(0,0); // Start at top- left corner display.print(F("Error writing to flash!\n")); display.display(); return p; } else { //Serial.print("Unknown error: 0x"); Serial.println(p, HEX); display.clearDisplay(); display.setTextSize(2); // Normal 2:2 pixel scale display.setTextColor(WHITE); // Draw white text display.setCursor(0,0); // Start at top- left corner display.print(F("Unknown error:\n")); display.display(); return p; } } //*****Check if there a Fingerprint ID to add***** void ChecktoAddID(){ HTTPClient http; //Declare object of class HTTPClient //Post Data postData = "Get_Fingerid=get_id"; // Add the Fingerprint ID to the Post array in order to send it // Post methode
--	---

```
,0x00,0x00,0x7c,0x3e,0x1f,0xff,0xfe,0x00
,0x00,0x03,0xf8,0x7c,0x3f,0xff,0xff,0x00
,0x00,0x1f,0xe0,0xf0,0x7f,0xff,0xff,0x00
,0x00,0xff,0x83,0xe0,0xfe,0xff,0xbf,0x80
,0x01,0xfc,0x07,0xc0,0xfc,0x7f,0x1f,0xc0
,0x01,0xe0,0x1f,0x01,0xf8,0x3e,0x0f,0xc0
,0x00,0x00,0xfe,0x01,0xf8,0x1c,0x07,0xe0
,0x00,0x03,0xf8,0x13,0xf8,0x00,0x0f,0xe0
,0x00,0x1f,0xe0,0x73,0xfc,0x00,0x1f,0xe0
,0x00,0x7f,0x81,0xf3,0xfe,0x00,0x3f,0xe0
,0x00,0xfc,0x03,0xe3,0xff,0x00,0x7f,0xe0
,0x00,0x60,0x0f,0xc3,0xff,0x80,0xff,0xe0
,0x00,0x00,0x3f,0x03,0xff,0x00,0x7f,0xe0
,0x00,0x00,0xfc,0x03,0xfe,0x00,0x3f,0xe0
,0x00,0x07,0xf0,0x13,0xfc,0x00,0x1f,0xe0
,0x00,0x0f,0xc0,0x79,0xf8,0x08,0x0f,0xe0
,0x00,0x0f,0x01,0xf9,0xf8,0x1c,0x0f,0xc0
,0x00,0x00,0x07,0xf1,0xfc,0x3e,0x1f,0xc0
,0x00,0x00,0x1f,0xc0,0xfe,0x7f,0x3f,0x80
,0x00,0x00,0x7e,0x00,0xff,0xff,0xff,0x80
,0x00,0x00,0xfc,0x00,0x7f,0xff,0xff,0x00
,0x00,0x00,0xf0,0x1f,0x3f,0xff,0xfe,0x00
,0x00,0x00,0x00,0x7f,0x1f,0xff,0xfc,0x00
,0x00,0x00,0x01,0xff,0x8f,0xff,0xf8,0x00
,0x00,0x00,0x03,0xe0,0xe3,0xff,0xe0,0x00
,0x00,0x00,0x01,0x80,0x00,0x7f,0x00,0x00
};
//-----
-----
#define FinPr_failed_width 64
#define FinPr_failed_height 64
```

```
http.begin(wifiClient, apiGetData); // get the
result (**the error code**)
http.addHeader("Content-Type",
"application/x-www-form-urlencoded");
//Specify content-type header

int httpCode = http.POST(postData); //Send
the request

String payload = http.getString(); //Get the
response payload

if (payload.substring(0, 6) == "add-id") {
String add_id = payload.substring(6);
Serial.println(add_id);
id = add_id.toInt();
getFingerprintEnroll();
}
http.end(); //Close connection
}
//*****Enroll a Fingerprint
ID*****

uint8_t getFingerprintEnroll() {

int p = -1;

display.clearDisplay();

display.drawBitmap( 34, 0, FinPr_scan_bits,
FinPr_scan_width, FinPr_scan_height, WHITE);

display.display();

while (p != FINGERPRINT_OK) {
p = finger.getImage();
```

<pre> const uint8_t PROGMEM FinPr_failed_bits[] = { 0x00,0x00,0x3f,0xe0,0x00,0x00,0x00,0x00 ,0x00,0x01,0xff,0xfe,0x00,0x00,0x00,0x00 ,0x00,0x0f,0xc0,0x1f,0x80,0x00,0x00,0x00 ,0x00,0x1e,0x00,0x03,0xc0,0x00,0x00,0x00 ,0x00,0x78,0x00,0x00,0xf0,0x00,0x00,0x00 ,0x00,0xe0,0x00,0x00,0x38,0x00,0x00,0x00 ,0x01,0xc0,0x00,0x00,0x1c,0x00,0x00,0x00 ,0x03,0x80,0x00,0x00,0x0e,0x00,0x00,0x00 ,0x07,0x00,0x7f,0xe0,0x07,0x00,0x00,0x00 ,0x06,0x01,0xff,0xf8,0x03,0x00,0x00,0x00 ,0x0c,0x03,0xc0,0x3c,0x03,0x80,0x00,0x00 ,0x1c,0x0f,0x00,0x0e,0x01,0x80,0x00,0x00 ,0x18,0x0c,0x00,0x03,0x00,0xc0,0x00,0x00 ,0x18,0x18,0x00,0x01,0x80,0xc0,0x00,0x00 ,0x30,0x38,0x00,0x01,0xc0,0xe0,0x00,0x00 ,0x30,0x30,0x0f,0x00,0xc0,0x60,0x00,0x00 ,0x30,0x30,0x3f,0xc0,0xe0,0x60,0x00,0x00 ,0x70,0x60,0x78,0xe0,0x60,0x60,0x00,0x00 ,0x60,0x60,0x60,0x60,0x60,0x70,0x00,0x00 ,0x60,0x60,0x60,0x60,0x60,0x30,0x00,0x00 ,0x60,0x60,0x60,0x60,0x30,0x30,0x00,0x00 ,0x60,0x60,0x60,0x30,0x30,0x20,0x00,0x00 ,0x60,0x60,0x60,0x30,0x30,0x01,0xe0,0x00 ,0x60,0x60,0x60,0x30,0x30,0x0f,0xfc,0x00 ,0x60,0x60,0x60,0x30,0x30,0x3f,0xff,0x00 ,0x60,0x60,0x60,0x30,0x18,0x78,0x03,0x80 ,0x60,0x60,0x60,0x30,0x1c,0x60,0x01,0x80 ,0x60,0x60,0x30,0x38,0x0c,0xc0,0x00,0xc0 ,0x00,0x60,0x30,0x18,0x00,0xc0,0x00,0xc0 </pre>	<pre> switch (p) { case FINGERPRINT_OK: //Serial.println("Image taken"); display.clearDisplay(); display.drawBitmap(34, 0, FinPr_valid_bits, FinPr_valid_width, FinPr_valid_height, WHITE); display.display(); break; case FINGERPRINT_NOFINGER: //Serial.println("."); display.setTextSize(1); // Normal 2:2 pixel scale display.setTextColor(WHITE); // Draw white text display.setCursor(0,0); // Start at top-left corner display.print(F("scanning")); display.display(); break; case FINGERPRINT_PACKETRECEIVEERR: display.clearDisplay(); display.drawBitmap(34, 0, FinPr_invalid_bits, FinPr_invalid_width, FinPr_invalid_height, WHITE); display.display(); break; case FINGERPRINT_IMAGEFAIL: Serial.println("Imaging error"); break; default: </pre>
--	--

,0x00,0x60,0x30,0x18,0x00,0xc0,0x00,0xc0 ,0x00,0xe0,0x30,0x0c,0x01,0xc0,0x00,0xe0 ,0x00,0xc0,0x18,0x0e,0x01,0xc0,0x00,0xe0 ,0x60,0xc0,0x18,0x07,0x01,0xc0,0x00,0xe0 ,0x01,0xc0,0x1c,0x03,0x81,0xc0,0x00,0xe0 ,0x01,0x80,0x0c,0x01,0xc1,0xc0,0x00,0xe0 ,0x03,0x80,0x0e,0x00,0xf1,0xc0,0x00,0xe0 ,0x0f,0x00,0x06,0x00,0x01,0xc0,0x00,0xe0 ,0x3e,0x01,0x03,0x00,0x01,0xc0,0x00,0xe0 ,0x30,0x03,0x83,0x80,0x1f,0xff,0xff,0xfe ,0x00,0x03,0x81,0xc0,0x3f,0xff,0xff,0xff ,0x00,0x07,0xc0,0xe0,0x30,0x00,0x00,0x03 ,0x00,0x0e,0xc0,0x78,0x30,0x00,0x00,0x03 ,0x00,0x3c,0x60,0x1e,0x30,0x00,0x00,0x03 ,0x00,0x78,0x70,0x0f,0x30,0x00,0x00,0x03 ,0x03,0xe0,0x38,0x03,0x30,0x00,0x00,0x03 ,0x07,0x80,0x1c,0x00,0x30,0x00,0x00,0x03 ,0xc0,0x00,0x0f,0x00,0x30,0x00,0x00,0x03 ,0xc0,0x00,0x03,0x80,0x30,0x01,0xe0,0x03 ,0x00,0x18,0x01,0xe0,0x30,0x03,0xf0,0x03 ,0x00,0x18,0x00,0x7c,0x30,0x07,0x38,0x03 ,0x00,0x0c,0x00,0x1f,0x30,0x06,0x18,0x03 ,0x18,0x0e,0x00,0x07,0x30,0x06,0x18,0x03 ,0x0c,0x07,0x80,0x00,0x30,0x07,0x38,0x03 ,0x0e,0x03,0xc0,0x00,0x30,0x03,0x30,0x03 ,0x07,0x00,0xf0,0x00,0x30,0x03,0x30,0x03 ,0x03,0x00,0x7e,0x00,0x30,0x03,0x30,0x03 ,0x01,0x80,0x1f,0xc0,0x30,0x03,0x30,0x03 ,0x01,0xc0,0x03,0xe1,0x30,0x07,0xf8,0x03 ,0x00,0xf0,0x00,0x01,0x30,0x03,0xf0,0x03 ,0x00,0x38,0x00,0x00,0x30,0x00,0x00,0x03	Serial.println("Unknown error"); break; } } // OK success! p = finger.image2Tz(1); switch (p) { case FINGERPRINT_OK: display.clearDisplay(); display.drawBitmap(34, 0, FinPr_valid_bits, FinPr_valid_width, FinPr_valid_height, WHITE); display.display(); break; case FINGERPRINT_IMAGEMESS: display.clearDisplay(); display.drawBitmap(34, 0, FinPr_invalid_bits, FinPr_invalid_width, FinPr_invalid_height, WHITE); display.display(); return p; case FINGERPRINT_PACKETRECEIVEERR: Serial.println("Communication error"); return p; case FINGERPRINT_FEATUREFAIL: Serial.println("Could not find fingerprint features"); return p; case FINGERPRINT_INVALIDIMAGE:
--	---

```
,0x00,0x1e,0x00,0x00,0x30,0x00,0x00,0x03
,0x00,0x07,0xc0,0x00,0x30,0x00,0x00,0x03
,0x00,0x01,0xff,0x80,0x3f,0xff,0xff,0xff
,0x00,0x00,0x3f,0x80,0x1f,0xff,0xff,0xfe
};
//-----
-----

#define FinPr_scan_width 64
#define FinPr_scan_height 64
const uint8_t PROGMEM FinPr_scan_bits[]
= {

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x1f,0xf8,0x00,0x00,0x00
,0x00,0x00,0x00,0x7f,0xff,0x00,0x00,0x00
,0x00,0x00,0x01,0xfc,0x7f,0xc0,0x00,0x00
,0x00,0x00,0x03,0xc0,0x03,0xe0,0x00,0x00
,0x00,0x00,0x07,0x80,0x00,0xf0,0x00,0x00
,0x00,0x00,0x0e,0x00,0x00,0x3c,0x00,0x00
,0x00,0x00,0x1c,0x1f,0xfc,0x1c,0x00,0x00
,0x00,0x00,0x38,0x7f,0xfe,0x0e,0x00,0x00
,0x00,0x00,0x78,0xf8,0x0f,0x87,0x00,0x00
,0x00,0x00,0x71,0xe0,0x03,0xc7,0x00,0x00
,0x00,0x00,0xe3,0x80,0x01,0xc3,0x80,0x00
,0x00,0x00,0xc3,0x83,0xc0,0xe3,0x80,0x00
,0x00,0x00,0xc7,0x0f,0xf0,0x71,0x80,0x00
,0x00,0x00,0x06,0x1f,0xf8,0x71,0xc0,0x00
```

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

return p;

default:

Serial.println("Unknown error");

return p;

}

display.clearDisplay();

display.setTextSize(2);      // Normal 2:2
pixel scale

display.setTextColor(WHITE); // Draw
white text

display.setCursor(0,0);      // Start at top-
left corner

display.print(F("Remove"));

display.setCursor(0,20);

display.print(F("finger"));

display.display();

//Serial.println("Remove finger");

delay(2000);

p = 0;

while (p != FINGERPRINT_NOFINGER) {

p = finger.getImage();

}

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

p = -1;

display.clearDisplay();

display.drawBitmap( 34, 0, FinPr_scan_bits,
FinPr_scan_width, FinPr_scan_height, WHITE);

display.display();

while (p != FINGERPRINT_OK) {
```

<pre> ,0x00,0x00,0x0e,0x1c,0x3c,0x31,0xc0,0x00 ,0x00,0x00,0x1c,0x38,0x1c,0x31,0xc0,0x00 ,0x00,0x00,0x38,0x70,0x0e,0x39,0xc0,0x00 ,0x00,0x01,0xf0,0x71,0x8e,0x39,0xc0,0x00 ,0x00,0x03,0xe0,0xe1,0x86,0x31,0xc0,0x00 ,0x00,0x03,0x81,0xe3,0x8e,0x31,0x80,0x00 ,0x00,0x00,0x03,0xc3,0x8e,0x33,0x80,0x00 ,0x00,0x00,0x07,0x87,0x0c,0x73,0x80,0x00 ,0x00,0x00,0x1f,0x0e,0x1c,0x73,0x80,0x00 ,0x7f,0xff,0xff,0xff,0xff,0xff,0xff,0xfe ,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff ,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff ,0x7f,0xff,0xff,0xff,0xff,0xff,0xff,0xfe ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 ,0x00,0x03,0xf0,0x1e,0x3e,0x1c,0x00,0x00 ,0x00,0x03,0x80,0x7c,0x77,0x0f,0x00,0x00 ,0x00,0x00,0x01,0xf0,0xe3,0x07,0xc0,0x00 ,0x00,0x00,0x07,0xe3,0xc3,0x81,0xf0,0x00 ,0x00,0x00,0x3f,0x87,0x81,0xc0,0x60,0x00 ,0x00,0x01,0xfc,0x1f,0x00,0xf0,0x00,0x00 ,0x00,0x01,0xe0,0x3c,0x00,0x7c,0x00,0x00 ,0x00,0x00,0x00,0xf8,0x78,0x1f,0x00,0x00 ,0x00,0x00,0x07,0xe0,0xfc,0x0f,0xc0,0x00 ,0x00,0x00,0x3f,0x83,0xef,0x03,0xc0,0x00 ,0x00,0x00,0xfc,0x0f,0x87,0x80,0x00,0x00 ,0x00,0x00,0x70,0x1f,0x03,0xe0,0x00,0x00 ,0x00,0x00,0x00,0x7c,0x00,0xf8,0x00,0x00 ,0x00,0x00,0x01,0xf0,0x00,0x3e,0x00,0x00 ,0x00,0x00,0x0f,0xc0,0xf8,0x0f,0x00,0x00 </pre>	<pre> p = finger.getImage(); switch (p) { case FINGERPRINT_OK: //Serial.println("Image taken"); display.clearDisplay(); display.drawBitmap(34, 0, FinPr_valid_bits, FinPr_valid_width, FinPr_valid_height, WHITE); display.display(); break; case FINGERPRINT_NOFINGER: //Serial.println("."); display.setTextSize(1); // Normal 2:2 pixel scale display.setTextColor(WHITE); // Draw white text display.setCursor(0,0); // Start at top-left corner display.print(F("scanning")); display.display(); break; case FINGERPRINT_PACKETRECEIVEERR: Serial.println("Communication error"); break; case FINGERPRINT_IMAGEFAIL: Serial.println("Imaging error"); break; default: Serial.println("Unknown error"); break; } </pre>
---	---

```
,0x00,0x00,0x1f,0x03,0xfe,0x02,0x00,0x00
,0x00,0x00,0x0c,0x0f,0x8f,0x80,0x00,0x00
,0x00,0x00,0x00,0x3f,0x03,0xe0,0x00,0x00
,0x00,0x00,0x00,0xf8,0x00,0xf0,0x00,0x00
,0x00,0x00,0x01,0xe0,0x00,0x30,0x00,0x00
,0x00,0x00,0x01,0xc0,0xf8,0x00,0x00,0x00
,0x00,0x00,0x00,0x07,0xfe,0x00,0x00,0x00
,0x00,0x00,0x00,0x0f,0x8e,0x00,0x00,0x00
,0x00,0x00,0x00,0x06,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
};
//*****
*****
***
void setup() {

    Serial.begin(115200);
    pinMode(D7,OUTPUT);
    servo.attach(D8);
    servo.write(130);

    //-----initiate OLED display-----

    // SSD1306_SWITCHCAPVCC = generate
    display voltage from 3.3V internally
```

```
}

// OK success!

p = finger.image2Tz(2);
switch (p) {
    case FINGERPRINT_OK:
        //Serial.println("Image converted");
        display.clearDisplay();
        display.drawBitmap( 34, 0,
FinPr_valid_bits, FinPr_valid_width,
FinPr_valid_height, WHITE);
        display.display();
        break;
    case FINGERPRINT_IMAGEMESS:
        Serial.println("Image too messy");
        return p;
    case FINGERPRINT_PACKETRECEIVEERR:
        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;
}
```

```

if(!display.begin(SSD1306_SWITCHCAPVCC,
0x3C)) { // Address 0x3D for 128x64
  Serial.println(F("SSD1306 allocation
failed"));
  for(;;); // Don't proceed, loop forever
}
// Show initial display buffer contents on
the screen --
// the library initializes this with an
Adafruit splash screen.
// you can delete these three lines if you
don't want to get the Adafruit logo appear
display.display();
delay(2000); // Pause for 2 seconds
display.clearDisplay();

//-----

connectToWiFi();

//-----

// set the data rate for the sensor serial
port
finger.begin(57600);
Serial.println("\n\nAdafruit finger detect
test");

if (finger.verifyPassword()) {

```

```

}

// OK converted!
Serial.print("Creating model for #");
Serial.println(id);

p = finger.createModel();
if (p == FINGERPRINT_OK) {
  //Serial.println("Prints matched!");
  display.clearDisplay();
  display.drawBitmap( 34, 0, FinPr_valid_bits,
FinPr_valid_width, FinPr_valid_height,
WHITE);
  display.display();
} else if (p ==
FINGERPRINT_PACKETRECEIVEERR) {
  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!");

```

<pre> Serial.println("Found fingerprint sensor!"); display.clearDisplay(); display.drawBitmap(34, 0, FinPr_valid_bits, FinPr_valid_width, FinPr_valid_height, WHITE); display.display(); } else { Serial.println("Did not find fingerprint sensor :("); display.clearDisplay(); display.drawBitmap(32, 0, FinPr_failed_bits, FinPr_failed_width, FinPr_failed_height, WHITE); display.display(); while (1) { delay(1); } } </pre>	<pre> display.clearDisplay(); display.drawBitmap(34, 0, FinPr_valid_bits, FinPr_valid_width, FinPr_valid_height, WHITE); display.display(); confirmAdding(); } else if (p == FINGERPRINT_PACKETRECEIVEERR) { 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; } } //*****Check if there a Fingerprint ID to add***** void confirmAdding(){ HTTPClient http; //Declare object of class HTTPClient //Post Data </pre>
---	---

	<pre> postData = "confirm_id=" + String(id); // Add the Fingerprint ID to the Post array in order to send it // Post methode http.begin(wifiClient, apiGetData); // get the result (**the error code**) http.addHeader("Content-Type", "application/x-www-form-urlencoded"); //Specify content-type header int httpCode = http.POST(postData); //Send the request String payload = http.getString(); //Get the response payload display.clearDisplay(); display.setTextSize(1.5); // Normal 1:1 pixel scale display.setTextColor(WHITE); // Draw white text display.setCursor(0,0); // Start at top- left corner display.print(payload); display.display(); delay(1000); Serial.println(payload); http.end(); //Close connection } </pre>
--	---

	<pre> //*****connect to the WiFi***** void connectToWiFi(){ WiFi.mode(WIFI_OFF); //Prevents reconnection issue (taking too long to connect) delay(1000); WiFi.mode(WIFI_STA); Serial.print("Connecting to "); Serial.println(ssid); WiFi.begin(ssid, password); display.clearDisplay(); display.setTextSize(1); // Normal 1:1 pixel scale display.setTextColor(WHITE); // Draw white text display.setCursor(0, 0); // Start at top-left corner display.print(F("Connecting to \n")); display.setCursor(0, 50); display.setTextSize(2); display.print(ssid); display.drawBitmap(73, 3, Wifi_start_bits, Wifi_start_width, Wifi_start_height, WHITE); display.display(); while (WiFi.status() != WL_CONNECTED) { delay(500); Serial.print("."); } </pre>
--	---

	<pre> Serial.println(""); Serial.println("Connected"); display.clearDisplay(); display.setTextSize(1.8); // Normal 1:1 pixel scale display.setTextColor(WHITE); // Draw white text display.setCursor(8, 2); // Start at top-left corner display.print(F("Connected \n")); display.drawBitmap(60, 0, Wifi_connected_bits, Wifi_connected_width, Wifi_connected_height, WHITE); display.setCursor(0, 50); display.setTextSize(2); display.print(ssid); display.display(); Serial.print("IP address: "); Serial.println(WiFi.localIP()); //IP address assigned to your ESP delay(500); } //===== ===== </pre>
--	--

== END ==