**FINGERPRINT BASED DOORLOCK SYSTEM**

EMPLOYABILITY SKILLS AND MINI PROJECT REPORT

SUBMITTE BY

**Tanvi Wadgaonkar(T400950072)**

**Shruti Suryavanshi(T400950063)**

**Snehal Kamble(T400950030)**

UNDER THE GUIDANCE OF

**PROF. PRASHANT AHIRE**

**TE (ELECTRONICS AND TELECOMMUNICATION)**



**DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION**

HOPE FOUNDATION’s

**INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY**

HINJAWADI, PUNE(MH)-411057

SAVITRIBAI PHULE PUNE UNIVERSITY

A.Y. 2020-25SEMESTER II

# CERTIFICATE

**DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION**

**HOPE FOUNDATION’s**

**INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY,**

**HINJAWADI, PUNE-411057**



This is to certify that

**Tanvi Wadgaonkar(T400950072)**

**Shruti Suryavanshi(T400950063)**

**Snehal Kamble(T400950030)**

Class: TE (E&TC) have successfully completed project titled, **“Fingerprint based doorlock system”** as a part of Semester II of Third Year of Bachelor of Engineering in Electronics and Telecommunication (A.Y. 2020-2021) of Savitribai Phule Pune University.

Prof. Prashant Ahire Dr. S. M. M Naidu

Project Guide HOD(E&TC)

Dr. Vaishali V Ptail

Principal

Place : Pune Date : External

# ABSTRACT

This project focuses on developing a fingerprint-based smart door lock system using the ESP32 microcontroller. The main goal is to improve security by allowing only authorized users to unlock the door using their fingerprints. The system uses an R307 fingerprint sensor to capture and verify fingerprints. When a user places their finger on the sensor, it scans the fingerprint and sends the data to the ESP32. The microcontroller then compares the scanned fingerprint with the ones stored in its memory. If a match is found, the system activates a relay module to unlock the door. If the fingerprint does not match any stored data, access is denied and the door remains locked. An OLED display is included to show system status messages such as “Place Finger,” “Access Granted,” or “Access Denied,” which makes the system more user-friendly. The system can store multiple fingerprints, making it suitable for homes, offices, and other secure areas. This project offers a reliable, keyless, and easy-to-use solution for modern access control.

This project presents a smart door lock system that uses fingerprint recognition for secure access control. The system is built using an ESP32 microcontroller and an R307 fingerprint sensor. When a user places their finger on the sensor, it scans the fingerprint and compares it with the stored fingerprints. If a match is found, the ESP32 unlocks the door by triggering a relay connected to an electronic lock. If the fingerprint does not match, the door remains locked. The system provides a reliable and convenient alternative to traditional keys, reducing the risk of unauthorized access. It is ideal for use in homes, offices, and other secure areas.

Contents

[CERTIFICATE ii](#_Toc197077857)

[ABSTRACT iii](#_Toc197077858)

[List of Figures vi](#_Toc197077859)

[List of Tables vii](#_Toc197077860)

[LIST OF ABBREVIATIONS viii](#_Toc197077861)

[Chapter 1 1](#_Toc197077862)

[1.1 Introduction 1](#_Toc197077863)

[1.2 Fingerprint Based Doorlock 1](#_Toc197077864)

[Chapter 2 3](#_Toc197077865)

[2.1 Literature Survey 3](#_Toc197077866)

[Chapter 3 5](#_Toc197077867)

[Proposed Methodology 5](#_Toc197077868)

[3.1 Problem Statement 5](#_Toc197077869)

[3.2 Problem Motivation 6](#_Toc197077870)

[3.3 Process description 7](#_Toc197077871)

[3.4 Requirement Analysis 8](#_Toc197077872)

[3.5 Impact Analysis 10](#_Toc197077873)

[3.6 Professional Ethics Practices to be followed 11](#_Toc197077874)

[Chapter 4 13](#_Toc197077875)

[Project Implementation 13](#_Toc197077876)

[4.1 Circuit Designing 13](#_Toc197077877)

[4.2 PCB designing 14](#_Toc197077878)

[4.3 Programming 15](#_Toc197077879)

[4.4 Project Component and Cost Estimate 30](#_Toc197077880)

[Chapter 5 31](#_Toc197077881)

[RESULTS 31](#_Toc197077882)

[Chapter 6 32](#_Toc197077883)

[References 32](#_Toc197077884)

# List of Figures

**Figure No Figure Name**

Fig 4.1 Circuit Diagram of fingerprint based doorlock

Fig 4.2 PCB layout

Fig 4.3 Actual PCB design

# List of Tables

**Table No Table Name**

Table 4.1 cost estimation

### LIST OF ABBREVIATIONS

|  |  |
| --- | --- |
| **ESP32** | Espressif Systems 32-bit Microcontroller |
| **GPIO** | General Purpose Input/Output |
| **R307** | Model Number of Optical Fingerprint Sensor |
| **LCD** | Liquid Crystal Display |
| **LED** | Light Emitting Diode |
| **RTC** | Real Time Clock |
| **PWM** | Pulse Width Modulation |
| **DC** | Direct Current |
| **USB** | Universal Serial Bus |
| **UART** | Universal Asynchronous Receiver Transmitter |
| **EEPROM** | Electrically Erasable Programmable Read-Only Memory |
| **I2C** | Inter-Integrated Circuit |
| **SPI** | Serial Peripheral Interface |
| **OLED** | Organic Light Emitting Diode (if used for display) |
| **BJT** | Bipolar Junction Transistor (if used in circuit design) |
| **MOSFET** | Metal Oxide Semiconductor Field Effect Transistor |

## Chapter 1

## 1.1 Introduction

Traditional door locking systems have been used for centuries to protect homes and properties, usually relying on mechanical keys. However, these systems have limitations, such as the risk of losing keys, duplication, or forced entry. As technology advanced, electronic locks and keypads were introduced to improve convenience and security. In recent years, **biometric systems**—especially fingerprint recognition—have become popular for access control due to their uniqueness and difficulty to duplicate. Fingerprints are widely used in smartphones, offices, and high-security areas because they offer a quick and reliable way to verify identity.With the development of affordable and powerful microcontrollers like the **ESP32**, it has become possible to build smart and connected security systems at a low cost. The **R307 fingerprint sensor** is a commonly used module for biometric authentication due to its ease of use, reliability, and ability to store multiple fingerprints.By combining these technologies—ESP32, R307 fingerprint sensor, and OLED display—this project brings together modern electronics and biometric security to create a smart, user-friendly, and secure door lock system.

### 1.2 Fingerprint Based Doorlock

With the increasing demand for secure and intelligent access control systems, biometric authentication has emerged as a reliable and efficient solution. Traditional mechanical locks and even keypad-based electronic locks are vulnerable to unauthorized access due to key duplication, password leakage, or physical tampering. To address these security challenges, biometric systems—particularly fingerprint recognition—are widely adopted, offering enhanced reliability by utilizing physiological traits that are unique to each individual.

This project involves the design and implementation of a **fingerprint-based door lock system** using the **ESP32 microcontroller**, the **R307 optical fingerprint sensor**, and an **OLED display for user interaction**. The ESP32 is a powerful and versatile microcontroller with built-in Wi-Fi and Bluetooth capabilities, making it ideal for smart IoT-based applications. The R307 fingerprint module is used to enroll, store, and match fingerprints locally, ensuring fast and offline biometric authentication.

The system works by capturing a fingerprint image from the sensor and matching it against the enrolled fingerprint templates stored in the sensor’s flash memory. Upon a successful match, the ESP32 triggers a **relay module** to activate an **electronic door lock**. The **OLED display** provides real-time system status and user feedback, such as fingerprint enrollment, verification results, and access granted or denied messages.

The primary objective of this project is to develop a **low-cost, scalable, and secure biometric access system** that eliminates the need for keys or passcodes, while offering an intuitive user experience. The project also lays the groundwork for future expansion, such as remote monitoring, mobile app integration, or cloud-based access management, leveraging the connectivity features of the ESP32.

## Chapter 2

## 2.1 Literature Survey

In recent years, biometric authentication systems have gained significant attention for enhancing security in access control applications. Fingerprint recognition, in particular, has emerged as a reliable and cost-effective biometric method due to its uniqueness, stability, and ease of implementation.

discuss various biometric authentication techniques and conclude that fingerprint recognition strikes a balance between usability, performance, and security, making it ideal for embedded access systems. Compared to traditional authentication methods such as keys or PINs, fingerprints cannot be easily duplicated or shared, thereby reducing the chances of unauthorized access.

The R307 optical fingerprint sensor has been widely adopted in microcontroller-based systems due to its built-in image processing and onboard fingerprint matching capability. Projects using similar modules such as the R305 and GT511C3 [2] have demonstrated successful local fingerprint verification in low-power environments, without relying on external computing devices.

Microcontrollers such as the **ESP32** have proven to be powerful platforms for building IoT-enabled security systems. Its dual-core processor, Wi-Fi, and Bluetooth connectivity enable real-time control and expandability. Espressif’s documentation and several IoT-based research projects [3] confirm the ESP32’s suitability for secure, connected applications like smart door locks and monitoring systems.

User interaction is another crucial component in embedded security systems. OLED displays are preferred in compact embedded systems for their low power usage and high contrast. Research [4] indicates that real-time feedback via displays improves the usability and reliability of such systems, guiding users through enrollment and verification steps.

Existing literature also highlights some limitations, including the potential for false rejections due to dirty or damaged fingers and spoofing risks. To mitigate these, some systems integrate multiple authentication factors or log data to the cloud for audit purposes [5].

This project builds upon these technologies by integrating the R307 fingerprint sensor with an ESP32 microcontroller and OLED display, creating a robust, offline-capable fingerprint-based door locking system suitable for home and office use.

## Chapter 3

## Proposed Methodology

### 3.1 Problem Statement

Conventional door locking systems, such as mechanical locks and keypad-based systems, are vulnerable to security threats like key loss, duplication, and password leakage. These methods also lack efficient access control and user authentication. To overcome these challenges, there is a need for a secure, reliable, and user-friendly system that can provide controlled access based on biometric identification. This project aims to design a fingerprint-based smart door lock system using the ESP32 microcontroller, R307 fingerprint sensor, and OLED display to enhance security and convenience in residential and commercial settings.

**Objective:**

1. To develop a fingerprint-based door lock system using ESP32 and R307 sensor.

2. To ensure secure user authentication through fingerprint matching.

3. To display system status using an OLED screen for better user interaction.

4. To control a relay-based electronic lock on successful verification.

5. To create a low-cost, standalone, and user-friendly access control system

### 3.2 Problem Motivation

In the modern world, security is a critical concern for both individuals and organizations. Whether it is residential buildings, office spaces, or restricted areas, controlling and monitoring access has become essential to prevent unauthorized entry, theft, or misuse. Traditional security systems, such as mechanical locks and password-based access, have served this purpose for decades. However, these systems suffer from serious limitations. Keys can be lost, stolen, or duplicated, and passwords can be forgotten, guessed, or leaked. These vulnerabilities make such systems unreliable for applications where high security is required.

With advancements in electronics and embedded systems, it is now possible to develop smarter and more secure alternatives using biometric technology. Among the various biometric methods available—such as facial recognition, iris scanning, and voice recognition—**fingerprint recognition** has proven to be the most practical and widely adopted. It is accurate, affordable, fast, and easy to implement, making it highly suitable for access control systems.

The motivation behind this project is to develop a **low-cost, standalone, fingerprint-based door locking system** that offers a higher level of security without relying on physical keys or internet connectivity. Using **biometric authentication** eliminates the risks associated with key loss and password sharing. The integration of an **ESP32 microcontroller** with the **R307 fingerprint sensor** enables efficient processing and local storage of fingerprints. Additionally, a **0.96" OLED display** provides a user-friendly interface that guides users through fingerprint enrollment and verification, enhancing the overall experience.

### 3.3 Process description

**1. System Initialization**

* When the system is powered on, the ESP32 initializes all connected hardware peripherals including the **R305 fingerprint module**, **OLED display**, and **relay module**.
* The OLED displays a status message such as **"System Ready"** or **"Place Finger"** indicating that the system is operational and ready for input.

**2. Fingerprint Enrollment**

* The enrollment process is used to register new users:

Triggered by an admin mode (via button or admin fingerprint).

The OLED displays a message like **"Enroll New Finger"**.

The R305 sensor prompts the user to place their finger on the sensor twice (or more)

To generate a consistent fingerprint template.

The captured fingerprint is processed and converted into a digital template which

is stored in the internal memory of the R305 with a unique ID.

On success, the system shows **"Fingerprint Enrolled Successfully"** on the OLED.

**3. Fingerprint Authentication / Verification**

* In normal mode, the system waits for a user to place their finger on the sensor.
* When a fingerprint is detected, the R305 captures the fingerprint image and internally compares it to stored templates.
* If a match is found:

The sensor notifies the ESP32.

The ESP32 proceeds to unlock the door.

* If no match is found:

The OLED displays **"Access Denied"**, and the door remains locked.

**4. Door Lock Control**

* On successful fingerprint match:

The ESP32 sends a HIGH signal to the **relay module**, activating it.

This unlocks the electronic door lock (or controls a solenoid/magnetic lock).

The OLED displays **"Access Granted"**.

After a short delay the ESP32 sends a LOW signal to the relay relocking the door.

**5. OLED User Interface**

* The OLED provides continuous visual feedback to the user throughout all stages:

System status,

Enrollment steps,

Authentication results,

Errors (e.g., “Try Again,” “Sensor Error”).

**6. Administration and Maintenance**

* The system may include an **admin fingerprint ID** or button for:

Adding/removing fingerprints,

Clearing memory,

Entering special settings mode.

* This ensures the system is manageable only by authorized personnel.

### 3.4 Requirement Analysis

**1. Techniques Used**

* **Biometric Authentication**: Utilization of fingerprint scanning and matching for identity verification, ensuring high security and uniqueness for each user.
* **Embedded System Programming**: Code development using C/C++ for ESP32 in Arduino IDE or PlatformIO environment for real-time control and device communication.
* **UART Serial Communication**: Communication protocol between the ESP32 microcontroller and the R305 fingerprint sensor.
* **Relay Switching**: Electronic switching technique used to control the lock mechanism based on fingerprint match status.
* **OLED Display Interface**: I2C communication is used to display user prompts, system status, and feedback to enhance user interaction.

**2. Hardware Resources Required**

* **ESP32 Microcontroller**: Core processing unit responsible for controlling all hardware, managing logic, and communication.
* **R305 Fingerprint Sensor**: Captures and verifies fingerprint data. Stores user templates and communicates with the ESP32 via UART.
* **OLED Display (0.96” I2C)**: Displays real-time system status, prompts, and messages to the user.
* **Relay Module**: Electrically switches the door lock based on the verification result.
* **Electronic Lock (e.g., Solenoid or Magnetic)**: Physical locking mechanism controlled via relay.
* **Push Buttons**: For enrolling, deleting, or entering admin modes (if included).
* **Power Supply (5V/9V/12V depending on the lock)**: Powers all the components.
* **Breadboard, jumper wires, resistors, casing**: For prototyping and assembly.

**3. Software Tools Required**

* **Arduino IDE**: Used for writing, compiling, and uploading code to the ESP32.
* **Fingerprint Library**: Used for interfacing with the R305 module (Adafruit Fingerprint or similar).
* **Wire.h & Adafruit\_SSD1306.h Libraries**: To interface and control the OLED display.
* **Serial Monitor**: For debugging and monitoring sensor outputs during development.

**4. Modern Engineering Tools and Methodologies**

* **Modular Coding**: Code structured into reusable functions and modules for easier testing and debugging.
* **Version Control (Optional)**: Git or GitHub for source code management and collaboration.
* **Simulation and Testing Tools**: Serial output, LED indicators, and test prints to verify logic without activating the physical lock during early stages.
* **Rapid Prototyping**: Breadboard-based testing before final soldering and enclosure assembly.

**5. Human-Machine Interaction**

* OLED screen provides user-friendly feedback for every action like enrollment, matching, success, or failure.
* Buttons (if used) enable direct physical interaction for managing fingerprint templates.

This requirement analysis ensures a structured development approach using **affordable hardware**, **open-source tools**, and **efficient engineering techniques**, resulting in a secure and practical fingerprint-based door lock solution.

### 3.5 Impact Analysis

Impact on Environment:

1.Low power consumption and energy-efficient components.

2. Components are reusable and recyclable.

3. Potential e-waste if not disposed of properly.

4. Batteries (if used) need eco-friendly handling.

Impact Analysis:

1. Provides enhanced security through biometric access.

2. Offers keyless and password-free convenience.

3. Useful in homes, offices, labs, and restricted areas.

4. Encourages adoption of smart and secure technology.

5. Minor risk if fingerprint data is not securely stored.

6. System failure may restrict access without backup.

### 3.6 Professional Ethics Practices to be followed

**1. Safety and Security**

* Ensure that the system is reliable and does not pose any harm to users.
* Handle biometric data responsibly to protect users’ privacy and identity.

**2. Honesty and Integrity**

* Provide accurate results and data in system testing and documentation.
* Avoid plagiarism by giving credit for open-source libraries or references used.

**3. Responsibility and Accountability**

* Take full responsibility for the design, development, and performance of the system.
* Fix bugs, improve the system, and ensure regular maintenance if deployed.

**4. Environmental Responsibility**

* Use components that are energy-efficient and minimize electronic waste.
* Ensure proper disposal or recycling of electronic parts and batteries.

**5. Respect for Intellectual Property**

* Use licensed tools and libraries.
* Avoid unauthorized duplication or misuse of third-party hardware or software.

**6. Continuous Learning**

* Keep up with emerging technologies in IoT and biometrics to improve the project.
* Be open to feedback and make improvements wherever necessary.

## Chapter 4

## Project Implementation

### 4.1 Circuit Designing

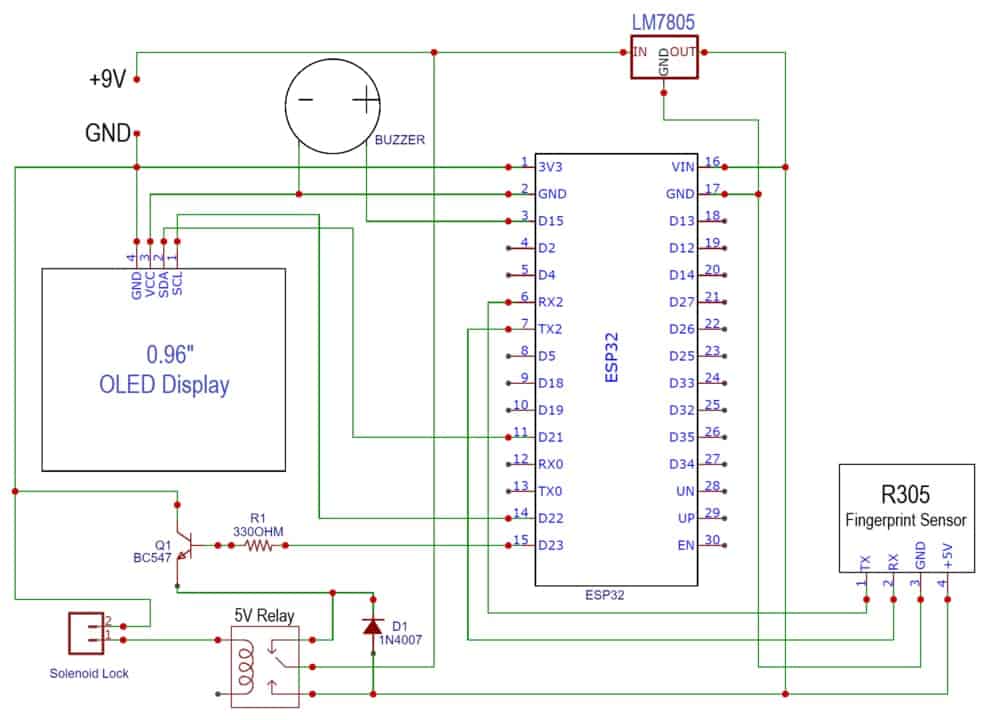


Fig 4.1 Circuit Diagram of fingerprint based doorlock

### 4.2 PCB designing

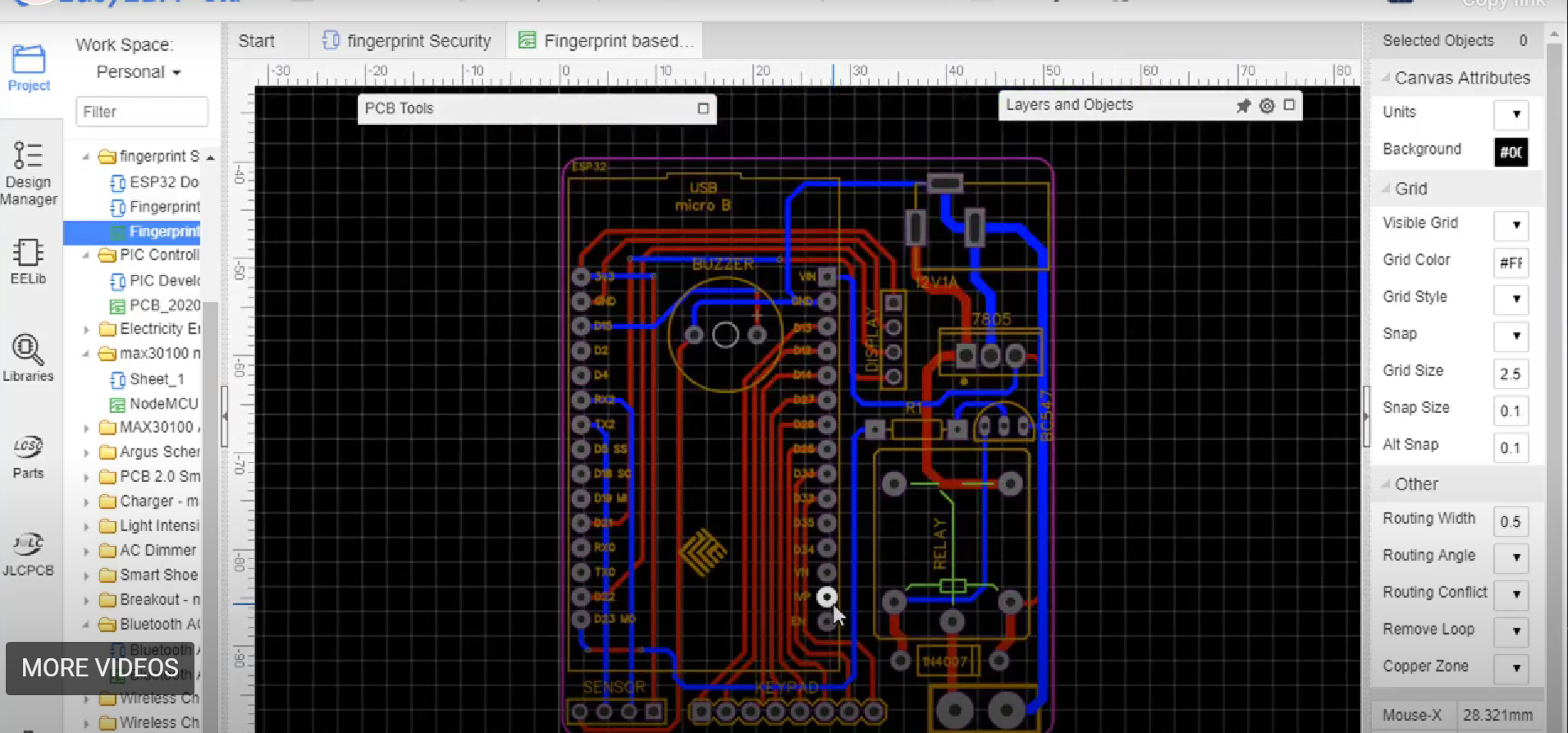


Fig 4.2 Pcb layout

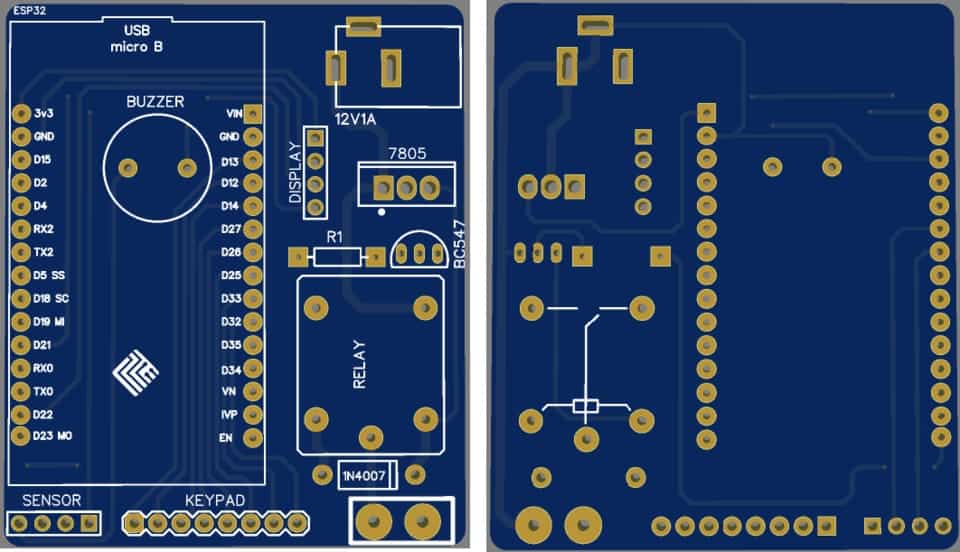


Fig 4.3 Actual PCB design

### 4.3 Programming

making this **Door Lock Security System**  need to **enroll the fingerprint**. The only an enrolled finger can be used to open the door. The unregistered user won’t be able to get any access. The code for **enrolling the fingerprint** is given below

#include <Adafruit\_Fingerprint.h>

#include <HardwareSerial.h>

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&Serial2);

uint8\_t id;

**void** setup()

{

Serial.begin(57600);

Serial2.begin(115200);

**while** (!Serial); *// For Yun/Leo/Micro/Zero/...*

delay(100);

Serial.println("\n\nAdafruit Fingerprint sensor enrollment");

*// set the data rate for the sensor serial port*

finger.begin(57600);

**if** (finger.verifyPassword()) {

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

} **else** {

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

**while** (1) { delay(1); }

}

}

uint8\_t readnumber(**void**) {

uint8\_t num = 0;

**while** (num == 0) {

**while** (! Serial.available());

num = Serial.parseInt();

}

**return** num;

}

**void** loop() *// run over and over again*

{

Serial.println("Ready to enroll a fingerprint!");

Serial.println("Please type in the ID # (from 1 to 127) you want to save this finger as...");

id = readnumber();

**if** (id == 0) {*// ID #0 not allowed, try again!*

**return**;

}

Serial.print("Enrolling ID #");

Serial.println(id);

**while** (! getFingerprintEnroll() );

}

uint8\_t getFingerprintEnroll() {

**int** p = -1;

Serial.print("Waiting for valid finger to enroll as #"); Serial.println(id);

**while** (p != FINGERPRINT\_OK) {

p = finger.getImage();

**switch** (p) {

**case** FINGERPRINT\_OK:

Serial.println("Image taken");

**break**;

**case** FINGERPRINT\_NOFINGER:

Serial.println(".");

**break**;

**case** FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

**break**;

**case** FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

**break**;

**default**:

Serial.println("Unknown error");

**break**;

}

}

*// OK success!*

p = finger.image2Tz(1);

**switch** (p) {

**case** FINGERPRINT\_OK:

Serial.println("Image converted");

**break**;

**case** FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

**return** p;

**case** FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

**return** p;

**case** FINGERPRINT\_FEATUREFAIL:

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

**return** p;

**case** FINGERPRINT\_INVALIDIMAGE:

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

**return** p;

**default**:

Serial.println("Unknown error");

**return** p;

}

Serial.println("Remove finger");

delay(2000);

p = 0;

**while** (p != FINGERPRINT\_NOFINGER) {

p = finger.getImage();

}

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

p = -1;

Serial.println("Place same finger again");

**while** (p != FINGERPRINT\_OK) {

p = finger.getImage();

**switch** (p) {

**case** FINGERPRINT\_OK:

Serial.println("Image taken");

**break**;

**case** FINGERPRINT\_NOFINGER:

Serial.print(".");

**break**;

**case** FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

**break**;

**case** FINGERPRINT\_IMAGEFAIL:

Serial.println("Imaging error");

**break**;

**default**:

Serial.println("Unknown error");

**break**;

}

}

*// OK success!*

p = finger.image2Tz(2);

**switch** (p) {

**case** FINGERPRINT\_OK:

Serial.println("Image converted");

**break**;

**case** FINGERPRINT\_IMAGEMESS:

Serial.println("Image too messy");

**return** p;

**case** FINGERPRINT\_PACKETRECIEVEERR:

Serial.println("Communication error");

**return** p;

**case** FINGERPRINT\_FEATUREFAIL:

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

**return** p;

**case** FINGERPRINT\_INVALIDIMAGE:

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

**return** p;

**default**:

Serial.println("Unknown error");

**return** p;

}

*// OK converted!*

Serial.print("Creating model for #"); Serial.println(id);

p = finger.createModel();

**if** (p == FINGERPRINT\_OK) {

Serial.println("Prints matched!");

} **else** **if** (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

**return** p;

} **else** **if** (p == FINGERPRINT\_ENROLLMISMATCH) {

Serial.println("Fingerprints did not match");

**return** p;

} **else** {

Serial.println("Unknown error");

**return** p;

}

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

p = finger.storeModel(id);

**if** (p == FINGERPRINT\_OK) {

Serial.println("Stored!");

} **else** **if** (p == FINGERPRINT\_PACKETRECIEVEERR) {

Serial.println("Communication error");

**return** p;

} **else** **if** (p == FINGERPRINT\_BADLOCATION) {

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

**return** p;

} **else** **if** (p == FINGERPRINT\_FLASHERR) {

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

**return** p;

} **else** {

Serial.println("Unknown error");

**return** p;

}

Once the enrolling is done, we can upload another code to the ESP32 board. The code is capable of **locking** and **unlocking the door** based on matched and unmatched cases. The code is given below.

#include <Adafruit\_Fingerprint.h>   //https://github.com/adafruit/Adafruit-Fingerprint-Sensor-Library

#include <HardwareSerial.h>

#include <Adafruit\_GFX.h>           //https://github.com/adafruit/Adafruit-GFX-Library

#include <Adafruit\_SSD1306.h>       //https://github.com/adafruit/Adafruit\_SSD1306

#include <SPI.h>

#include <Wire.h>

#define SCREEN\_WIDTH 128 // OLED display width, in pixels

#define SCREEN\_HEIGHT 64 // OLED display height, in pixels

*// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)*

#define OLED\_RESET     -1 // Reset pin # (or -1 if sharing Arduino reset pin)

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET);

*// Icon of Fingerprint*

#define LOGO\_HEIGHT   64

#define LOGO\_WIDTH    128

static const **unsigned** **char** PROGMEM logo\_bmp[] =

{

  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, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xff, 0xfc, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xff, 0x83, 0xff, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x0f, 0xf0, 0x00, 0x0f, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0x80, 0x00, 0x03, 0xf8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x7e, 0x03, 0xff, 0xc0, 0xfe, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0xfc, 0x3f, 0xff, 0xf8, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0xf0, 0xff, 0xff, 0xfe, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0xe1, 0xff, 0x00, 0xff, 0x87, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0xc7, 0xf0, 0x00, 0x1f, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0x8f, 0xc0, 0x00, 0x03, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0x03, 0xff, 0xc1, 0xf8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x3e, 0x0f, 0xff, 0xf0, 0x7c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x7c, 0x3f, 0xff, 0xfc, 0x3e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0xf8, 0xff, 0x00, 0xff, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0xf1, 0xf8, 0x00, 0x1f, 0x8f, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0xe3, 0xf0, 0x18, 0x0f, 0xc7, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x03, 0xc7, 0xc1, 0xff, 0x83, 0xe3, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x03, 0xc7, 0x87, 0xff, 0xe1, 0xf1, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x07, 0x8f, 0x1f, 0xff, 0xf0, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x07, 0x1e, 0x3f, 0x01, 0xf8, 0x78, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x07, 0x1e, 0x7e, 0x00, 0x7c, 0x79, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x3c, 0x78, 0x00, 0x3e, 0x3d, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x78, 0xf0, 0xff, 0x1f, 0x3d, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x79, 0xe1, 0xff, 0x8f, 0x1c, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0xf1, 0xe3, 0xff, 0xc7, 0x1e, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0xe3, 0xc7, 0xc3, 0xc7, 0x9e, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x07, 0xe3, 0x8f, 0x01, 0xe7, 0x9e, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x0f, 0xc7, 0x8f, 0x00, 0xe3, 0x9e, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x0f, 0x8f, 0x9e, 0x18, 0xf3, 0x9e, 0x70, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x0e, 0x0f, 0x1e, 0x38, 0xf3, 0x9e, 0x70, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x1e, 0x3c, 0x38, 0xf3, 0x9e, 0x70, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x3c, 0x3c, 0x78, 0xf7, 0x9c, 0x70, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x7c, 0x78, 0x78, 0xe7, 0x9c, 0x70, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0xf0, 0xf0, 0xf1, 0xe7, 0x9c, 0x70, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x03, 0xe1, 0xf1, 0xe1, 0xe7, 0x3c, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x0f, 0xc3, 0xe3, 0xe3, 0xc7, 0x3c, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x0f, 0x87, 0xc3, 0xc3, 0xcf, 0x3c, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x0f, 0x0f, 0x87, 0x87, 0x8f, 0x3c, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0x0f, 0x0f, 0x8f, 0x38, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x7e, 0x1f, 0x1f, 0x0f, 0x3c, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0xfc, 0x3e, 0x1e, 0x0f, 0x3d, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0xf0, 0x7c, 0x3c, 0xcf, 0x3c, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x07, 0xe0, 0xf8, 0x79, 0xcf, 0x1e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x0f, 0xc1, 0xf0, 0xf9, 0xcf, 0x1e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x0f, 0x07, 0xe1, 0xf1, 0xc7, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x0e, 0x0f, 0xc3, 0xe3, 0xc7, 0x87, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0x07, 0xc3, 0xc7, 0x87, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x3e, 0x1f, 0x83, 0xc3, 0xc3, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0xfc, 0x3e, 0x01, 0xc3, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0xf8, 0x7c, 0x01, 0xc1, 0xf8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0xf0, 0xf8, 0x01, 0xe0, 0xfc, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x01, 0xc3, 0xf3, 0xf8, 0xf0, 0x7e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xe7, 0xfc, 0xf8, 0x1c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x0f, 0xcf, 0xfe, 0x7c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x0f, 0x1f, 0x1e, 0x3e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x0e, 0x3e, 0x0f, 0x1e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3c, 0x0f, 0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x18, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00

};

Adafruit\_Fingerprint finger = Adafruit\_Fingerprint(&Serial2);

**int** relayPin = 23;

**int** buzzerPin = 15;

**void** setup()

{

pinMode(relayPin, OUTPUT);

pinMode(buzzerPin, OUTPUT);

digitalWrite(relayPin, LOW);

digitalWrite(buzzerPin, LOW);

Serial.begin(57600);

Serial2.begin(115200);

**if**(!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C))

{

    Serial.println(F("SSD1306 allocation failed"));

**for**(;;); *// Don't proceed, loop forever*

  }

**while** (!Serial);

delay(100);

display.clearDisplay();

display.drawBitmap(0, 0, logo\_bmp, LOGO\_WIDTH, LOGO\_HEIGHT, 1);

display.display();

Serial.println("Fingerprint Door Lock");

delay(3000);

display.clearDisplay();

*// set the data rate for the sensor serial port*

finger.begin(57600);

**if** (finger.verifyPassword())

{

   Serial.println("Fingerprint Sensor Connected");

   display.clearDisplay();

   display.setTextSize(2);             *// Normal 1:1 pixel scale*

   display.setTextColor(SSD1306\_WHITE);        *// Draw white text*

   display.setCursor(25, 0);            *// Start at top-left corner*

   display.println(("Sensor"));

   display.setCursor(5, 35);

   display.println("Connected");

   display.display();

   delay(3000);

*// display.clearDisplay();*

}

**else**

{

   display.clearDisplay();

   display.setTextSize(2);             *// Normal 1:1 pixel scale*

   display.setTextColor(SSD1306\_WHITE);        *// Draw white text*

   display.setCursor(25, 0);            *// Start at top-left corner*

   display.println(("Sensor"));

   display.setCursor(5, 35);

   display.println("Not Found");

   display.display();

   Serial.println("Unable to find Sensor");

   delay(3000);

   Serial.println("Check Connections");

**while** (1) {

     delay(1);

   }

}

display.clearDisplay();

}

**void** loop()                     *// run over and over again*

{

getFingerprintIDez();

delay(50);            *//don't need to run this at full speed.*

}

*// returns -1 if failed, otherwise returns ID #*

**int** getFingerprintIDez()

{

uint8\_t p = finger.getImage();

**if** (p != FINGERPRINT\_OK)

{

   display.clearDisplay();

   display.drawBitmap(0, 0, logo\_bmp, LOGO\_WIDTH, LOGO\_HEIGHT, 1);

   display.display();

   Serial.println("Waiting For Valid Finger");

**return** -1;

}

p = finger.image2Tz();

**if** (p != FINGERPRINT\_OK)

{

   display.clearDisplay();

   display.setTextSize(2);             *// Normal 1:1 pixel scale*

   display.setTextColor(SSD1306\_WHITE);        *// Draw white text*

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

   display.println(("Messy  Image"));

   display.setCursor(0, 35);

   display.println("Try Again");

   display.display();

   Serial.println("Messy Image Try Again");

   delay(3000);

   display.clearDisplay();

**return** -1;

}

p = finger.fingerFastSearch();

**if** (p != FINGERPRINT\_OK)  {

   display.clearDisplay();

   display.setTextSize(2);             *// Normal 1:1 pixel scale*

   display.setTextColor(SSD1306\_WHITE);        *// Draw white text*

   display.setCursor(20, 0);            *// Start at top-left corner*

   display.println(("Invalid"));

   display.setCursor(25, 20);

   display.println("Finger");

   display.setCursor(10, 40);

   display.println("Try Again");

   display.display();

   Serial.println("Not Valid Finger");

   delay(3000);

   display.clearDisplay();

**return** -1;

}

*// found a match!*

   display.clearDisplay();

   display.setTextSize(2);             *// Normal 1:1 pixel scale*

   display.setTextColor(SSD1306\_WHITE);        *// Draw white text*

   display.setCursor(40, 0);            *// Start at top-left corner*

   display.println(("Door"));

   display.setCursor(15, 20);            *// Start at top-left corner*

   display.println(("Unlocked"));

   display.setCursor(20, 40);

   display.println("Welcome");

   display.display();

   digitalWrite(relayPin, HIGH);

   digitalWrite(buzzerPin, HIGH);

   delay(3000);

   display.clearDisplay();

   display.setTextSize(2);             *// Normal 1:1 pixel scale*

   display.setTextColor(SSD1306\_WHITE);        *// Draw white text*

   display.setCursor(20, 10);            *// Start at top-left corner*

   display.println(("Closing"));

   display.setCursor(20, 30);            *// Start at top-left corner*

   display.println(("the Door"));

   display.display();

   digitalWrite(relayPin, LOW);

   digitalWrite(buzzerPin, LOW);

   delay(3000);

   display.clearDisplay();

   Serial.println("Door Unlocked Welcome");

**return** finger.fingerID;

}

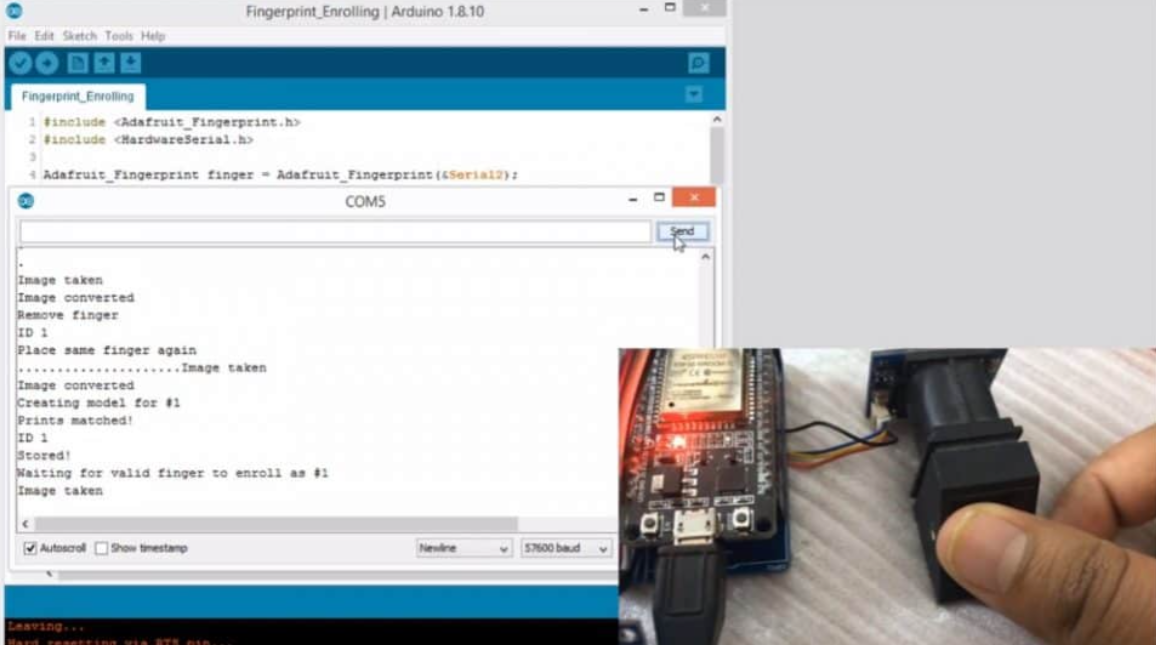
### 4.4 Project Component and Cost Estimate

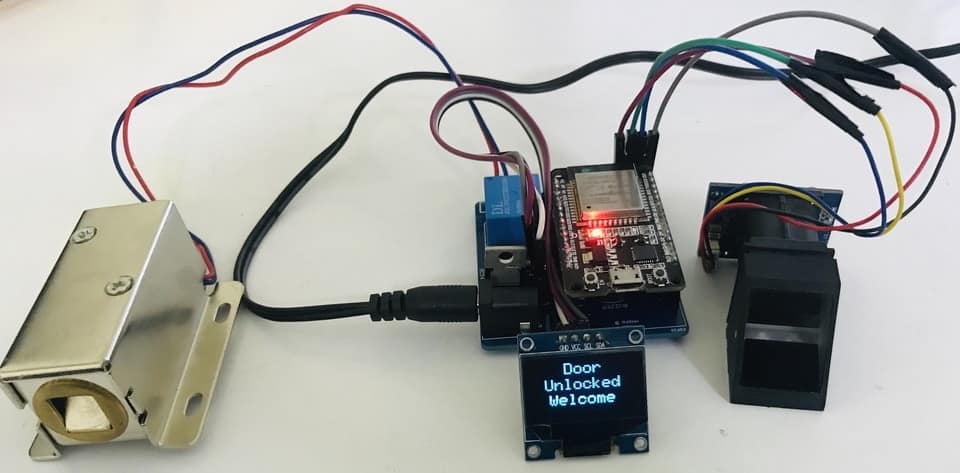
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Srno | Component | Quantity | Cost/unit | Total cost |
| 1 | ESP32 Board | 1 | 489 | 489 |
| 2 | Fingerprint Sensor | 2 | 802 | 1604 |
| 3 | Solenoid Lock | 2 | 350 | 700 |
| 4 | OLED Display | 1 | 790 | 790 |
| 5 | Relay 5V | 1 | 45 | 45 |
| 6 | Voltage Regulator 7806 | 1 | 175 | 175 |
| 7 | Diode 1N4007 | 1 | 39 | 39 |
| 8 | Resistor 330-ohm | 1 | 15 | 15 |
| 9 | Buzzer 5V | 1 | 175 | 175 |
| 10 | BC547 Transistor | 1 | 29 | 29 |
| 11 | DC Power Jack | 2 | 249 | 498 |
| 12 | 12-0-12 3A transistor | 1 | 199 | 199 |
| 13 | 3A Bridge rectifier | 4 | 50 | 200 |
| 14 | IC 7809 | 2 | 149 | 298 |
| 15 | IC 7812 | 2 | 169 | 338 |
| 16 | IC 7805 | 2 | 200 | 400 |
| 17 | Female Headers | Multiple Jack | 249 | 249 |
| 18 | Connecting Wires | Multiple Jack | 198 | 198 |
|  |  |  | Total cost | 6441 |

Table 4.1 cost estimation

## Chapter 5

### RESULTS

****

**Fig 5.1 Output on Arduino IDE**

**Fig 5.2 Final Output**

## Chapter 6

### References

[1] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 4–20, Jan. 2004.

[2] R. Kumar and A. Agrawal, "Design and implementation of biometric access control system using Arduino and fingerprint sensor," *International Journal of Scientific & Engineering Research*, vol. 7, no. 4, pp. 123–128, 2016.

[3] Espressif Systems, “ESP32 Technical Reference Manual,” 2022. [Online]. Available: https://www.espressif.com/en/products/socs/esp32

[4] S. Patil and A. Kulkarni, "Smart door locking system using fingerprint authentication," *International Journal of Engineering Research and Technology (IJERT)*, vol. 9, no. 6, 2020.

[5] M. Al-Qurabat, H. A. Mohammed, and S. A. Khudhur, "IoT-based smart door lock system with face and fingerprint recognition," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 15, no. 20, pp. 96–109, 2021