# Design of a Mechatronic System Using Arduino and/or Raspberry Pi 2

**(A3073-221)**

**Submitted by: Lim Zheng Ting**

**Matriculation Number: U2022755K**

**Supervisor: Prof Cheng Tee Hiang**

**Examiner: Ast/P Amer Mohammad Yusuf Mohammad Ghias**

**School of Electrical & Electronic Engineering**

A final year project report presented to the Nanyang Technological University in partial fulfilment of the requirements of the degree of Bachelor of Engineering 2022/2023 Semester 2 (Aug 22 - May 23)

**Table of Contents**

# ABSTRACT

Locker systems are secure storage containers used for users to store various things. However, locker theft is caused by conventional locking methods that are insecure and are easily hackable. There are biometric methods used to authenticate users but these are not used in locker systems currently.

In this project, a locker system using secure authentication methods to unlock will be built. Fingerprint sensor and face recognition are used to authenticate the user's fingerprint or face respectively.

An Arduino UNO board will be used as the mainboard to power the locker system and ESP32-CAM will be used to power the face recognition system. Standard off-the-shelf components will be used for the locker system and fingerprint sensor. Machine learning will be used to create the face recognition system. The prototype will be built in a smaller form made of corrugated cardboard to validate the capabilities of the locker system.

# ACKNOWLEDGEMENT

I would like to sincerely thank everyone who has supported me throughout the course of the Final Year Project. I am grateful for all the feedback and suggestions I have received during the process of the project.

I would also like to thank Prof. Cheng Tee Hiang for making the time for presentations and reading my logbooks. His constructive advice and suggestions have tremendously improved my prototype and ultimately my progress.

Lastly, I would also like to thank everyone who has contributed to my market research survey. Your feedback and suggestions have made my overall direction of the project smoother and more coherent.

# Acronyms

FYP – Final Year Project

LCD – Liquid Crystal Display

FNN – Fully Convolutional Network

CNN – Convolutional Neural Network

NMS – Non-maximum Supression

ICSP- In-Circuit Serial Programming

DSP – Digital Signal Processing

TTL – Transistor-Transistor Logic

FTDI – Future Technology Services International Limited

VCC – Voltage at Common Collector

TX - Transmit

RX - Receive

SCL – Serial Clock

SDA – Serial Data

CAD – Computer-aided Design

PSRAM – Pseudostatic Random Access Memory

CPU – Central Power Unit

GPU – Graphics Power Unit

OOM – Out of Memory

MB - Mega Bytes

# Table of Figures

# 1. Introduction

## 1.1. Background

Locker systems are a set of compartments that are commonly used as storage containers in places such as schools and offices [1]. People use it to secure their belongings temporarily while they go other places to fulfil tasks. In 2021, the locker market is worth $1.3 billion globally and is expected to reach $1.7 billion by 2026 [2].

There are many types of authentication systems ranging from lock and key to mobile applications. Despite the lockers looking conspicuously secure, users may overlook the possibilities of locker theft. According to a recent news article by KUSA-TV, there were about 150 reports of locker theft from 2020 to 2022 in Colorado, US [3].

Studies have found that conventional locking methods have their own flaws. Users using lock and key method are prone to lock picking and can easily misplace keys [4] while those using PIN numbers or passwords are prone to making the lock less secure by using PIN/passwords that are easy to remember and related to their everyday lifestyle [5]. Using Bluetooth or mobile applications to unlock can be insecure as the devices can be hacked and allow unauthorised access to the hackers [6]. In addition, most of the elderly might find it hard to use mobile applications to unlock lockers and will require some form of education to use them. According to a recent study done in Singapore [7], technology adoption by elderly Singaporeans is on the rise but many may feel frustrated with the ability to learn and adapt to the digital environment.

Biometric methods are secure as the fingerprint [8] or the face [9] used to authenticate are unique [10]. It will also prevent users from forgetting their PIN numbers or lose the keys. Although there have been studies on developing and implementing biometric methods as locker systems, there are no specific attempts to create an improved prototype made with sustainable materials while keeping costs low.

## 1.2. Objective

The aim of this project is to build a locker system using fingerprint sensor and/or face recognition systems and to determine whether these authentication systems provide better security and less difficulty to unlock than conventional systems while keeping it low-cost and sustainable. An online-based market research survey will be conducted and be given to random surveyors to evaluate in terms of usability, difficulty, and sustainability. Results from the survey will be used to improve the capabilities of the locker system.

An Arduino UNO will be used as the mainboard to power the locker system by sending commands to the components of the system and process the fingerprint/face for authentication. Standard off-the-shelf components will be used for the locker system and fingerprint sensor. Machine learning will be used to create the face recognition system. Nearer to the completion of the project, the product will be improved based on feedback from the survey and ideas.

## 1.3. Project Workflow

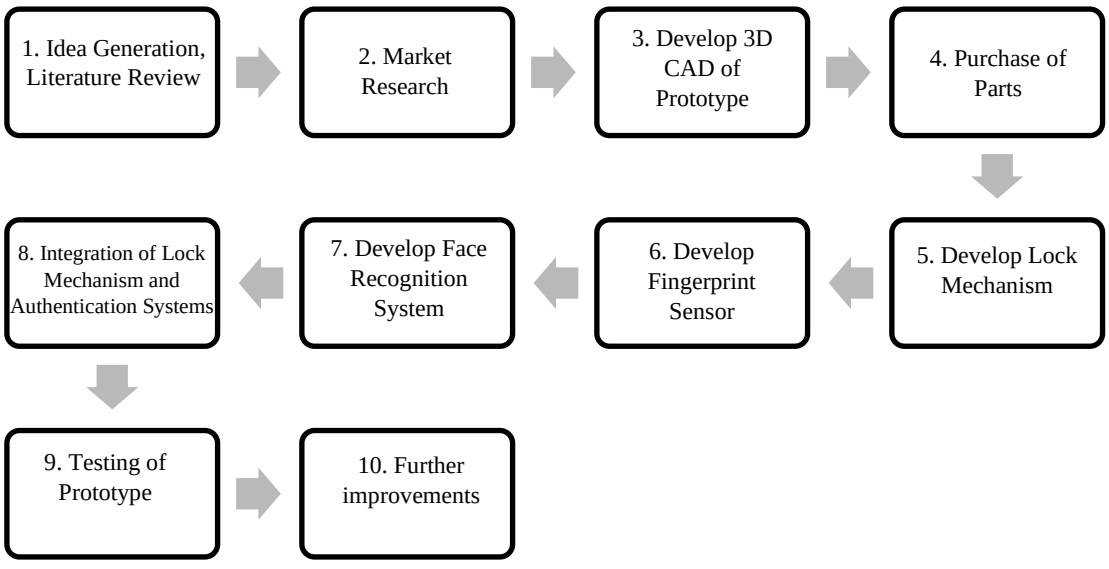



Figure 1. Project Workflow

## 1.4. Project Timeline

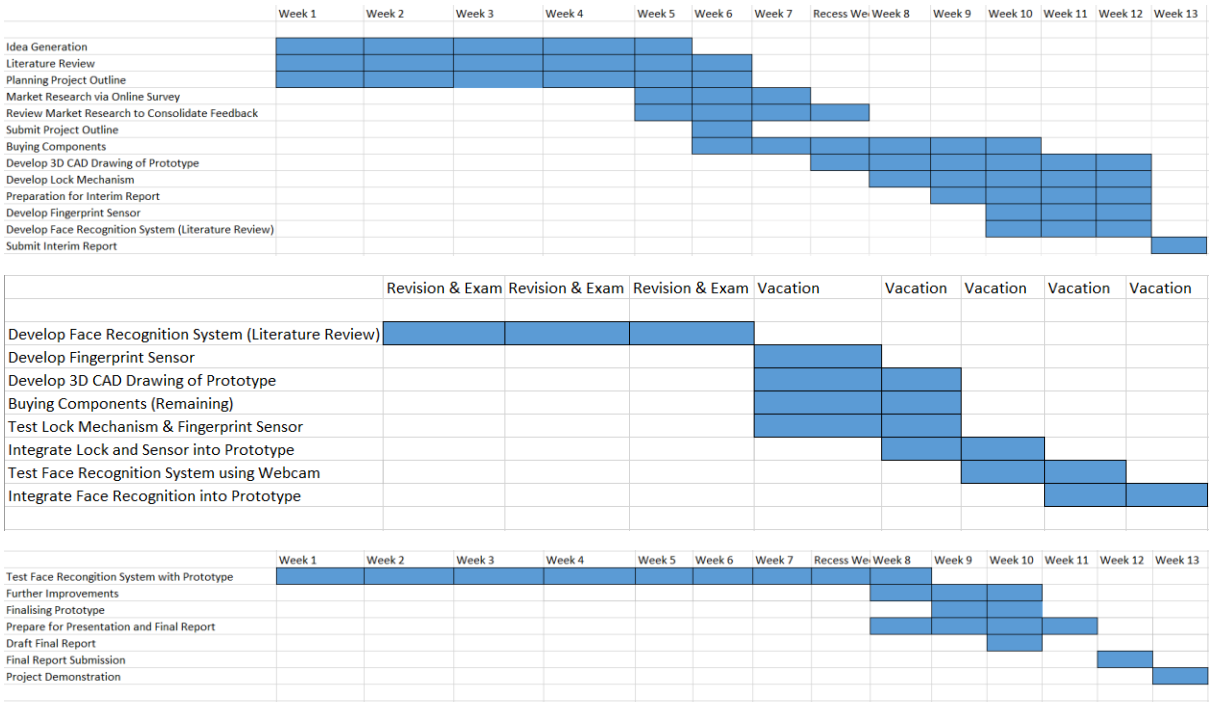



Figure 2. Project Timeline

The timeline is updated to reflect the shortcomings of the tasks that were given more time to complete.

# 2. Literature Review

## 2.1. Locker System

The prototype will be a box made of cardboard and has two doors for the locker system. It will showcase the authentication methods used to unlock the locker. The components that make up the locker system will be placed on the mainboard box on top of the main locker box. The lock will be used to lock and unlock the locker when the user is authenticated. The fingerprint sensor and ESP32-CAM will be placed outside the door to facilitate user to authenticate depending on the authentication system. There will be LCD displays to allow the user to view the state of the locker box and errors if the user fails to authenticate.

## 2.2. Fingerprint sensor

The fingerprint sensor is used for user authentication using the user's fingerprints. It can be easily bought online. There is a python library by Adafruit called Adafruit Fingerprint Sensor Library that can save and store fingerprints and is compatible with Arduino UNO. The fingerprint sensor can enroll, search and delete fingerprints of the users.

## 2.3. Face recognition system

The face recognition system is used for user authentication using the user's face. The ESP32-CAM is used to take a picture of the user's face and do inference on the faces to output result based on the similarity of the pictures taken. It can also be easily bought online like the fingerprint sensor.

There are various models to implement face recognition such as PCA (Principal Component Analysis) and One-Shot Learning. However, I will be implementing a version of face recognition developed by Espressif, the same developers of ESP32-CAM.

The face detection model is called MTMN which is a combination of MT-CNN (Multi-Task Cascaded Convolutional Networks) and MN (MobileNetV2). MobileNetV2 is a CNN architecture that is based on lightweight depth-separable convolution layers as a starting layer followed by inverted residual blocks with bottlenecking features [11].

This reduces the number of hidden layers compared to the original MobileNet, thereby reducing the model size. This advantage is suitable for deployment into mobile devices and microcontrollers.

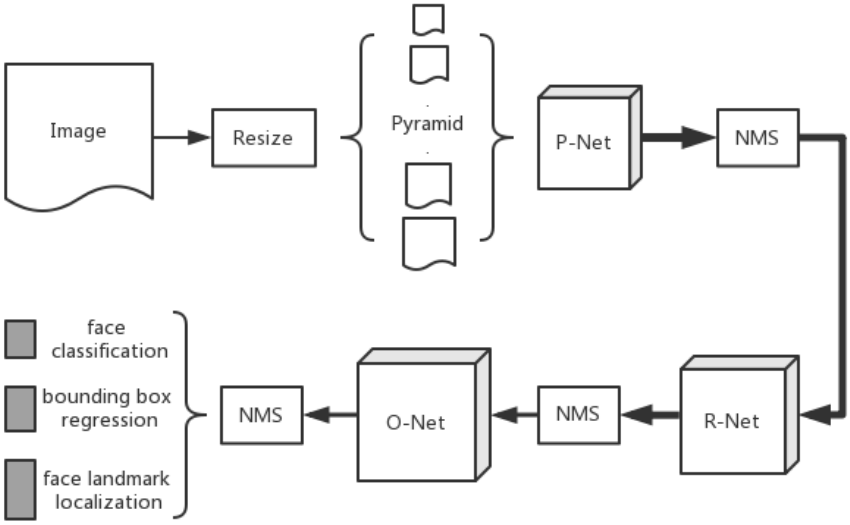



Figure 3. Face Detection Model

MT-CNN is a face detection and alignment algorithm that integrate both tasks using multi-task learning [12]. It is implemented using three stages of the algorithm, namely the Proposal Network (P-Net), the Refine Network (R-Net) and the Output Network (O-Net).

The P-Net is a fully convolutional network (FCN) model that aims to propose candidate windows and their bounding regression box vectors for face detection. It does not have a dense layer compared to other CNN models. Bounding box regression is required to detect the face and localise the area where the face is in the image. The proposed candidate windows and their bounding regression box vectors are sent to the R-Net.

The R-Net is a CNN model that checks the bounding regression box vectors from P-Net by reducing the number of candidates, calibrate bounding regression box vectors and implements non-maximum supression (NMS) to merge overlapping candidates. It outputs the classification if it’s a face or not, bounding box made of 4 element vector and facial landmark localisation made of 10 element vector.

The outputs from R-Net are then sent to the O-Net, where it outputs the results as bounding box, confidence coefficient and the 5 facial landmarks' positions.

The face recognition model is called FRMN which is a combination of the face recognition model FR (ArcFace) and MN (MobileNetV2). ArcFace or called Additive Angular Margin Loss is a loss function used commonly in face recognition tasks [13]. While Softmax is commonly used for face recognition, the ArcFace is better at optimising the feature embeddings to get a higher similarity score for intraclass images and more diversity for interclass images. It expresses the logits of each class as the cosine of the angle between the properties in a hypersphere of unit radius and the ground truth weights. The similarity score can be outputted using cosine similarity formula with inputs obtained from the logits.

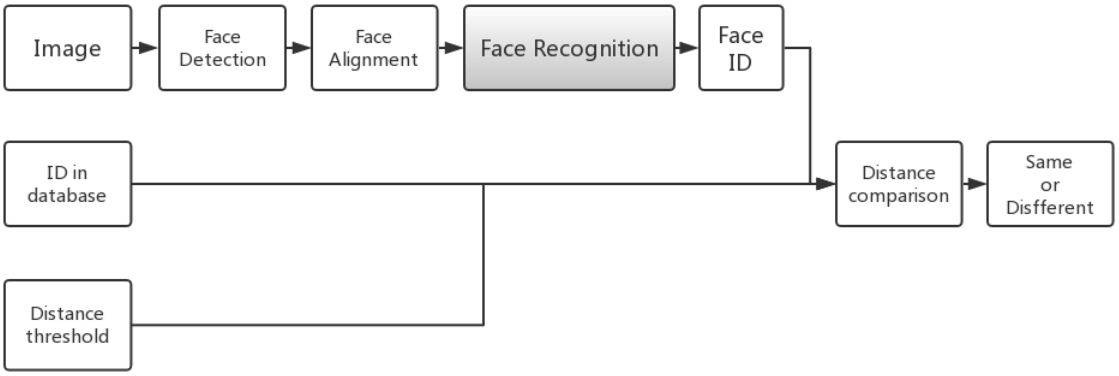



Figure 4. Overview of Face Detection and Face Recognition

# 3. Project Proposal

## 3.1. Proposal

The 4 major components for the locker system in consideration are the Mainboard, locker system mechanism, fingerprint sensor and camera vision.

**Component 1: Mainboard**

Arduino UNO is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins, 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an In Circuit Serial Programming (ICSP) header and a reset button. It is powered using a power supply or a battery. It will be used to connect with the servos in the locker system mechanism, fingerprint sensor, ESP32-CAM and other components.


Figure 5. Arduino UNO

**Component 2: Locker System Mechanism**

To make the project more sustainable, the locker system mechanism will be controlled by a servo instead of buying a physical lock. It is made up of 2 different parts – a 3D printed lock and a servo to drive the lock to unlock and lock. The servo will turn to unlock the lock when the user is authenticated and turn back to lock it when the user is done with using the locker at any time.

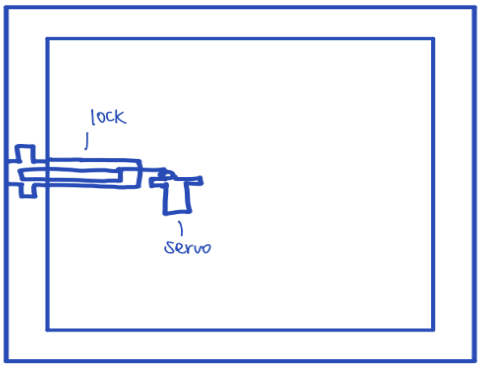



Figure 6. Locker System Mechanism

**Component 3: Fingerprint Sensor**

The fingerprint sensor is used to capture fingerprints and store them when the user uses the locker and returns for verification. It has a Digital Signal Processor (DSP) chip that does image rendering and fingerprint searching. It is connected via TTL serial and can store up to 162 fingerprints.

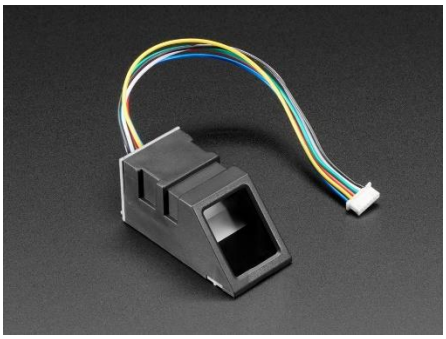


Figure 7. Fingerprint Sensor

**Component 4: Camera Vision**

The ESP32-CAM board will be used for face recognition. It uses OV2620 camera module and by FTDI programmer to be programmed using Arduino IDE. It will be used to capture users' faces and process it using machine learning to allow locker access to the user.


Figure 8. ESP32-CAM

# 4. Market Research

## 4.1. Market Research Survey

Market Research will be done via survey online. The purpose of market research is to study the demand of the locker systems using authentication systems I am proposing in the project. I want to find out if the users can find fingerprint sensor/face recognition systems beneficial instead of the conventional key/PIN number system in terms of ease of use, time spent to lock/unlock and other factors.

## 4.2. Market Research Responses

The responses are generally positive. All 25 responses agree that having fingerprint sensor/face recognition systems are beneficial. Most of the responses (19) prefer to use fingerprint sensor than other forms of authentication systems as shown in figure 10.

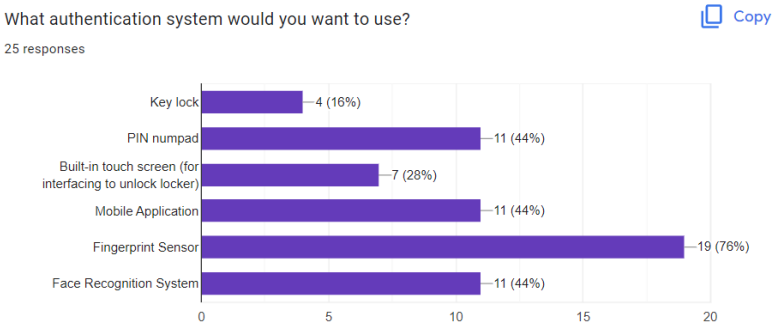



Figure 9. Responses regarding most preferred authentication system to use

In figure 11, respondents were asked to select in the scale of 1 to 5 on how easily do they lose locker keys or forget PIN numbers. The results were varied as most responses (11) who chose 1 or 2 do not lose keys or being forgetful easily while 9 responses chose 4 or 5 lose keys or forget numbers easily.

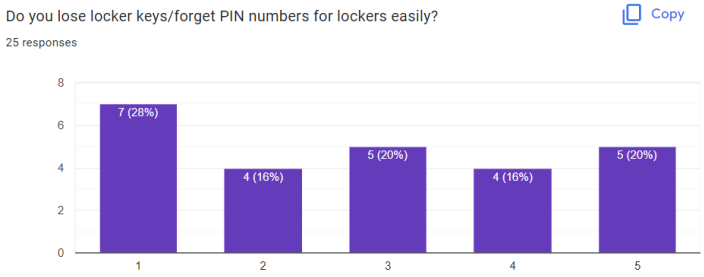



Figure 10. Responses regarding how respondents lose locker keys/forget PIN numbers easily

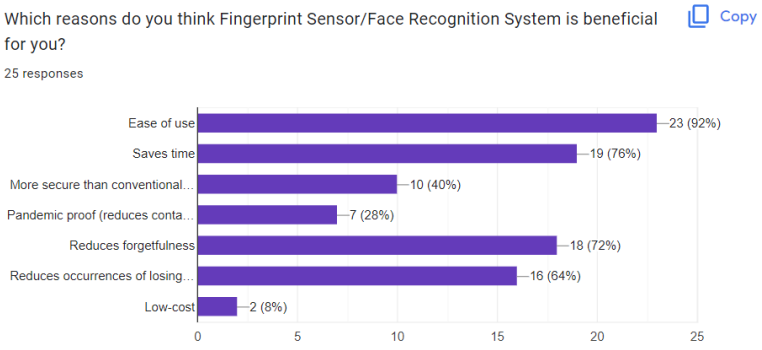



Figure 11. Responses regarding the reasons why Fingerprint sensor/Face Recognition System is beneficial

In figure 12, most of the respondents thought that the Fingerprint sensor/Face Recognition is easy to use (23), followed by saving time (19) and reduces forgetfulness (18).

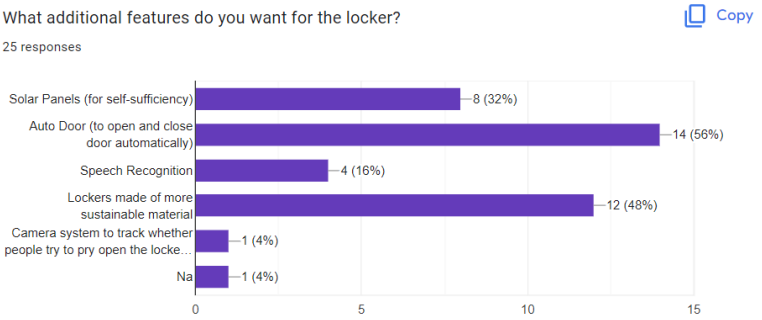



Figure 12. Responses regarding additional features

In figure 13, most of the respondents like to see an auto door or lockers made of sustainable material built into it. There is 1 response that wants a camera system to track unauthorised people attempting to pry open the locker.

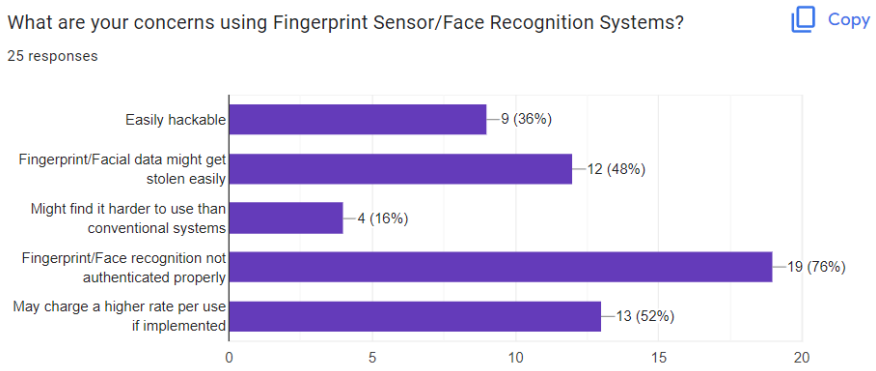



Figure 13. Responses regarding concerns using Fingerprint Sensor/Face Recognition Systems

In figure 14, most respondents were concerned about the Fingerprint Sensor/Face Recognition Systems not authenticated properly (19) followed by charging a higher rate per use (13) and fingerprint/facial data might get stolen easily (12).

# 5. Hardware

## 5.1. Buying Components

I have decided to buy the components using my own money instead of claiming from the school because the prices for each component are cheap and affordable.

The total prices for each component are as follows:

Table 1. Components List and its Prices

| Product | Cost | Qty | Total Cost |
|---|---|---|---|
| Fingerprint Sensor | $11 | 1 | $11 |
| ESP32-CAM | $10.11 | 1 | $10.11 |
| SG-90 Servo | $2.20 | 2 | $4.40 |
| LCD Display | $5.08 | 2 | $10.16 |
| Buttons | $4.71 | 1 | $4.71 |
| Magnetic Reed Switch | $1.29 | 2 | $2.58 |
| Self-Closing Door Hinge x2 | $13.67 | 1 | $13.67 |
| Power Barrel Jack for Arduino UNO | $7.45 | 1 | $7.45 |
| FTDI Programmer | $2.59 | 1 | $2.59 |
| 3D Printing Filament | $1.73 | 3 | $5.19 |
| Corrugated Cardboard | $0.85 | 2 | $1.70 |
| Soldering Helping Hands | $12.56 | 1 | $12.56 |
| Wires | $2.70 | 5 | $13.50 |
| USB to Mini USB Cable | $4.62 | 1 | $4.62 |
| Male Pin Header | $1.31 | 1 | $1.31 |
| Total Cost | | | $105.55 |

## 5.2. Body of the Prototype

### 5.2.1. Corrugated Cardboard

The prototype is a box made of cardboard or 3D printed material and has a door for the locker system. It will showcase the authentication methods used to unlock the locker. Behind the door, the components that make up the locker system will be placed. The fingerprint sensor/webcam will be placed outside the door to facilitate user to authenticate depending on the authentication system. There will be LED displays to allow the user to view the state of the locker box and errors if the user fails to authenticate.

Figure 14. Corrugated Cardboard

### 5.2.2. Locker Mechanism

The 3D printed parts for the lock mechanism have been printed and have been tested to ensure the respective parts work together. The sliding bolt bar is connected to the servo using a servo rod connector. The servo bracket will be placed beside the sliding bolt main body. The sliding bolt bar will be inserted into sliding bolt receiver to lock the locker door.

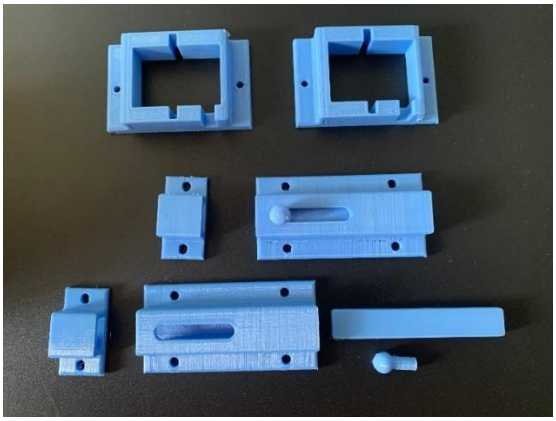


Figure 15. 3D Printed Parts of Locker Mechanism

### 5.2.3. SG-90 Servo

The SG-90 servos are used to lock and unlock the locker doors. They are connected by 5V VCC, ground and digital pins.


Figure 16. SG-90 Servo

### 5.2.4. Self-Closing Door Hinge

The door hinge has a self-closing mechanism that allows the door to be closed on its own when the user chooses to stop using effort to open the door. The door hinge will allow the user to use minimum effort to open the door. It is made of stainless steel to prevent any user from tempering with it.

Figure 17. Self-Closing Door Hinge

### 5.2.5. Magnetic Reed Switches

The magnetic reed switches are used to tell if the user has closed the locker door after each use. This allows the mainboard to interact with the servos to close the doors. They are connected via 5V VCC, ground and digital pins.

The wires of magnetic reed switches are soldered into male 1-pin headers to be plugged into the breadboard as they are initially thin wires and are able to come out of the breadboard easily.

Figure 18. Magnetic Reed Switches

### 5.2.6. Buttons

The buttons allow the user to interact with the locker system. There are 2 buttons that the user can choose to press according to the scenarios presented on the LED displays. It is connected via 5V VCC, ground and digital pins.

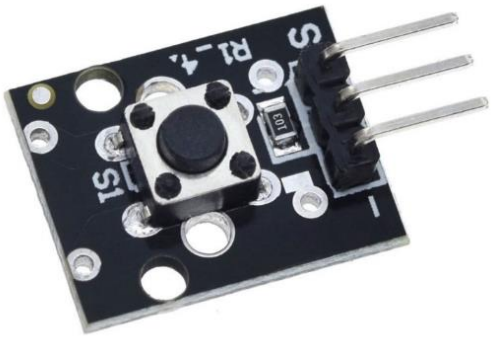


Figure 19. Button

## 5.3. Mainboard Box of the Prototype

### 5.3.1. Fingerprint Sensor

The fingerprint sensor uses the Adafruit Fingerprint Sensor Arduino library to operate and has 2 functions – enrolling and searching. Enrolling assigns new fingerprints to an ID to be queried later and searching identifies the fingerprint and retrieves the ID of the fingerprint to be used for various purposes, in this case unlocking the locker door.

The wires given had an end-to-end connector. To connect to Arduino UNO, the wires of one end of the small connector had to be cut and strip off to connect to the mainboard. After I strip off the wires, it is shown to have made up of thin wires instead of 1 singular wire. In order to keep them in place, soldering is used to keep the singular wires in place and soldered to a male 4-pin header, allowing the wires to be plugged into the breadboard with ease.

Figure 20. Fingerprint Sensor

### 5.3.2. ESP32-CAM

The ESP32-CAM is used for face recognition has a built-in OV2640 camera to take pictures. It uses Espressif's header files and functions to capture images and recognise faces using its own deep learning library. The face recognition model used can enroll, search and delete faces.

A FTDI programmer is used to connect ESP32-CAM. It connects with 5V voltage, ground, TX and RX pins. The TX and RX pins are used to compile code from Arduino IDE to ESP32-CAM. After it has been compiled, the code starts up after pressing the reset button of the microcontroller. The 5V voltage and ground can still power ESP32-CAM without the need for TX and RX pins to be connected as those pins can be used for other purposes like SoftwareSerial with Arduino UNO.


Figure 21. ESP32-CAM

### 5.3.3. LCD Display

The LCD display is used to output messages from Arduino UNO and ESP32-CAM. It prompts the user to press buttons and output important messages like errors and whether the fingerprint/face is registered. It is connected using SCL, SDA, 5V voltage and ground. This LCD display has a controller that makes use of I2C and reduces the number of pins used to connect to Arduino UNO and ESP32-CAM, saving pins for other uses.

The LCD display in Arduino UNO uses the SDA and SCL pins for communication. However, the ESP32-CAM does not have actual SDA and SCL pins for the LCD display to use. The Wire.h library is used to assign pins to empty pins that are not used

by ESP32-CAM. Pins 15 and 14 are used for ESP32-CAM to interact with the LCD display.

Figure 22. LCD Display

## 5.4. Miscellaneous

### 5.4.1. Power barrel jack for UNO

The power barrel jack for UNO is used to power the Arduino UNO when it is operating standalone without a laptop to power and program the microcontroller. This is important for project demonstration which the locker system is expected to operate on its own without a laptop prompting the system to operate. The power output is 9V 2A to allow Arduino UNO to be powered safely without oversupply of voltage.


Figure 23. Power Barrel Jack

### 5.4.2. FTDI Programmer

The FTDI programmer is used to power the ESP32-CAM and to program the microcontroller using a laptop. It has 4 pins – 5V VCC, ground, TX and RX for programming ESP32-CAM.

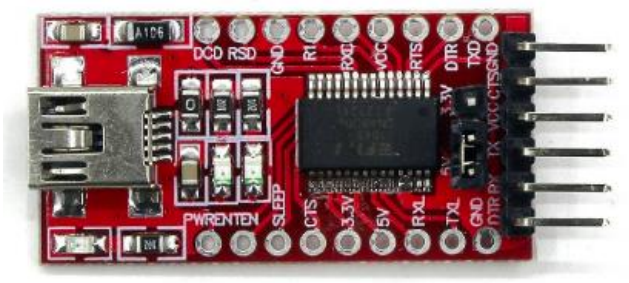



Figure 24. FTDI Programmer

### 5.4.3. 3D Printing Filament

The 3D printing filament is used to print the lock mechanism for the locker doors.

Figure 25. 3D Printing Filament

### 5.4.4. Soldering Helping Hands

The helping hands are used to help facilitate the testing of ESP32-CAM for face recognition system. It is also used to help facilitate soldering for fingerprint sensor and magnetic reed switches.

Figure 26. Soldering Helping Hands

### 5.5.5. Wires

The wires are used for wiring in the circuits of Arduino UNO and face recognition system.

Figure 27. Wires

### 5.5.6. USB to Mini USB Cable

The USB to mini USB cable is used to connect the FTDI programmer to the laptop for debugging and code compiling purposes. It can also be plugged to the power plug for external power source without the laptop programming it.


Figure 28. USB to Mini USB Cable

### 5.5.7. Male Pin Header

The male pin header is used for soldering into the wires of fingerprint sensor and magnetic reed switches.

Figure 29. Male Pin Header

# 6. Software

## 6.1. Arduino UNO

### 6.1.1. Arduino Libraries

The programming for Arduino UNO is done using Arduino IDE 1.0 as Arduino IDE 2.0 may have compatibility problems with the Arduino libraries. Arduino libraries are used for additional functionality to interact with the Arduino UNO. It can come from the library manager in the IDE or download from online sources.

The first line is to initialise the fingerprint sensor functions for enrolling, searching and deleting fingerprint. The second line is to initialise the Software Serial library to communicate serially with ESP32-CAM. The third line is the string.h library to parse strings to ESP32-CAM. The Servo.h function initialises the functions to control the servo. The last line initialises the LCD display library to control the LCD display.

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
#include "string.h"

#include <Servo.h>
#include <LiquidCrystal_I2C.h>
```

Figure 30. Arduino Libraries

### 6.1.2. Fingerprint Sensor

The fingerprint sensor is powered by the Adafruit Fingerprint Sensor Library, written by Limor Fried (Ladyada) from Adafruit Industries. The fingerprint sensor library is used to interact with the fingerprint sensor by enabling enroll, search and delete functions.

The locker system will prompt the user to register the fingerprint via enrolling by placing the finger onto the fingerprint sensor if he/she uses the locker for the first time. This fingerprint will be registered in a specific ID. When the user comes back to unlock the locker, he/she will be prompted to place the finger onto the fingerprint sensor to search if the fingerprint is the same as one of the IDs stored in the database. If it matches, the locker will unlock for the user to use. In the event the user wants to stop

using the locker, he/she can choose the option to delete the fingerprint, which allows the library to delete the fingerprint and its ID from its database. This prevents any data leakage on old fingerprint data should an attempted hack occur.

The function getFingerprintEnroll() enrolls the user. First, it gets the first fingerprint from the user. Then the user is prompted to remove the finger momentarily and the function gets the second fingerprint from the user. The fingerprint sensor library will compare both fingerprints and outputs its result to the user. If both fingerprints match, the locker door is unlocked. If both fingerprints do not match, the user is prompted to scan his/her fingerprints again.

The function getFingerprintID() searches the user's fingerprints. The user is required to scan his/her fingerprint only once to search for the user ID. If the fingerprint matches to the user ID of the locker desired for unlocking, the locker door is unlocked. If it does not match, the user will have to rescan his/her fingerprint to unlock the door. If the user decides to delete his/her fingerprint, the function deleteFingerprint() is used to delete the fingerprint of the user.

### 6.1.3. LCD Display

The LCD display is used to display important information about the locker system and prompts the user to do something for the next course of action to be done. The LCD display's library interacts with the LCD display via I2C communication protocol. This allows SDA and SCL pins to be used to interact with the LCD display instead of more pins used normally for an LCD display without a I2C module.

The LCD display does not have the ability to refresh letters on specific places if the next word output is shorter than the previous word. Using lcd.clear() allows the LCD display to clear all the output from the display and new word output can be used to display new messages.

```
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("locker 1 or 2?");
```

Figure 31. LCD Display Code

### 6.1.4. Buttons

The buttons are used to allow the user to interact with the locker system. The user can interact with 2 buttons while the 3rd button is used for debug purposes. The 3rd button allows the serial communication to be debugged if the serial data received is the wrong message.

```
while ((digitalRead(buttonApin) == HIGH) and (digitalRead(buttonBpin) == HIGH));
if (digitalRead(buttonApin) == LOW) {
  //Serial.println("Selected fingerprint!");
  finger_face = 1;
}
else if (digitalRead(buttonBpin) == LOW) {
  //Serial.println("Selected face!");
  finger_face = 2;
}
```

Figure 32. Button Code

### 6.1.5. Servos

The servos are used for opening and closing locker doors. The servos are set at 0 degrees first to lock the doors. When the code to tell the servo to open the locker door passes, the servos will turn at 180 degrees. After the door is closed, the magnetic reed switches will inform the servo that the door is closed and the servo will turn back at 0 degrees to lock the door.

### 6.1.6. Magnetic Reed Switches

The reed switch is used to check if the door has been closed or not. It has 2 inputs – Digital pin and ground pin. The code is similar to the button code where it goes into a while loop when the switch does not detect the magnet. When it detects the magnet, the while loop is exited and will tell Arduino UNO that the door has been closed.

```
while (digitalRead(switch1) == HIGH);
if (digitalRead(switch1) == LOW) {
  Serial.println(F("Closing locker 1"));
}
```

Figure 33. Magnetic Reed Switch Code

### 6.1.7. Software Serial

Software Serial is used for serial communication between the Arduino UNO and ESP32-CAM. When the user wants to use the face recognition system for enrolling

and searching functions, the Arduino UNO sends a message to ESP32-CAM using SoftSerial.write() to do face recognition for enrolling, searching and deleting faces. After the ESP32-CAM finish doing its job, it will send another message to the Arduino UNO to tell that it is done. The messages are then read using SoftSerial.readStringUntil(). Then the Arduino UNO will take its next course of action depending on the success of the face being recognised.

```
SoftSerial.listen();

//face recognition system
if (id == 1) {
  SoftSerial.write("aaa");
}
else if (id == 2) {
  SoftSerial.write("ccc");
}
```

Figure 34. Software Serial Write Code

```
while (!done) {
  Serial.println(F("Waiting"));
  delay(50);
  if (SoftSerial.available()) {
    terminalText = SoftSerial.readStringUntil('\n');
    Serial.println(F("SoftSerial in"));
    Serial.println(terminalText);
    delay(1000);
    if ((terminalText.indexOf("b") != -1) or (terminalText.indexOf("d") != -1)) {
      Serial.println(F("Image is True"));
      if (terminalText.indexOf("b") != -1) {
        facelist1 = 1;
        Serial.println(F("facelist1 used"));
      }
      else if (terminalText.indexOf("d") != -1) {
        facelist2 = 1;
        Serial.println(F("facelist2 used"));
      }
      done = true;
```

Figure 35. Software Serial Receive Code

### 6.1.8. Main Code

The main code implements the overall locker system and fingerprint sensor. It also sends serial data to ESP32-CAM if the user wants to use the face recognition system. The user will be greeted by the main menu to use the locker system either by enrolling

as a new user or search for fingerprint or face as an old user. The user then can interact with the locker system using buttons.

If the user chooses the enrolment function, the user can choose to use either the fingerprint sensor or face recognition system, or both systems. If the user chooses the first option, he/she can choose either the fingerprint sensor or face recognition system and the Arduino UNO will remember the user's choice. The user then chooses which locker to use. After that, the user can start to enroll using his/her fingerprint or face, or both. If the locker is in use by another user, an error will be outputted by the LCD display and the locker system will go back to the main menu. After the user has finished enrolling according to its choice of authentication system, the locker door will open according to the desired locker the user wants to use. Once the user has finished placing his/her belongings into the locker, the user will manually close the locker door to be locked and complete the enrolment process.

If the user chooses the searching function, the user can choose to unlock the desired locker first. If the particular locker has both fingerprint and face data, it will prompt the user to choose either fingerprint sensor or face recognition system to unlock the locker. Next, the user can choose to continue using the locker or not. If the user does not continue to use the locker, the fingerprint or face data will be deleted right after authentication and before locker is open. Once the user is authenticated, the locker door will open and the user will manually close the locker door to be locked and complete the searching process.

The third function used for debugging purposes is to send serial data to ESP32-CAM to calibrate the serial communication and clean out any stray strings before the user can use the locker system.

## 6.2. ESP32-CAM

### 6.2.1. ESP32 Libraries

The programming for ESP32-CAM is done using Arduino IDE 1.0 as Arduino IDE 2.0 may have compatibility problems with interacting with ESP related systems and libraries. Like the Arduino UNO, ESP32 libraries are used for additional functionality to interact with the ESP32-CAM, which can come from the library manager in the IDE or download from online sources.

The first 4 lines are for initialising the ESP32-CAM camera, face detection and face recognition functions. The next 2 lines are for disabling brownout problems. Whenever the ESP32-CAM suffers from drawing low voltages (< 3.3V), "brownout detector has been triggered" problem occurs. To prevent this, these 2 libraries are necessary to initialise to prevent unforeseen resets from happening. The last 2 lines are for the LCD display. Unlike the LCD display code in Arduino UNO, there is a Wire.h library initialised. ESP32-CAM does not have dedicated pins for I2C as the camera and SD card occupies multiple pins. The Wire.h library can set predefined pins for the LCD display to use, in this case pins 14 and 15. The pins can be used as long as ESP32-CAM is not using them for other purposes.

```
#include "esp_camera.h"
#include "fd_forward.h" //works with esp32 board library 1.0.5
#include "fr_forward.h"
#include "fr_flash.h"

#include "soc/soc.h"           // Disable brownout problems
#include "soc/rtc_cntl_reg.h"  // Disable brownout problems

#include <Wire.h>
#include <LiquidCrystal_I2C.h>
```

Figure 36. ESP32-CAM Libraries

### 6.2.2. Face Recognition System

The face recognition system is powered by the ESP32-CAM microcontroller with the built-in camera and uses the face recognition code from Espressif's ESP-DL library. The face recognition system is made up of two systems – the face detection model and the face recognition model.

MTMN first resizes the image to 56x56 input shape to be detected by the model. It goes through the MT-CNN model to get the bounding box for the face in the image. Finally, it goes through face alignment to crop out the bounding box of the image and obtain the result.

FRMN first obtains the input image from the output of MTMN model. Then it obtains the face ID uniquely assigned to the input image after implementing the face recognition model to it. This is called face enrolling.

Face searching is implemented by taking another picture of the same face, obtained the face using face detection and alignment and assigned the face ID to the input face image. Then it uses cosine distance to measure the similarity score between the first face and the second face.

### 6.2.3. LCD Display

Like the Arduino UNO, the LCD display is used to display important information about the face recognition system and prompts the user to do something for the next course of action to be done. The LCD display's library interacts with the LCD display via I2C communication protocol. The SDA and SCL pins are in pins 15 and 14 as ESP32-CAM does not have actual SDA and SCL pins and have to use Wire.h library to assign pins to empty pins that are not used by ESP32-CAM.

The LCD display does not have the ability to refresh letters on specific places if the next word output is shorter than the previous word. Using lcd.delete() allows the LCD display to clear all the output from the display and new word output can be used to display new messages.

```
lcd.setCursor(0, 0);
lcd.print("list 2 deleted");
Serial.write("lll"); //True
```

Figure 37. LCD Display Code

### 6.2.4. Software Serial

Software Serial is used to receive messages from Arduino UNO and ESP32-CAM will take its next course of action. After completing its actions, ESP32-CAM will send a message back to Arduino UNO to tell that it is done. This is used when the user wants to use the face recognition system for enrolling and searching functions. The Arduino UNO will receive the message from ESP32-CAM and take its next course of action.

```
if (cap_num == 1) {
  while ( delete_face(&id_list1) > 0 ) {
    lcd.setCursor(0, 0);
    lcd.print("Deleting Face for list 1");
  }
  lcd.setCursor(0, 0);
  lcd.print("list 1 deleted");
  Serial.write("jjj"); //True
}
else{
  Serial.write("ooo"); //False
}
```

Figure 38. Software Serial Code

### 6.2.5. Main Code

The main code implements the face recognition system based on the serial data received from Arduino UNO. When it receives the specific string from serial data, the ESP32-CAM will carry out the tasks according to the string received. The ESP32-CAM will carry out the tasks based on 6 strings it receives from Arduino UNO.

The first two tasks – enrollone and enrolltwo, enrolls the user's face into the face recognition system. One and Two refers to the locker number respectively. The face recognition system will take 5 pictures of the user's face and store it as a face ID for the desired locker to be unlocked. If the face is not aligned properly, the user will have his/her face taken again. This process is automatic and will require the user's face to be right in front of the ESP32-CAM camera. A string will be sent to Arduino UNO to inform that the enrolment process is done.

The next two tasks – searchone and searchtwo, searches the user's face of the desired locker to be unlocked. The face recognition system will take the picture of the user and compare it to the picture taken during enrolment process for 100 seconds. If it does not match, the user will be required to have another picture taken again. If it matches, a string will be sent to Arduino UNO to inform that the searching process is done and the user's faces matches. If time runs out, a string will be sent to Arduino UNO to inform that the searching process has failed and the user's faces does not match.

The last two tasks – searchanddeleteone and searchanddeletetwo, does the same tasks as searchone and searchtwo but adds the deleting function into it. Once the faces have been matched, the delete function will be used to delete the faces from the system. A

string will be sent to Arduino UNO to inform that both searching and deleting processes are done.

# 7. Project Development

## 7.1. Assembly of Locker Prototype

### 7.1.1. 3D CAD of Prototype

The 3D CAD of prototype is made using the FreeCAD software. The dimensions of the prototype are 357 mm x 191 mm x 230 mm for the main locker box. There are two cut outs that form 2 locker rooms.

On top of the locker rooms' box, it will have a mainboard box that contains the mainboard, fingerprint sensor, ESP32-CAM and LCD screen. I placed all the main components in this box to make it easier to connect one another as the wires used are short and will require them to be placed together. The mainboard box sits on top of the main locker box cover.

The doors contain the lock mechanism and door hinges. A wall in the middle of the locker prototype separates the two lockers.

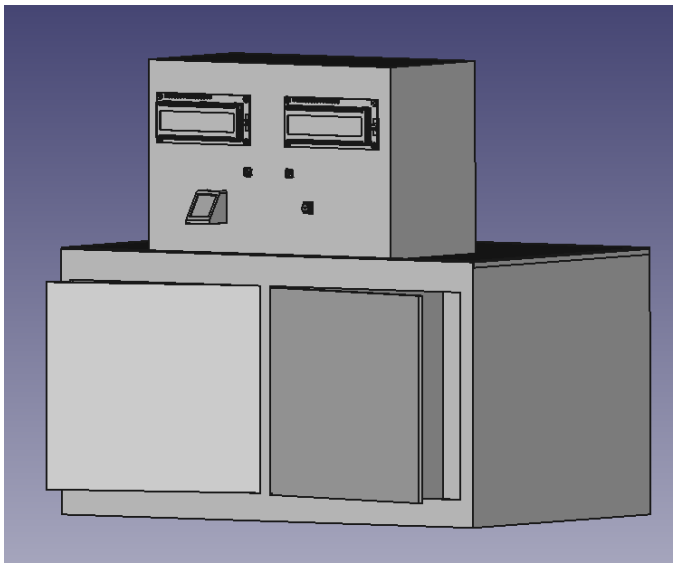


Figure 39. 3D CAD Prototype

### 7.1.2 Physical Prototype

The prototype is made of double-sided corrugated cardboard to make sure the prototype is stable. I considered to use 3D printing to print the prototype but decided to use cardboard as it is cheaper and is more sustainable. I can also easily make edits to the prototype with leftover cardboard. The flaps of the top side of the main locker box and top and bottom side of mainboard box are cut off to integrate various boxes easily.

Figure 40. Physical Prototype with Doors Closed

Figure 41. Physical Prototype with Door Open

The locker system is separated into two locker rooms. Each locker room has a door that automatically open by the door hinge and unlocked/locked by the lock mechanism containing 3D printed parts and servo. The 3D printed parts are the sliding bolt main body, sliding bolt bar, sliding bolt receiver and servo bracket. The sliding bolt bar is connected to a servo rod connector linked to the servo.

The mainboard box contains all the components needed to run the locker system. There are holes for the fingerprint sensor, ESP32-CAM, 2 LCD displays and 2 buttons for the user to interact with. There are also holes for Arduino UNO and FTDI programmer to be plugged to power source.

## 7.2. Assembly of Fingerprint Sensor and Face Recognition Systems

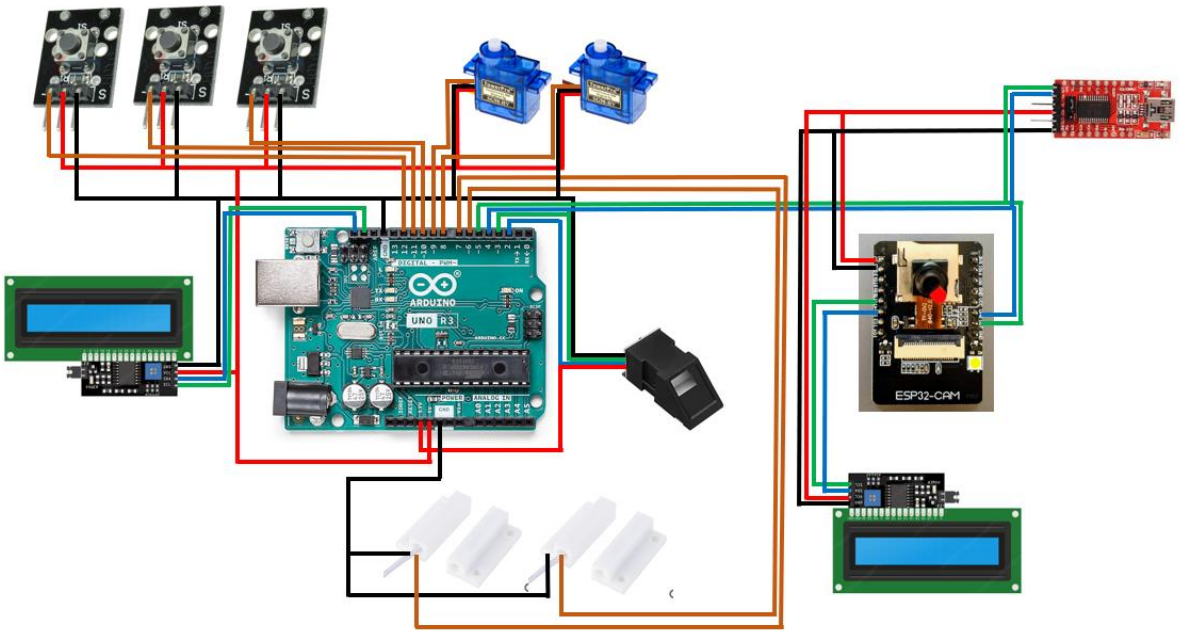



Figure 42. Wiring Diagram

The Arduino UNO controls the fingerprint sensor, 2 buttons, LCD display, magnetic reed switches and servos. It also controls the Software Serial via software UART using digital pins to ESP32-CAM. The LCD display is connected via I2C pins (SDA and SCL). The rest of the components are connected with digital pins. Pins 0 and 1 are left untouched for serial monitor.

The ESP32-CAM controls the LCD display to output messages related to face recognition system only. It is also connected to Software Serial pins using UART communication pins to Arduino UNO.

# 8. Problems Encountered

## 8.1. Locker System Prototype

Initially, I thought about using 3D printing to print out the locker prototype. However, it needs a large amount of filament for 3D printing. Realistically, it will be too expensive and take up a long time to print out a box suitable for project demonstration. Therefore, I bought a big cardboard box and cut out holes for locker doors. I also bought smaller boxes for mainboard enclosure and other parts that require some cardboard to be fitted in the locker prototype.

## 8.2. SG-90 Servo

The SG-90 Servo is very fragile and can break if not in use for a long time. It is also prone to bad QC (Quality Control), which is why the servo is very cheap. 1 of SG-90 Servo that I bought did not work and overheated when connected to Arduino UNO. The gears also did not move and vibrates constantly. I bought a few more spares afterwards to replace the broken servo and to replace servos if they do break in the future.

## 8.3. ESP32-CAM Connector

When I bought ESP32-CAM, it came with a connector called ESP32-CAM-MB connector. Initially, it was troublesome to troubleshoot the ESP32 code as I must remove the ESP32-CAM from breadboard to ES32-CAM-MB connector every time to compile new code and mount back to breadboard. I solved the problem by buying a FTDI programmer to program ESP32-CAM. I do not need to constantly remove the microcontroller from the FTDI programmer as I can just plug in necessary pins into it. Additionally, it also acts as power supply for ESP32-CAM. I can plug the USB cable into a mobile phone charger and it will work fine without debugging.

## 8.4. FTDI Programmer

After I unplugged the TX and RX connectors from FTDI programmer to ESP32-CAM and connect another TX and RX connectors meant for serial communication to Arduino UNO, the serial monitor stop outputting any messages. I managed to solve the problem by connecting the LCD display to ESP32-CAM to see the output.

### 8.5. Face Recognition System – Espressif

After getting the output for the image of aligned face, I needed to confirm whether the code works. I decided to add a SD card into ESP32-CAM and write the images to the SD card. Afterwards, I remove the SD card from the microcontroller and put into my laptop to check the images. The images were fine according to the output I want.

A problem I encountered was compiling the code for the face recognition system into ESP-32 CAM. There were a few errors about 'undefined reference' of several functions used in the code. After searching the internet for solutions, I found out that the ESP32 board library installed is a newer version and will not work for certain functions. I decided to use an older version of the library and the errors were resolved.

### 8.6. Fingerprint Sensor Serial Communication

I wanted to use a separate serial communication library for the fingerprint sensor as I thought that the serial communication with ESP32-CAM may interrupt the fingerprint sensor from working. I used another SoftwareSerial (AltSoftSerial) library and used pins 7 and 8, but during testing the implementation did not work. I decided to change back to SoftwareSerial but the fingerprint sensor couldn't work anymore, so I bought a new fingerprint sensor.

I still do not know why the fingerprint sensor stopped working, despite checking the wire connections and whether I gave it a high voltage (5V instead of 3.3V). I suspect that due to the pins 7 and 8 having a hardware timer, it may have disrupted the fingerprint sensor's internal timer. My fears that fingerprint sensor might get interrupted by the serial communication were unfounded after using SoftwareSerial's Two Port Receive, where I use port.listen() whenever I need to switch to either the fingerprint sensor or serial communication with ESP32-CAM.

### 8.7. Fingerprint Sensor and Magnetic Reed Switch wires

The connector wires of the older fingerprint sensor needed to be cut off then strip off the wires. After stripping off, it is made of thin wires instead of 1 singular wire. I soldered them to keep it in place and solder it together with pin headers to be connected via the breadboard.

The newer fingerprint sensor I bought has the wires already stripped off at one end of the wires. They are made of 1 singular wire instead of multiple thin wires. I soldered them to the pin headers to be connected to the breadboard.

After getting the magnetic reed switches, I realised that the wires are thin and will fall off the breadboard when I test them. Like the fingerprint sensors, I soldered them to the pin headers to be connected to the breadboard.

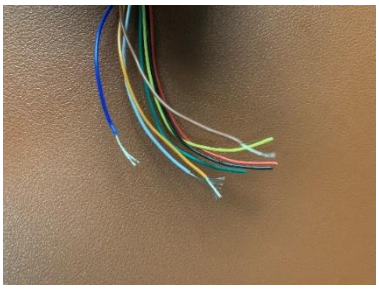


Figure 43. Fingerprint Sensor Wires

## 8.8. Face Recognition System – TensorFlow Lite

The problems detailed in this sub-section are about using Siamese networks via one-shot learning for face recognition [14]. Due to various problems with the implementation, I have decided to change to Espressif's implementation of the face recognition system detailed in section 6.

Instead of gathering a dataset of the users' faces and having to train them, I could use a small dataset of random samples of each class to train the model to classify faces accurately and prepare it to classify future unknown samples. Then I test it with a new sample by saving it as a new class and have it compared with a new sample of the same class. If both samples compared gives a high accuracy, it is deemed to be the same.

Triplet loss would also be applied by comparing an anchor sample and a positive sample of the same class, and a negative sample of a different class. This approach minimises the distance between the anchor and positive samples of the same class and maximises the distance between the anchor and negative samples of different classes.

This results in feature embedding, where similar faces produce embeddings that are small and allows for more accurate identification and verification of samples.

To do this, I will use ESP32-CAM to take pictures of the user which will be connected to the Arduino UNO mainboard. Then, I will download and use the ESP32 library code to transmit and receive data from ESP32-CAM. Python code will used to house the face recognition code as I will be using OpenCV library to implement the system. The CNN model I will be using is called FaceNet, which is used for face recognition using pre-trained weights.

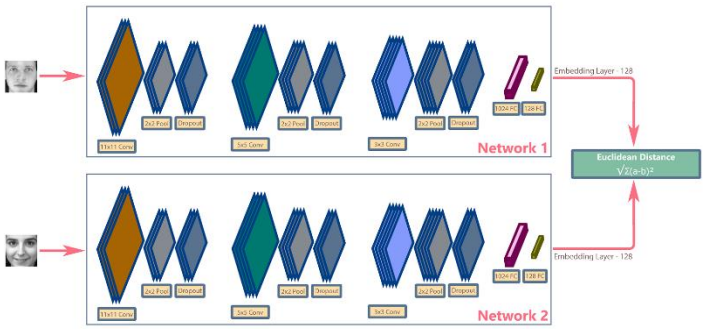



Figure 44. Siamese Network

My plan was to create a Siamese network model in Jupyter Notebook to be used for face recognition, train with a training dataset and test with a test dataset to be converted into a TensorFlow Lite model for deployment into ESP32-CAM. First I import the necessary libraries and get the dataset that I had already downloaded from the internet. The dataset contains the images of faces that are pre-aligned. I separated the images into training and test datasets.

Next, I created the model using transfer learning based on Inception V3 model as transfer learning can give the model a higher accuracy by using pre-trained weights and hidden layers. I also compiled the model with the loss function and Adam optimiser. Then, I train the model with 25 epochs to decrease the loss as much as possible and increase the accuracy with every epoch.

```python
class DistanceLayer(layers.Layer):
    # A layer to compute ‖f(A) - f(P)‖² and ‖f(A) - f(N)‖²
    def __init__(self, **kwargs):
        super().__init__(**kwargs)

    def call(self, anchor, positive, negative):
        ap_distance = tf.reduce_sum(tf.square(anchor - positive), -1)
        an_distance = tf.reduce_sum(tf.square(anchor - negative), -1)
        return (ap_distance, an_distance)


def get_siamese_network(input_shape = (128, 128, 3)):
    encoder = get_encoder(input_shape)

    # Input Layers for the images
    anchor_input   = layers.Input(input_shape, name="Anchor_Input")
    positive_input = layers.Input(input_shape, name="Positive_Input")
    negative_input = layers.Input(input_shape, name="Negative_Input")

    ## Generate the encodings (feature vectors) for the images
    encoded_a = encoder(anchor_input)
    encoded_p = encoder(positive_input)
    encoded_n = encoder(negative_input)

    # A Layer to compute ‖f(A) - f(P)‖² and ‖f(A) - f(N)‖²
    distances = DistanceLayer()(
        encoder(anchor_input),
        encoder(positive_input),
        encoder(negative_input)
    )

    # Creating the Model
    siamese_network = Model(
        inputs  = [anchor_input, positive_input, negative_input],
        outputs = distances,
        name = "Siamese_Network"
    )
    return siamese_network

siamese_network = get_siamese_network()
siamese_network.summary()
```

Figure 45. Siamese Network Code

After training the model, I extracted the encoder of the model to be used for prediction and test it with the test dataset and get the test accuracy to ensure the model does not overfit. If the model overfits, the test accuracy will be lower than training accuracy which is not the model I want to deploy.

```python
def extract_encoder(model):
    encoder = get_encoder((128, 128, 3))
    i=0
    for e_layer in model.layers[0].layers[3].layers:
        layer_weight = e_layer.get_weights()
        encoder.layers[i].set_weights(layer_weight)
        i+=1
    return encoder

encoder = extract_encoder(siamese_model)
encoder.save_weights("encoder")
encoder.summary()
```

Figure 46. Code to Extract Encoder

The model is then converted into a TensorFlow Lite model. I use post-integer quantisation to reduce the model further by converting float calculations to integer.

```
def representative_data_gen():
    for input_value in tf.data.Dataset.from_tensor_slices(train_images).batch(1).take(100):
        yield [input_value]


# Convert the model.
converter = tf.lite.TFLiteConverter.from_keras_model(encoder)
converter.optimizations = [tf.lite.Optimize.DEFAULT]
converter.representative_dataset = representative_data_gen
# Ensure that if any ops can't be quantized, the converter throws an error
converter.target_spec.supported_ops = [tf.lite.OpsSet.TFLITE_BUILTINS_INT8]
# Set the input and output tensors to uint8 (APIs added in r2.3)
converter.inference_input_type = tf.uint8 #tf_uint8, int8 to be more optimized
converter.inference_output_type = tf.uint8 #tf_uint8

tflite_model_quant = converter.convert()

# Save the model.
with open('model_quant.tflite', 'wb') as f:
    f.write(tflite_model_quant)
```

Figure 47. Post-integer Quantisation and TFLite Conversion Code

However, there was a problem encountered with deploying the model. After doing post-integer quantisation, I found the model to be too large to be deployed into ESP32-CAM's meagre 4MB of PSRAM. The model size was 17MB which was too big for the Arduino IDE to compile. The complexity of the Inception V3 model along with the Siamese network having 3 classes to classify in the model meant the model would have been too big to be deployed to microcontrollers compared to the more simple models provided by TensorFlow. I tried to use pruning and clustering methods to reduce the model size but was unsuccessfully implemented as the model's layers are incompatible with the official Keras layers such as normal or sequential.

I managed to find a solution relatively quickly after encountering this problem. While encountering a problem with face alignment of the images taken using the ESP32-CAM camera, I managed to find out that Espressif has a face recognition model that requires the face to be aligned before using it in the system. To align the face, they have a face detection model that detects the face in the image and outputs a bounding box around the face. The bounding box is used to crop out the face as a 58x58 image. I managed to change to the face recognition model that Espressif has made using FRMN model based on ArcFace. The face detection model uses the MTMN model based on MobileNetV2 model and Multi-task Cascaded Convolutional Networks.

Another problem I encountered was that I found a script on the internet that does face recognition with the requirements that I need but the problem is that it relies on the training dataset built using my own face resulting in the model being more bias. It will compare both images of me having high similarity score but when I tried to use 2 images of another person's face, the result output a low similarity score. I believe this is due to the training dataset relying on my face to do the training, resulting in the model having a high bias and output low similarity score of any face that is not my face, even though I tried to compare 2 images of another person's face.

A problem I also faced was that my Jupyter Notebook uses CPU instead of GPU which took too long to train my model. After creating a virtual environment in anaconda and using the GPU version of Tensorflow, I managed to get the script to use GPU cores for model training.

```
tensorflow                2.6.0           gpu_py39he88c5ba_0
tensorflow-base           2.6.0           gpu_py39hb3da07e_0
```

Figure 48. GPU Version of Tensorflow

However, another problem I faced was during model training, I got an OOM error which is related to the model using the GPU RAM fully and will stop training when this happens. I used the code shown below to check if TensorFlow uses the GPU or not. I also changed the batch size from 256 to 16 and the model did not get errors after. The accuracy is high at 0.91674 and training loss is 0.01077 after 15 epochs of training.

```
# Avoid OOM errors by setting GPU Memory Consumption Growth
gpus = tf.config.experimental.list_physical_devices('GPU')
for gpu in gpus:
    tf.config.experimental.set_memory_growth(gpu, True)
```

```
print("Num GPUs Available: ", len(tf.config.experimental.list_physical_devices('GPU')))
```

```
Num GPUs Available:  1
```

```
tf.config.list_physical_devices('GPU')
```

```
[PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]
```

Figure 49. Code to set GPU

In this section of the code, I created a function that will do inference on the images I collected. However, the model expects the input shape of [1, height, width, 3] but the shape of my images are [2, height, width, 3]. Therefore, I resize the tensor inputs to fit the images into the input. get_tensor function returns the inference which is the encodings of the images. This will be used to calculate similarity distance and output the prediction.

In [36]:
```
# Get tensor of output data after setting input data as input tensor
#https://heartbeat.comet.ml/running-tensorflow-lite-image-classification-models-in-python-92ef44b4cd47
def interpret(face_list1, face_list2, threshold=1.3):
    interpreter.resize_tensor_input(input_index[0]['index'],[len(face_list1), 128, 128, 3])
    interpreter.allocate_tensors()
    interpreter.set_tensor(input_index[0]['index'], face_list1)
    interpreter.invoke()
    tensor1 = interpreter.get_tensor(output_index[0]['index'])

    interpreter.resize_tensor_input(input_index[0]['index'],[len(face_list2), 128, 128, 3])
    interpreter.allocate_tensors()
    interpreter.set_tensor(input_index[0]['index'], face_list2)
    interpreter.invoke()
    tensor2 = interpreter.get_tensor(output_index[0]['index'])

    # After getting tensor1 and tensor2, calculate similarity distance and output prediction
    distance = np.sum(np.square(tensor1-tensor2), axis=-1)
    prediction = np.where(distance<=threshold, 0, 1)
    return prediction

    #check tensor1 and tensor2 contains what thing first on monday
```

In [37]:
```
for n in range(5):
    print(user_triplet[n])
```

```
(('1', '0.jpg'), ('1', '1.jpg'), ('379', '2.jpg'))
(('1', '2.jpg'), ('1', '3.jpg'), ('1357', '1.jpg'))
(('1', '4.jpg'), ('1', '5.jpg'), ('742', '3.jpg'))
(('1', '6.jpg'), ('1', '7.jpg'), ('1267', '1.jpg'))
(('1', '8.jpg'), ('1', '9.jpg'), ('1452', '0.jpg'))
```

Figure 50. Function to do Inference

In [39]:
```
pos_list = np.array([])
neg_list = np.array([])

for data in get_batch_user(user_triplet):
    a, p, n = data
    print(len(a))
    # Use classify_images function to output similiarity score
    pos_list = np.append(pos_list, interpret(a, p))
    neg_list = np.append(neg_list, interpret(a, n))
    break
print(pos_list)
print(neg_list)
```

```
5
[0. 0. 0. 0. 0.]
[1. 1. 1. 1. 1.]
```

Figure 51. Code to Call Inference Function and Result

# 9. Future Development

## 9.1. Prototype Material

I intend to demonstrate the cardboard material for my final presentation, but I desire a material that is low cost but is durable and fool proof for my future final product. Currently, there are materials that are durable and lightweight like carbon fibre or titanium but they are very expensive and meant for professional use.

## 9.2. Face Recognition System

The face recognition system would be more impressive had the TensorFlow Lite implementation worked. The model size needs to be reduced to fit in ESP32-CAM or I would buy a microcontroller that can support a bigger size RAM to fit in the model.

## 9.3. Multiple Lockers

I would add in more lockers if I could build a bigger prototype. Adding in more lockers would help in demonstrating that the locker system can keep track of multiple lockers being used or vacant.

## 9.4. Auto Door

I desire the auto door hinge to be replaced by a more automatic system using servos to control the movement of the door. This will allow a more hands-off approach suitable during pandemics or for people who do not like to get into contact with surfaces often like people suffering from OCD (Obsessive Compulsive Disorder).

## 9.5. Lithium Battery

Initially, I envisioned the prototype to be more self-reliant and sustainable to include a lithium battery instead of powering from the power plug. However, due to lack of time, I was not able to implement the battery.

Figure 52. Lithium Battery

### 9.6. Solar Panels

The solar panels along with the lithium battery are meant to complement each other and make the prototype more sustainable. However, the solar panels on the market are very expensive and will break my budget for the project.

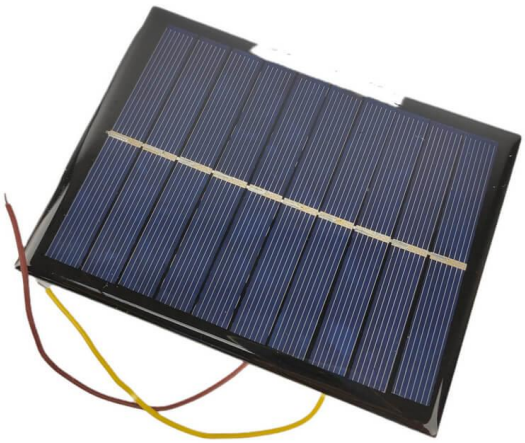


Figure 53. Solar Panels

### 9.7. Text to Speech

Text to Speech would allow the user to understand the messages outputted by the locker system verbally. This also helps the visually impaired as they could not see the LCD displays visually.

### 9.8. Speech Recogniser

Like Text to Speech, the speech recogniser will help the visually impaired to command the locker system verbally. It also brings benefits to normal people who want to interact with the locker system without the need to touch the buttons or surfaces of the locker system, reducing contact. This will be very beneficial during a pandemic or users who suffer from OCD (Obsessive Compulsive Disorder).

### 9.9. Mobile Application

If possible, I desire a mobile application for users to control the locker system. The application can do face or fingerprint authentication if the mobile device allows and can monitor and track the usage and condition of the locker they are using.

### 9.10. Internal Camera for Locker System

The internal camera would be installed inside the locker system to allow users to view the locker contents and also keep track if the lockers were to break in by thieves.

# 10. Conclusion

The locker system uses fingerprint sensor and face recognition to authenticate users to use the lockers. The Arduino UNO board controls the overall locker system while ESP32-CAM controls the face recognition system. However, there are a few more kinks to iron out before the overall locker system is smooth to use.

The design of the locker system can be accommodated for future ideas and developments. One of the goals for the project is to make the locker system more sustainable using solar panels and a battery to power the mainboards instead of wall plugs.

Another goal for the project is to use TensorFlow Lite implementation for the face recognition system. Although there was no time to finish troubleshooting the problems that plagued the implementation, I hope the framework will be more mature in the future to be able to deploy in ESP32-CAM or any future microcontroller that is more powerful to deploy the model.

# 11. Reflection on Learning Outcome Attainment

## 11.1 Engineering Knowledge

Before taking on this project, I did not have a lot of knowledge on Arduino and ESP systems. However after taking on this project, I managed to learn a lot about the systems and the components that can be used with Arduino and ESP systems.

I also did not have knowledge on biometric authentication systems before taking on this project. I realise the components required for biometric authentication are cheap and easy to implement. I also learnt more about the face recognition system and the components that run the system. As I am also interested in machine learning, I learnt more about Tensorflow Lite and its implementation even though I did not implement successfully in this project.

## 11.2 Design/Development of Solutions

In this project, I managed to design a circuit on the biometric authentication systems on my own. I bought my own components and implemented them into the circuit. I also bought corrugated cardboard and modified it to become a locker prototype.

There are times when I realised after buying the components I need that I needed to buy more components because I ran out of them or I realised I needed to buy those that I do not have.

## 11.3 Problem Analysis

I encountered many problems doing this project. At first, I was stressed whether I could resolve the problem. After searching for solutions online, I managed to find one that can resolve the problem. I felt very relieved and rewarded resolving the problems I faced.

With ChatGPT released a few months ago, I managed to write prompts to understand the problems I faced. Although the answers are not perfect, I could interpret what ChatGPT was trying to explain and get the solution relatively quicker than searching for solutions online from scratch.

# 12. References

[1] Hi Tech Lockers, “An Introduction to Essential Locker Features - Locker Basic Information,” 2022. [Online]. Available: https://www.hitechlockers.com/introduction-essential-locker-features/.

[2] GlobeNewswire, “Locker Market Size 2021 Analysis By Global Trends,” 17 November 2021. [Online]. Available: GlobeNewswire.

[3] KUSA-TV, “Littleton Police warn about increase in storage facility thefts,” 27 May 2022. [Online]. Available: https://www.9news.com/article/news/crime/storage-facility-theft-increase-littleton/73-1006fbad-30e6-43c1-b4f1-3b99ac43fb32.

[4] S. Ashley, “Under lock and key,” in *Mechanical engineering,* 1993.

[5] A. &. C. L. &. J. G. &. C. M. De Angeli, “Usability and user Authentication: Pictorial passwords vs. PIN,” *Contemporary Ergonomics,* pp. 253-258, 2003.

[6] M. A. A. V.-R. a. I. B. Prada-Delgado, “Physical unclonable keys for smart lock systems using bluetooth low energy,” *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society,* pp. 4808-4813, 2016.

[7] A. a. I. A. M. Perdana, “Seniors’ adoption of digital devices and virtual event platforms in Singapore during Covid-19,” *Technology in Society 68,* p. 101817, 2022.

[8] M. M. e. a. Ali, “Overview of fingerprint recognition system,” *2016 international conference on electrical, electronics, and optimization techniques (ICEEOT),* 2016.

[9] S. e. a. Chanda, “Face recognition-A one-shot learning perspective,” *2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS),* pp. 113-119, 2019.

[10] S. S. P. C. R. a. K. B. R. Harakannanavar, “Comprehensive study of biometric authentication systems, challenges and future trends,” *International Journal of Advanced Networking and Applications 10.4,* vol. 10, no. 4, pp. 3958-3968, 2019.

[11] A. H. M. Z. A. Z. L.-C. C. Mark Sandler, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” 21 March 2019. [Online]. Available: https://arxiv.org/pdf/1801.04381.pdf.

[12] Z. Z. Z. L. Y. Q. Kaipeng Zhang, “Joint Face Detection and Alignment using,” [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1604/1604.02878.pdf.

[13] I. Kontaev, “Face Recognition with ArcFace,” 1 Feburary 2021. [Online]. Available: https://learnopencv.com/face-recognition-with-arcface/.

[14] J. Brownlee, “Machine Learning Mastery,” Machine Learning Mastery, 10 June 2020. [Online]. Available: https://machinelearningmastery.com/one-shot-learning-with-siamese-networks-contrastive-and-triplet-loss-for-face-recognition/. [Accessed 30 August 2022].

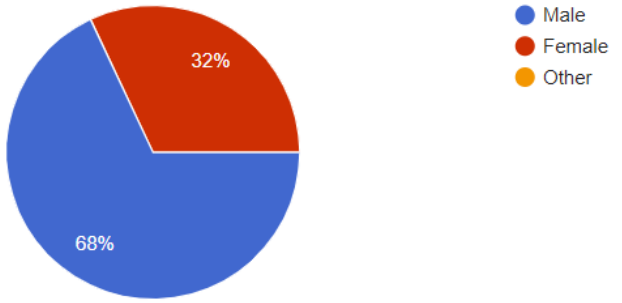
# 13. Appendix

## Appendix A. Survey Response

What is your gender?

25 responses

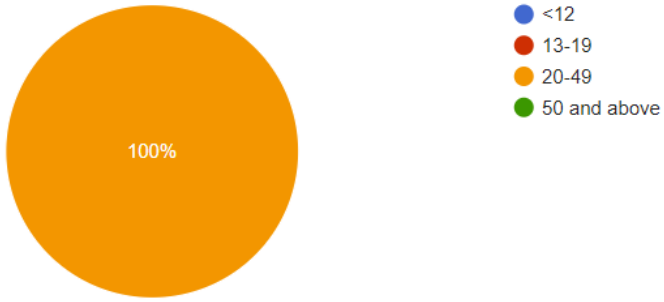
Copy




What is your age group?

25 responses

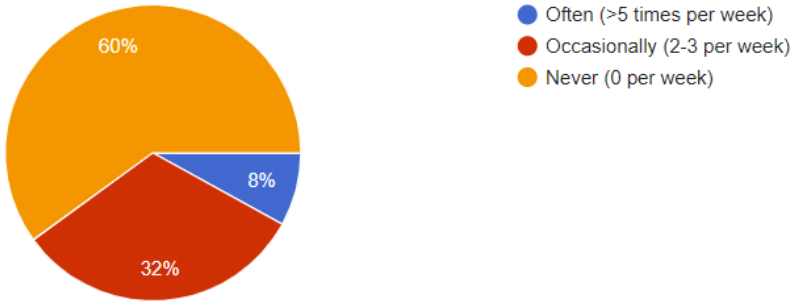
Copy




How often do you use lockers?

25 responses

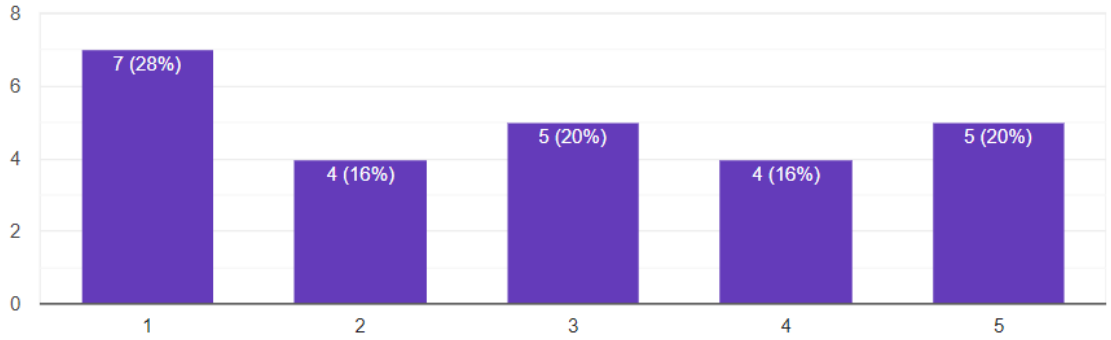
Copy



Do you lose locker keys/forget PIN numbers for lockers easily?

Copy

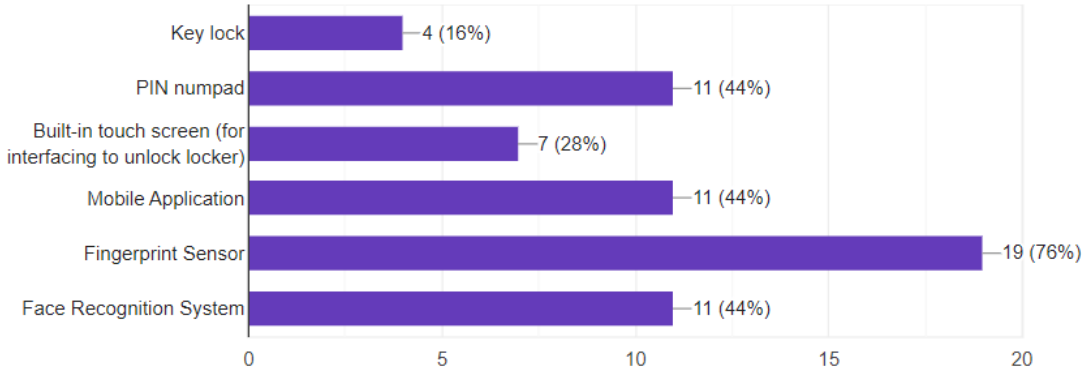
25 responses




What authentication system would you want to use?

Copy

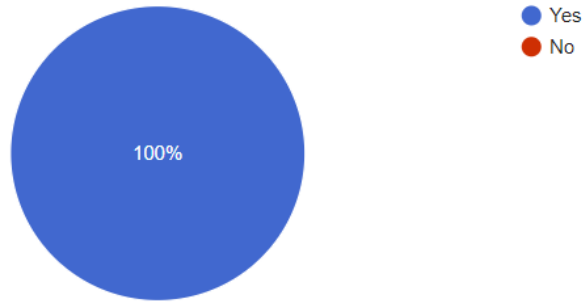
25 responses




Do you think having a Fingerprint Sensor/Face Recognition System in lockers is beneficial for you?

Copy

25 responses

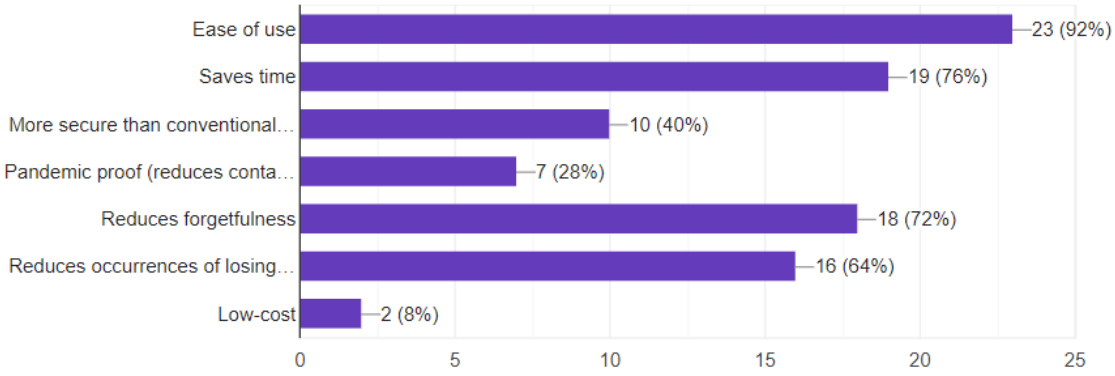


## Beneficial

Which reasons do you think Fingerprint Sensor/Face Recognition System is beneficial for you?

Copy

25 responses




## Not beneficial

Which reasons do you think Fingerprint Sensor/Face Recognition System is not beneficial for you?

0 responses

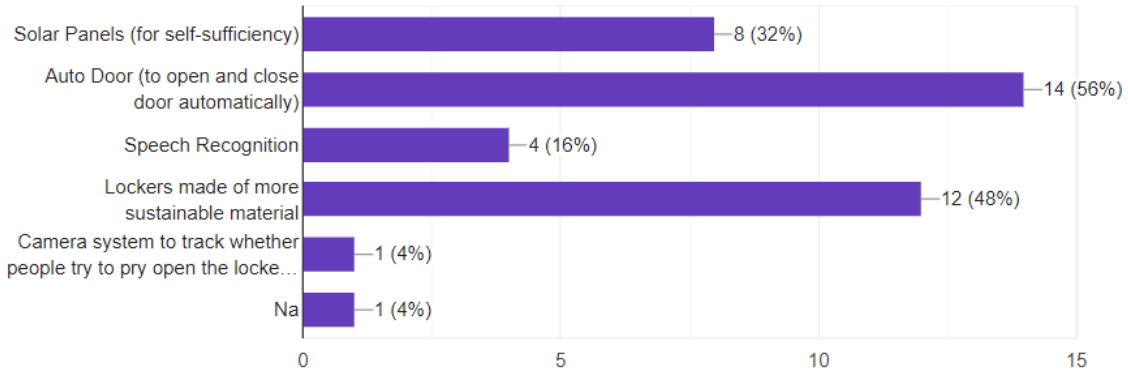
No responses yet for this question.

## Additional Features

### What additional features do you want for the locker?

Copy

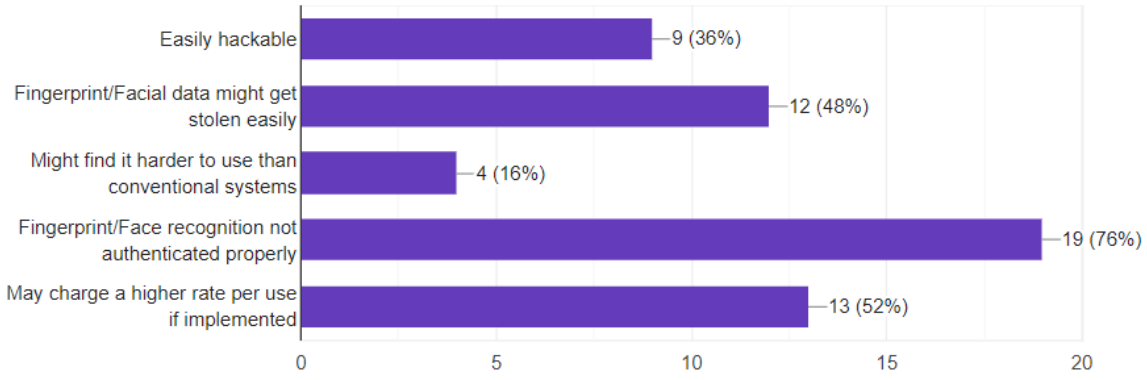
25 responses




### What are your concerns using Fingerprint Sensor/Face Recognition Systems?

Copy

25 responses



**Appendix B: Arduino UNO Code**

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
#include "string.h"

#include <Servo.h>
#include <LiquidCrystal_I2C.h>

#if (defined(__AVR__) || defined(ESP8266)) && !defined(__AVR_ATmega2560__)
// For UNO and others without hardware serial, we must use software serial...
// pin #2 is IN from sensor (GREEN wire)
// pin #3 is OUT from arduino  (WHITE wire)
// Set up the serial port to use softwareserial..
SoftwareSerial mySerial(2, 3);   //serial to fingerprint sensor
SoftwareSerial SoftSerial(4, 5); //serial to esp32 cam

#else
// On Leonardo/M0/etc, others with hardware serial, use hardware serial!
// #0 is green wire, #1 is white
#define mySerial Serial1

#endif

Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);

//variables to keep track of various tasks
uint8_t id;
uint8_t number;
uint8_t number_delete;
uint8_t user_delete;
```

```
uint8_t finger_face;
uint8_t i;
uint8_t enroll_ok;
uint8_t both;
uint8_t es_finger;
uint8_t es_face;
uint8_t choose;
uint8_t ok_search;
uint8_t mismatch;

//variable for serial communication
bool done;

//variables to keep track of user using either fingerprint or face recognition
uint8_t user1 = 0;
uint8_t user2 = 0;
uint8_t user_pref;

//variables to keep track of locker availability
int facelist1 = 0;
int facelist2 = 0;

//servo variables
Servo myServo1;
Servo myServo2;
int pos = 0;

//magnetic reed switch and button variables
int switch1 = 6;
int switch2 = 7;
int buttonApin = 10;
```

```
int buttonBpin = 11;
int buttonCpin = 12;

// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  SoftSerial.begin(115200);
  SoftSerial.write("");

  lcd.begin();
  lcd.setCursor(0, 0);
  lcd.print("UNO System!");

  pinMode(switch1, INPUT_PULLUP);
  pinMode(switch2, INPUT_PULLUP);
  pinMode(buttonApin, INPUT_PULLUP);
  pinMode(buttonBpin, INPUT_PULLUP);
  pinMode(buttonCpin, INPUT_PULLUP);
  myServo1.attach(8);
  myServo2.attach(9);
  delay(100);
  myServo1.write(0);
  myServo2.write(0);

  finger.begin(57600);

  // find fingerprint sensor
  if (finger.verifyPassword()) {
```

```
    Serial.println("Found fingerprint sensor!");
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1) {
      delay(1);
    }
  }
}

//function to read inputs from serial monitor
uint8_t readnumber(void) {
  uint8_t num = 0;

  while (num == 0) {
    while (! Serial.available());
    num = Serial.parseInt();
  }
  return num;
}

void loop() {
  //initialise and reset everytime loop comes here
  String terminalText;
  id = 0;
  finger_face = 0;
  user_pref = 0;
  user_delete = 0;
  number_delete = 0;
  choose = 0;

  lcd.clear();
```

```
  lcd.setCursor(0, 0);
  lcd.print("1 new user");
  lcd.setCursor(0, 1);
  lcd.print("2 old user");
  //Serial.println(F("Select: 1 for new user (enroll) and 2 for old user (searching)"));


  while ((digitalRead(buttonApin) == HIGH) and (digitalRead(buttonBpin) == HIGH) and
(digitalRead(buttonCpin) == HIGH));
  if (digitalRead(buttonApin) == LOW) {
    number = 1;
  }
  else if (digitalRead(buttonBpin) == LOW) {
    number = 2;
  }
  else if (digitalRead(buttonCpin) == LOW) {
    number = 3;
  }
  delay(1000);
  switch (number) {
    case 1:
      //use either fingerprint/face recognition or both
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Use 1 or both?");
      while ((digitalRead(buttonApin) == HIGH) and (digitalRead(buttonBpin) == HIGH));
      if (digitalRead(buttonApin) == LOW) {
        both = 1;
      }
      else if (digitalRead(buttonBpin) == LOW) {
        both = 2;
        finger_face = 3;
```

```
}

delay(1000);

//if user choose either 1, choose either fingerprint or face
if (finger_face != 3) {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("1 fingerprint");
  lcd.setCursor(0, 1);
  lcd.print("2 face");
  //Serial.println(F("Select: 1 for fingerprint sensor and 2 for face recognition system"));

  while ((digitalRead(buttonApin) == HIGH) and (digitalRead(buttonBpin) == HIGH));
  if (digitalRead(buttonApin) == LOW) {
    finger_face = 1;
  }
  else if (digitalRead(buttonBpin) == LOW) {
    finger_face = 2;
  }
}
delay(1000);

//enroll function
Serial.println(F("Selected enrolling"));
if (finger_face == 1) {
  Serial.println(F("Ready to enroll a fingerprint!"));
}
else if (finger_face == 2) {
  Serial.println(F("Ready to enroll a face!"));
}
```

```
//select locker 1 or 2
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("locker 1 or 2?");
while ((digitalRead(buttonApin) == HIGH) and (digitalRead(buttonBpin) == HIGH));
if (digitalRead(buttonApin) == LOW) {
  Serial.println(F("Selected locker 1"));
  id = 1;
  if (user1 != 0) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Locker 1 in use!");
    finger_face = 0;
  }
}
else if (digitalRead(buttonBpin) == LOW) {
  id = 2;
  if (user2 != 0) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Locker 2 in use!");
    finger_face = 0;
  }
}
delay(1000);

//start to enroll
if (finger_face == 1) {
  enroll_ok = finger_enroll(id);
```

```
  if (enroll_ok == 1) {
    locker(id, 1);
  }
  delay(50);
  enroll_ok = 0;
 }
 else if (finger_face == 2) {
  enroll_ok = face_enroll(id);
  if (enroll_ok == 1) {
    locker(id, 2);
  }
 }
 else if (finger_face == 3) {
  es_finger = finger_enroll(id);
  es_face = face_enroll(id);
  if (es_finger == es_face) {
    locker(id, 3);
  }

 }
 break;
case 2:
 //choose either locker 1 or 2
 lcd.clear();
 lcd.setCursor(0, 0);
 lcd.print("locker 1 or 2");
 //Serial.println(F("Select : 1 for locker 1 and 2 for locker 2"));
 //number = readnumber();

 while ((digitalRead(buttonApin) == HIGH) and (digitalRead(buttonBpin) == HIGH));
 if (digitalRead(buttonApin) == LOW) {
```

```
  number = 1;
}
else if (digitalRead(buttonBpin) == LOW) {
  number = 2;
}

//retrieve int variable for fingeprint/face from finger_face
if (number == 1) {
  user_pref = user1;

}
else if (number == 2) {
  user_pref = user2;
}
delay(1000);

//if user don't want to continue using, delete data after searching is successful
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Continue using locker?");

while ((digitalRead(buttonApin) == HIGH) and (digitalRead(buttonBpin) == HIGH));
if (digitalRead(buttonApin) == LOW) {
  number_delete = 1;
}
else if (digitalRead(buttonBpin) == LOW) {
  number_delete = 2;
}
delay(1000);

//retrieve int variable for fingeprint/face from finger_face
```

```
if (number_delete == 1) {
  if (number == 1) {
    user_delete = 1; //don't delete for user 1
  }
  else if (number == 2) {
    user_delete = 2; //don't delete for user 2
  }
}
else if (number_delete == 2) {
  if (number == 1) {
    user_delete = 3; //delete for user 1
  }
  else if (number == 2) {
    user_delete = 4; //delete for user 2
  }
}

//start search function
if (user_pref == 3) {
  if (number_delete != 2) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("fingerprint or face");
    while ((digitalRead(buttonApin) == HIGH) and (digitalRead(buttonBpin) == HIGH));
    if (digitalRead(buttonApin) == LOW) {
      choose = 1;
    }
    else if (digitalRead(buttonBpin) == LOW) {
      choose = 2;
    }
  }
```

```
}

if (user_pref == 1) {
  ok_search = finger_search(user_pref, user_delete, 0);
  delay(1000);
}
else if (user_pref == 2) {
  ok_search = face_search(user_pref, user_delete);
}
else if (user_pref == 3) {
  if (choose == 1) {
    es_finger = finger_search(user_pref, user_delete, 0);
  }
  else if (choose == 2) {
    es_face = face_search(user_pref, user_delete);
  }
  else {
    es_finger = finger_search(user_pref, user_delete, 1);
    es_face = face_search(user_pref, user_delete);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("All deleted!");
  }
}

if (user_pref == 0) {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Locker not registered!");
}
delay(1000);
```

```
    break;
  case 3:
    //debugging function
    SoftSerial.listen();

    SoftSerial.write("zzz");
    done = false;
    while (!done) {
      Serial.println(F("Waiting"));
      delay(50);
      if (SoftSerial.available()) {
        terminalText = SoftSerial.readStringUntil('\n');
        Serial.println(F("SoftSerial in"));
        Serial.println(terminalText);
        if (terminalText.indexOf("y") != -1) { //True
          Serial.println("Image is True");
          done = true;
        }
        else {
          done = true;
        }
        delay(500);
        terminalText = SoftSerial.readStringUntil('\n');
      }
    }
    break;
  default:
    Serial.println("Error!");
    delay(50);
    break;
}
```

```
}

//locker function to open and close locker
void locker(int id, int auth) {
  //open locker
  if (id == 1) {
    myServo1.write(180);
    delay(50);
  }
  else if (id == 2) {
    myServo2.write(180);
    delay(50);
  }

  //close locker
  if (id == 1) {
    while (digitalRead(switch1) == HIGH);
    //if switch1 is low, close locker
    myServo1.write(0);
    if (auth == 1) {
      user1 = 1;
    }
    else if (auth == 2) {
      user1 = 2;
    }
    else if (auth == 3) {
      user1 = 3;
    }
  }
  else if (id == 2) {
```

```
    while (digitalRead(switch2) == HIGH);
    //if switch2 is low, close locker
    myServo2.write(0);
    if (auth == 1) {
      user2 = 1;
    }
    else if (auth == 2) {
      user2 = 2;
    }
    else if (auth == 3) {
      user2 = 3;
    }
  }
  delay(1000);
}

//enroll fingerprint function
int finger_enroll(uint8_t id) {
  int finger_ok;
  mySerial.listen();

  delay(50);
  for (i = 0; i <= 2; i++) {
    getFingerprintEnroll();
    delay(500);
    if (mismatch == 1) {
      enroll_ok = 0;
      mismatch = 0;
    }
    if (enroll_ok == 1) {
      //get out of for loop early if fingerprint is registered
```

```
      finger_ok = 1;
      break;
    }
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Try again!");
  }
  return finger_ok;
}

//enroll face function
int face_enroll(int id) {
  int face_ok;
  String terminalText;
  SoftSerial.listen();

  //face recognition system
  if (id == 1) {
    SoftSerial.write("aaa");
  }
  else if (id == 2) {
    SoftSerial.write("ccc");
  }
  delay(5000);

  done = false;
  while (!done) {
    delay(50);
    if (SoftSerial.available()) {
      terminalText = SoftSerial.readStringUntil('\n');
      delay(1000);
```

```
    if ((terminalText.indexOf("b") != -1) or (terminalText.indexOf("d") != -1)) { //True
      if (terminalText.indexOf("b") != -1) {
        facelist1 = 1;
      }
      else if (terminalText.indexOf("d") != -1) {
        facelist2 = 1;
      }
      done = true;
      face_ok = 1;
    }
    else if (terminalText.indexOf("s") != -1) { //False
      done = true;
      face_ok = 0;
    }
  }
 }
 return face_ok;
}

//search fingerprint function
int finger_search(int id, int user_delete, int skip) {
 int search_ok;
 mySerial.listen();

 //if doesn't detect after 50 times, exit searching function
 for (i = 0; i <= 50; i++) {
  getFingerprintID();
  if (finger.fingerID) {
    break;
  }
 }
```

```
if (finger.fingerID == 0) {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("fingerprint not found!");
  search_ok = 0;
}
delay(50);

if (finger.fingerID == 1) {
  if ((skip != 1) && (finger.fingerID == id)) {
    locker(1, 0);
  }
  else {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Authentication failed!");
  }
  search_ok = 1;
  if (user_delete == 3) {
    //delete function
    deleteFingerprint(finger.fingerID);
    user1 = 0;
    delay(50);
  }
  else if (user_delete == 1) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Have a nice day!"));
  }
}
else if (finger.fingerID == 2) {
```

```
    if ((skip != 1) && (finger.fingerID == id)) {
      locker(2, 0);
    }
    else {
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Authentication failed!");
    }
    search_ok = 1;
    if (user_delete == 4) {
      //delete function
      deleteFingerprint(finger.fingerID);
      delay(50);
      user2 = 0;
    }
    else if (user_delete == 2) {
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Have a nice day!"));
    }
  }
  return search_ok;
}

//search face function
int face_search(int id, int user_delete) {
  int search_ok;
  String terminalText;
  SoftSerial.listen();

  if (user_delete == 1) {
```

```
  SoftSerial.write("eee");
}
else if (user_delete == 2) {
  SoftSerial.write("ggg");
}
else if (user_delete == 3) {
  SoftSerial.write("iii");
}
else if (user_delete == 4) {
  SoftSerial.write("kkk");
}

delay(5000);

done = false;
while (!done) {
  delay(50);
  if (SoftSerial.available()) {
    terminalText = SoftSerial.readStringUntil('\n');
    delay(1000);
    if ((terminalText.indexOf("f") != -1) or (terminalText.indexOf("h") != -1)) { //True
      done = true;
      locker(number, 0);
      search_ok = 1;
    }
    else if ((terminalText.indexOf("j") != -1) or (terminalText.indexOf("l") != -1)) { //True
      done = true;
      if (terminalText.indexOf("j") != -1) {
        facelist1 = 0;
        user1 = 0;
      }
```

```
      else if (terminalText.indexOf("l") != -1) {
        facelist2 = 0;
        user2 = 0;
      }
      locker(number, 0);
      search_ok = 1;


    }
    else if ((terminalText.indexOf("m") != -1) or (terminalText.indexOf("n") != -1) or
(terminalText.indexOf("o") != -1) or (terminalText.indexOf("p") != -1)) { //False
      done = true;
      search_ok = 0;
    }
   }
  }
  return search_ok;
}




//Adafruit function
uint8_t getFingerprintEnroll() {


  int p = -1;
  Serial.print("Waiting for valid finger to enroll as #"); Serial.println(id);
  delay(500);
  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");
  mismatch = 1;
  return p;
} else {
  Serial.println("Unknown error");
  return p;
}

Serial.print("ID "); Serial.println(id);
p = finger.storeModel(id);
if (p == FINGERPRINT_OK) {
  enroll_ok = 1;
  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;
  }

  return true;
}

//Adafruit function
uint8_t deleteFingerprint(uint8_t id) {
  uint8_t p = -1;

  p = finger.deleteModel(id);

  if (p == FINGERPRINT_OK) {
    Serial.println("Deleted!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
  } else if (p == FINGERPRINT_BADLOCATION) {
    Serial.println("Could not delete in that location");
  } else if (p == FINGERPRINT_FLASHERR) {
    Serial.println("Error writing to flash");
  } else {
    Serial.print("Unknown error: 0x"); Serial.println(p, HEX);
  }

  return p;
}
```

```
//Adafruit function
uint8_t getFingerprintID() {
  //reset fingerID to 0 if this function has been used before
  finger.fingerID = 0;
  //keep looping function until ID has been found
  while (1) {
    uint8_t p = finger.getImage();
    switch (p) {
      case FINGERPRINT_OK:
        Serial.println("Image taken");
        break;
      case FINGERPRINT_NOFINGER:
        Serial.println("No finger detected");
        return p;
      case FINGERPRINT_PACKETRECIEVEERR:
        Serial.println("Communication error");
        return p;
      case FINGERPRINT_IMAGEFAIL:
        Serial.println("Imaging error");
        return p;
      default:
        Serial.println("Unknown error");
        return p;
    }

    // OK success!

    p = finger.image2Tz();
    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!
p = finger.fingerSearch();
if (p == FINGERPRINT_OK) {
  Serial.println("Found a print match!");
} else if (p == FINGERPRINT_PACKETRECIEVEERR) {
  Serial.println("Communication error");
  //return p;
} else if (p == FINGERPRINT_NOTFOUND) {
  Serial.println("Did not find a match");
  //return p;
} else {
  Serial.println("Unknown error");
  //return p;
```

```
    }

    // found a match!
    if (finger.fingerID != 0) {
      Serial.print("Found ID #"); Serial.print(finger.fingerID);
      Serial.print(" with confidence of "); Serial.println(finger.confidence);

      return finger.fingerID;
    }
  }
}
```

**Appendix C: ESP32-CAM Code**

```
#include "esp_camera.h"
#include "fd_forward.h" //works with esp32 board library 1.0.5
#include "fr_forward.h"
#include "fr_flash.h"

#include "soc/soc.h"           // Disable brownout problems
#include "soc/rtc_cntl_reg.h"  // Disable brownout problems

#include <Wire.h>
#include <LiquidCrystal_I2C.h>

#define ENROLL_CONFIRM_TIMES 5
#define FACE_ID_SAVE_NUMBER 7

#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27
#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
```

```
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22

// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 16, 2);

bool initCamera() {

  camera_config_t config;

  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;
  config.frame_size = FRAMESIZE_QVGA;
```

```
  config.jpeg_quality = 10;
  config.fb_count = 1;

  esp_err_t result = esp_camera_init(&config);

  if (result != ESP_OK) {
    return false;
  }

  return true;
}

//variables to keep track
int pictureNumber = 0;
int deleted_id;

mtmn_config_t mtmn_config = {0};
int detections = 0;

//variables to store face id
static face_id_list id_list1 = {0};
static face_id_list id_list2 = {0};

//variables to hold images
static dl_matrix3du_t *image_matrix = NULL;
dl_matrix3du_t *aligned_face = dl_matrix3du_alloc(1, FACE_WIDTH, FACE_HEIGHT,
3);
dl_matrix3du_t *resized_matrix = dl_matrix3du_alloc(1, 128, 128, 3);

void setup() {
  WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); //disable brownout detector
```

```
  Serial.begin(115200);
  Wire.begin(15, 14); //SCL, SDA

  // initialize the LCD
  lcd.begin();
  lcd.backlight();

  lcd.setCursor(0, 0);
  lcd.print("Face Recognition System!");

  if (!initCamera()) {
    lcd.setCursor(0, 0);
    lcd.print("Failed to initialize camera...");
    return;
  }

  mtmn_config = mtmn_init_config();

  face_id_init(&id_list1, FACE_ID_SAVE_NUMBER, ENROLL_CONFIRM_TIMES);
  face_id_init(&id_list2, FACE_ID_SAVE_NUMBER, ENROLL_CONFIRM_TIMES);
}

//function to read number
uint8_t readnumber(void) {
  uint8_t num = 0;

  while (num == 0) {
    while (! Serial.available());
    num = Serial.parseInt();
  }
```

```
  return num;
}

//variables to keep track
uint8_t number;
int cap_num;
int id;
int cap_choose;

void loop() {
  //reset variables to 0 when loop comes here
  String terminalText = "";
  cap_num = 0;
  number = 0;
  id = 0;
  while (Serial.available() > 0) {
    terminalText = Serial.readStringUntil('\n');
    if (terminalText.indexOf("a") != -1) {
      number = 1;
      lcd.clear();
      lcd.setCursor(0, 1);
      lcd.print("enrollone");
    }
    if (terminalText.indexOf("c") != -1) {
      number = 2;
      lcd.clear();
      lcd.setCursor(0, 1);
      lcd.print("enrolltwo");
    }
    if (terminalText.indexOf("e") != -1) {
      number = 3;
```

```
  id = 1;
  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("searchone");
 }
 if (terminalText.indexOf("g") != -1) {
  number = 4;
  id = 2;
  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("searchtwo");
 }
 if (terminalText.indexOf("i") != -1) {
  number = 5;
  id = 1;
  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("searchdeleteone");
 }
 if (terminalText.indexOf("k") != -1) {
  number = 6;
  id = 2;
  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("searchdeletetwo");
 }
 if (terminalText.indexOf("z") != -1) {
  //number = 7;
  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("serialcycle");
```

```
  Serial.write("yyy");
  lcd.setCursor(0, 0);
  lcd.print("Sending back");
  delay(50);
 }
switch (number) {
  case 1:
    //enroll one
    lcd.setCursor(0, 0);
    lcd.print("start enro1llone");
    cap_num = capture_and_store(1);
    if (cap_num == 1) {
      Serial.write("bbb");
    }
    break;
  case 2:
    //enroll two
    cap_num = capture_and_store(2);
    if (cap_num == 1) {
      Serial.write("ddd");
    }
    break;
  case 3:
    //search one
    cap_num = capture_and_recognize(1);
    if (cap_num == 1) {
      Serial.write("fff"); //True
    }
    else{
      Serial.write("mmm"); //False
    }
```

```
  break;
case 4:
  //search two
  cap_num = capture_and_recognize(2);
  if (cap_num == 1) {
    Serial.write("hhh"); //True
  }
  else{
    Serial.write("nnn"); //False
  }
  break;
case 5:
  //search and delete one
  cap_num = capture_and_recognize(1);
  if (cap_num == 1) {
    while ( delete_face(&id_list1) > 0 ) {
      lcd.setCursor(0, 0);
      lcd.print("Deleting Face for list 1");
    }
    lcd.setCursor(0, 0);
    lcd.print("list 1 deleted");
    Serial.write("jjj"); //True
  }
  else{
    Serial.write("ooo"); //False
  }
  break;
case 6:
  //search and delete two
  cap_num = capture_and_recognize(2);
  if (cap_num == 1) {
```

```
        while ( delete_face(&id_list2) > 0 ) {
          lcd.setCursor(0, 0);
          lcd.print("Deleting Face for list 2");
        }
        lcd.setCursor(0, 0);
        lcd.print("list 2 deleted");
        Serial.write("lll"); //True
       }
       else{
        Serial.write("ppp"); //False
       }
       break;
     case 7:
       Serial.write("ttt");
       lcd.setCursor(0, 0);
       lcd.print("Sending back");
       delay(50);
       break;
     default:
       break;
   }
  }
}

//capture image and store as face id
int capture_and_store(int cap_choose) {
  int cap_store;
  int taken_img = 0;
  if (cap_choose == 1) {
    //face_list1
    while (taken_img < 5) {
```

```
    //captures image and store as face_id
    camera_fb_t * frame;
    frame = esp_camera_fb_get();

    dl_matrix3du_t *image_matrix = dl_matrix3du_alloc(1, frame->width, frame->height,
3);
    uint32_t res = fmt2rgb888(frame->buf, frame->len, frame->format,
image_matrix->item);

    if (!res) {
      lcd.setCursor(0, 0);
      lcd.print("to rgb888 failed");
      dl_matrix3du_free(image_matrix);
    }

    esp_camera_fb_return(frame);

    box_array_t *boxes = face_detect(image_matrix, &mtmn_config);
    if (boxes != NULL) {
      if (align_face(boxes, image_matrix, aligned_face) == ESP_OK) {
        int new_id = enroll_face(&id_list1 , aligned_face);
        lcd.clear();
        lcd.setCursor(0, 1);
        lcd.print("aligned face is ok");
        lcd.setCursor(0, 0);
        lcd.print(String(new_id) + " id for list 1");
        taken_img++;
      }
      else {
        lcd.setCursor(0, 0);
        lcd.print("Face Not Aligned");
```

```
      }
      dl_lib_free(boxes->score);
      dl_lib_free(boxes->box);
      dl_lib_free(boxes->landmark);
      dl_lib_free(boxes);
    }
    dl_matrix3du_free(image_matrix);
  }
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("list1 saved");
  cap_store = 1;
 }
 else if (cap_choose == 2) {
  //face_list2
  while (taken_img < 5) {
    //captures image and store as face_id
    camera_fb_t * frame;
    frame = esp_camera_fb_get();

    dl_matrix3du_t *image_matrix = dl_matrix3du_alloc(1, frame->width, frame->height,
3);
    uint32_t res = fmt2rgb888(frame->buf, frame->len, frame->format,
image_matrix->item);

    if (!res) {
      lcd.setCursor(0, 0);
      lcd.print("to rgb888 failed");
      dl_matrix3du_free(image_matrix);
    }

    esp_camera_fb_return(frame);
```

```
    box_array_t *boxes = face_detect(image_matrix, &mtmn_config);
    if (boxes != NULL) {
      if (align_face(boxes, image_matrix, aligned_face) == ESP_OK) {
        int new_id = enroll_face(&id_list2 , aligned_face);
        lcd.setCursor(0, 0);
        lcd.print(String(new_id) + " id for list 2");
        taken_img++;
      }
      else {
        lcd.setCursor(0, 0);
        lcd.print("Face Not Aligned");
      }
      dl_lib_free(boxes->score);
      dl_lib_free(boxes->box);
      dl_lib_free(boxes->landmark);
      dl_lib_free(boxes);
    }
    dl_matrix3du_free(image_matrix);

  }
  cap_store = 1;
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("list2 saved");
 }
 taken_img = 0;
 return cap_store;
}

//capture image and compare image to face id
```

```
int capture_and_recognize(int cap_choose) {
  int true_image = 0;
  int reco = 0;
  int cap_box;
  box_array_t *boxes;
  while (reco < 100) {
    //captures image and recognise face
    cap_box = 0;
    boxes = NULL;
    camera_fb_t * frame;
    frame = esp_camera_fb_get();

    dl_matrix3du_t *image_matrix = dl_matrix3du_alloc(1, frame->width, frame->height, 3);
    uint32_t res = fmt2rgb888(frame->buf, frame->len, frame->format, image_matrix->item);

    if (!res) {
      lcd.setCursor(0, 0);
      lcd.print("to rgb888 failed");
      dl_matrix3du_free(image_matrix);
    }

    esp_camera_fb_return(frame);

    boxes = face_detect(image_matrix, &mtmn_config);
    if (boxes != NULL) {
      lcd.setCursor(0, 0);
      lcd.print(String(reco) + " cap_box is good");
      cap_box = 10;
      //break;
    }
    else {
```

```
  reco++;
 }
 if (boxes != NULL) {
  if (align_face(boxes, image_matrix, aligned_face) == ESP_OK) {
    if (cap_choose == 1) {
      int matched_id = recognize_face(&id_list1, aligned_face);
      if (matched_id >= 0) {
        lcd.setCursor(0, 0);
        lcd.print(String(matched_id) + " id list1");
        true_image = 1;
        break;
      } else {
        lcd.setCursor(0, 0);
        lcd.print("No Match Found for id list 1");
        matched_id = -1;
        reco++;
      }
    }
    else if (cap_choose == 2) {
      int matched_id = recognize_face(&id_list2, aligned_face);
      if (matched_id >= 0) {
        lcd.setCursor(0, 0);
        lcd.print(String(matched_id) + " id list2");
        true_image = 1;
        break;
      } else {
        lcd.setCursor(0, 0);
        lcd.print("No Match Found for id list 2");
        matched_id = -1;
        reco++;
      }
```

```
      }
     }
     else {
      lcd.setCursor(0, 0);
      lcd.print("Face Not Aligned");
     }
     dl_lib_free(boxes->score);
     dl_lib_free(boxes->box);
     dl_lib_free(boxes->landmark);
     dl_lib_free(boxes);
    }
    dl_matrix3du_free(image_matrix);
   }
   if (reco == 100) {
    lcd.setCursor(0, 1);
    lcd.print("Incorrect face!");
    true_image = 0;
   }
   if (true_image == 1) {
    lcd.setCursor(0, 1);
    lcd.print("Correct face!");
   }
   reco = 0;
   return true_image;
}
```