

MAIN PROJECT REPORT
ELECTRONIC VAULT WITH TWO FACTOR
AUTHENTICATION

Submitted By

ADARSH S S (CEC19EC001)

BASTIN BABU(CEC19EC013)

HARIKRISHNA U(CEC19EC016)

KARTHIK P(LCEC19EC027)

under the esteemed guidance of

Ms. VINITHA GEORGE

(Associate Professor)

Department of Electronics and Communication Engineering



JUNE 2023

Department of Electronics and Communication Engineering
College of Engineering, Pallippuram P O, Cherthala,
Alappuzha Pin: 688541,
Phone: 0478 2553416, Fax: 0478 2552714
<http://www.cectl.ac.in>

MAIN PROJECT

**ELECTRONIC VAULT WITH TWO FACTOR
AUTHENTICATION**

Submitted By

ADARSH S S (CEC19EC001)

under the esteemed guidance of

Ms. VINITHA GEORGE

(Associate Professor)

In partial fulfillment of the requirements for the award of the degree

of

Bachelor of Technology

in

Electronics and Communication Engineering

of

APJ Abdul Kalam Technological University



JUNE 2023

Department of Electronics and Communication Engineering,

Pallippuram P O, Cherthala,

Alappuzha Pin: 688541,

Phone: 0478 2553416, Fax: 0478 2552714

<http://www.cectl.ac.in>

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING
COLLEGE OF ENGINEERING CHERTHALA
ALAPPUZHA-688541**



C E R T I F I C A T E

This is to certify that, the project report titled ***ELECTRONIC VAULT WITH TWO FACTOR AUTHENTICATION*** is a bonafide record of the **MAIN PROJECT** presented by **ADARSH S S (CEC19EC001)** Eighth Semester B. Tech. Electronics & Communication Engineering student, under our guidance and supervision, in partial fulfillment of the requirements for the award of the degree, **B. Tech. Electronics & Communication Engineering** of **APJ Abdul Kalam Technological University**.

Guide

Co-ordinator

HoD

Ms. Vinitha George

Associate Professor

Dept. of Electronics

Mr. Sreekumar K

Assistant Professor

Dept. of Electronics

Dr. Ashok kumar

Associate Professor

Dept. of Electronics

ACKNOWLEDGEMENT

This work would not have been possible without the support of many people. First and the foremost, I give thanks to Almighty God who gave us the inner strength, resource and ability to complete my project successfully.

I would like to thank **Dr. Jaya V L**, the Principal, who has provided with the best facilities for the project completion and presentation. I would also like to thank my HoD **Dr. Ashok Kumar** (Associate Professor in Department of Electronics Engineering), my project coordinator **Mr. Sreekumar K** (Assistant Professor in Department of Electronics Engineering), and my guide **Ms. Vinitha George** (Associate Professor in Department of Electronics Engineering) for the help extended and also for the encouragement and support given to me while doing the project.

I would like to thank my dear friends for extending their cooperation and encouragement throughout the project work, without which I would never have completed the project this well. Thank you all for your love and also for being very understanding.

ABSTRACT

In this project, we present a robust implementation of two-factor authentication using Raspberry Pi, leveraging the power of biometric authentication through fingerprint and facial recognition. This advanced system introduces an additional layer of security using a MySQL database to safeguard user accounts against unauthorized access and potential breaches.

The Raspberry Pi serves as the central processing unit, equipped with a camera module for capturing facial images and a fingerprint sensor module for scanning fingerprints. Through sophisticated machine learning algorithms, the Raspberry Pi analyzes and stores the biometric data securely. The system compares the captured facial image with the registered image stored in the database and authenticates the user's fingerprint against the registered prints.

By combining both facial and fingerprint identification, the system ensures that access is granted only when both biometric modalities match, effectively eliminating the vulnerabilities associated with traditional password-based authentication. This innovative approach not only enhances security but also offers a seamless and user-friendly authentication process.

Given the escalating threats in the digital landscape, this cutting-edge technology has significant potential for widespread implementation. It can effectively mitigate the risks of unauthorized access, safeguarding sensitive information and aid overall security measures in various domains, including personal devices, corporate networks, and sensitive data repositories. With its proven effectiveness and robustness, this biometric recognition system holds great promise in the ongoing battle against cybersecurity threats.

Contents

1	INTRODUCTION	1
1.1	Introduction About The Project	1
1.2	Aim Of The Project	2
2	LITERATURE SURVEY	3
2.1	Different Methodologies Reviewed	3
3	DESIGN	6
3.1	Design Descriptions	6
3.1.1	Block Diagram	7
3.1.2	Flow Chart	8
3.1.3	Circuit Diagram	9
3.2	Components List	10
3.2.1	Raspberry Pi 4 Model B	10
3.2.2	Fingerprint Module	11
3.2.3	Servo Motor	12
3.2.4	Webcam	13
3.2.5	LCD Display	14
3.2.6	Power Supply	15
3.2.7	Breadboard	15
3.2.8	Jumper Wire	16
3.3	Estimated Cost	17
3.4	Software	18

3.5	Implementation	20
3.6	Work Plan	23
4	RESULT	24
5	CONCLUSION	25
5.1	Future Scope	25
	REFERENCES	27
A	PROGRAM CODE	29
A.1	DATA ENROLLMENT	29
A.2	DELETING DATA	34
A.3	AUTHENTICATION	36

List of Figures

3.1	Block diagram of the system	7
3.2	Flow chart	8
3.3	Circuit diagram of the system	9
3.4	Raspberry Pi 4 Model B	11
3.5	Raspberry Pi 4 Model B Pheriperals	11
3.6	Fingerprint Module	12
3.7	Fingerprint Module Peripherals	12
3.8	Towerpro SG90 Motor	13
3.9	Towerpro SG90 Motor Pinout	13
3.10	hp w100 Webcam	14
3.11	16x2 LCD Display Pinout	15
3.12	9v Power Supply	15
3.13	Breadboard	16
3.14	Jumper Wires	16
3.15	Cost Estimation	17
3.16	Installing OpenCV	19
3.17	Installing MySQL Database	20
3.18	Full Hardware setup (front view)	22
3.19	Full Hardware setup (top view)	22
4.1	Final output	24

Chapter 1

INTRODUCTION

1.1 Introduction About The Project

Electronic vaults are increasingly becoming popular among individuals and organizations that seek to secure their sensitive data and valuables. The conventional methods of securing these valuables, such as traditional lock-and-key systems, are no longer sufficient in today's digital age, where data breaches and cyber attacks are commonplace. As a result, electronic vaults that use sophisticated technologies to ensure maximum security are now being widely adopted. One such technology is two-factor authentication, which adds an extra layer of security to electronic vaults. In two-factor authentication, a user is required to provide two different types of authentication factors before being granted access to the vault. This ensures that even if one of the factors is compromised, the user's data or valuables remain secure.

Fingerprint and facial recognition are two of the most popular forms of two-factor authentication used in electronic vaults. These technologies are widely used in smartphones and other electronic devices, making them easily accessible and affordable. The Raspberry Pi is a small, low-cost computer that has become increasingly popular for its versatility and ease of use. It can be used in a wide range of projects, including electronic vaults. By combining Raspberry Pi with fingerprint and facial recognition technologies, it is possible to create a highly secure electronic vault that is both affordable and easy to use. Electronic vaults that use two-factor authentication, such as fingerprint and facial recognition, are becoming increasingly popular as a means of securing sensitive data and valuables. By using Raspberry Pi, it is possible to create an affordable and easy-to-use electronic vault that is highly secure. However, it is important to design the user

interface, database, and algorithms carefully to ensure maximum security and usability

1.2 Aim Of The Project

The objective of the project is to design a sophisticated electronic vault system using Raspberry Pi that enhances security by integrating two-factor biometric authentication and leveraging a secure database to store and verify user profiles. This system offers an extra level of protection and remote monitoring capabilities to traditional biometric authentication-based locker systems.

Chapter 2

LITERATURE SURVEY

2.1 Different Methodologies Reviewed

The project is divided in two main parts: Biometric user data enrollment and authentication. The first step in enrollment is to acquire the fingerprint data. Minutiae-based fingerprint recognition is used. It leverages the unique and distinctive features of an individual's fingerprint to establish identity. The algorithm focuses on extracting and analyzing the minutiae points, such as ridge endings and bifurcations, which serve as key reference points for fingerprint matching.

During the enrollment process, the algorithm captures a high-resolution fingerprint image and preprocesses it to enhance the clarity of ridge structures. It then applies advanced image processing techniques to locate and extract minutiae points accurately. The minutiae are typically represented by their coordinates, orientations, and types.

In the verification or identification phase, the algorithm compares the extracted minutiae points from the captured fingerprint with the minutiae template stored during enrollment. Matching algorithms analyze the spatial relationship, orientation, and types of minutiae to calculate a similarity score. If the score exceeds a predefined threshold, the fingerprint is deemed a match, and authentication is granted.

Minutiae-based fingerprint recognition offers several advantages. It provides a high level of accuracy, ensuring reliable identification even in cases of partial or distorted fingerprints. It is also robust against variations caused by factors such as skin conditions or different scanning devices. Furthermore, the compact representation of minutiae templates enables efficient storage and retrieval, making it scalable for large-scale applications.

However, there are challenges associated with minutiae-based fingerprint recognition. The accuracy of the algorithm heavily relies on the quality of the captured fingerprint image. Poor image quality, such as smudges or low-resolution scans, can hinder accurate minutiae extraction and matching. Additionally, there is a risk of spoofing attacks where artificial fingerprints or replicas can deceive the system. Techniques like liveness detection are necessary to counter these vulnerabilities. Haar cascade-based facial recognition is the second step used for detecting and recognizing faces in images and videos. This approach is based on the Haar-like features and the cascade classifier, which allows for efficient and accurate face detection.

In the Haar cascade-based facial recognition, the algorithm uses a set of Haar-like features, which are simple rectangular patterns that can capture various facial characteristics such as edges, lines, and texture variations. These features are calculated at different scales and positions in the input image, creating a feature vector that represents the presence or absence of these patterns.

The cascade classifier, often implemented with machine learning algorithms like AdaBoost, is used to efficiently evaluate these Haar-like features. The classifier applies a series of weak classifiers in a cascaded structure, where each classifier focuses on a specific set of Haar-like features. The cascade classifier employs a combination of thresholding and weighted voting to determine whether a region of the image contains a face or not. This hierarchical approach allows for fast rejection of non-face regions, reducing computational complexity.

During the recognition phase, once a face is detected, additional steps can be performed, such as alignment and feature extraction, to improve the accuracy of face recognition. These steps involve normalizing the detected face region, extracting facial landmarks, and representing the face using a feature vector, such as Eigenfaces or Local Binary Patterns (LBP).

Haar cascade-based facial recognition offers several advantages. It is computationally efficient, enabling real-time face detection and recognition on various devices, including smartphones and surveillance systems. The method is robust to variations in pose, illumination, and facial expressions, making it suitable for diverse real-world scenarios. Additionally, Haar cascade-based algorithms can be trained to detect other objects beyond faces, providing versatility in object recognition tasks.

However, this approach also has limitations. It may struggle with detecting faces under extreme pose variations or occlusions. The accuracy of recognition heavily relies on the quality

of the training dataset and the threshold settings, requiring careful tuning. Furthermore, the Haar cascade-based approach focuses primarily on detecting faces rather than capturing fine-grained facial details, limiting its ability to distinguish between similar individuals.

Chapter 3

DESIGN

3.1 Design Descriptions

The recognition system is designed to integrate fingerprint and facial recognition technologies using the R307 fingerprint sensor module and a webcam. The system consists of the following components:

a. Fingerprint Recognition Module: The R307 fingerprint sensor module is used to capture and process fingerprint data. It provides a reliable and accurate fingerprint recognition capability by utilizing the built-in fingerprint recognition algorithm of the module. The module communicates with the main system through a serial interface.

b. Facial Recognition Module: A webcam is utilized for capturing facial images. The images are processed using a facial recognition algorithm such as Haar cascade-based or deep learning-based methods. The facial recognition module performs face detection, alignment, feature extraction, and matching against enrolled templates.

c. Database Management: The system includes a database to store enrolled templates generated from both fingerprint and facial recognition. The templates are securely stored and associated with the corresponding user's identity.

d. Decision Making: The system compares the matching results from both the fingerprint and facial recognition modules and data stored in the database to make a final authentication decision by running program on raspberry pi.

3.1.1 Block Diagram

In this project, Fingerprint sensor Module (R307) and Webcam are the input devices and takes biometric data (fingerprint and facial features) as input from the user which is then stored in a database. The LCD module is an output device and displays every function as well as interrupts and errors. The servo receives control signals from raspberry pi and actuates the door of the vault to grant access. An external power supply has to be provided for the servo motor. At the authentication stage, the pre-processed image is analysed by raspberry pi to extract unique features of the fingerprint and face, such as the ridge patterns and minutiae points. The extracted features are compared to a database of known fingerprints and facial features to find a match. If a match is found, the system outputs the identity of the person associated with the matching fingerprint and face through the LCD Display. The servo receives control signals from raspberry pi and actuates the door of the vault.

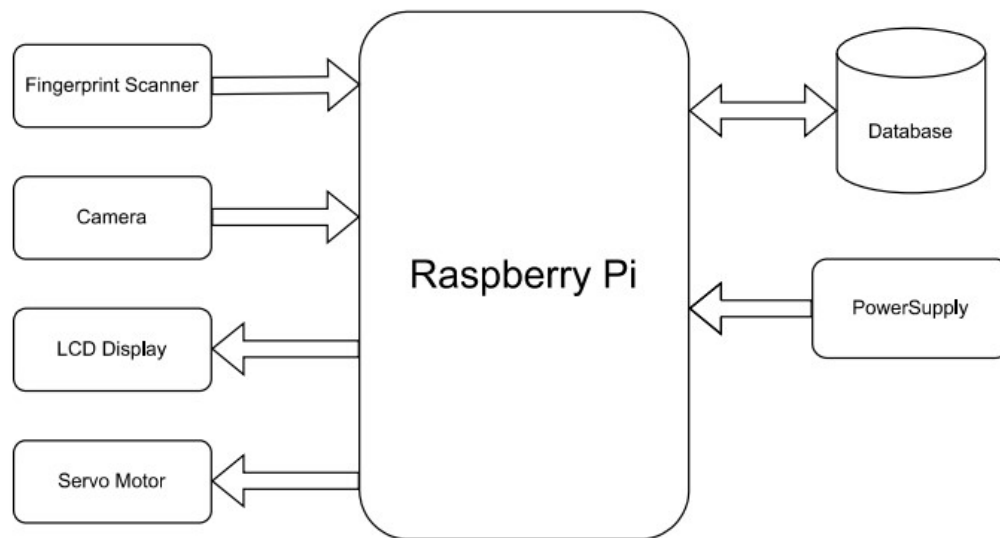


Fig. 3.1: Block diagram of the system

3.1.2 Flow Chart

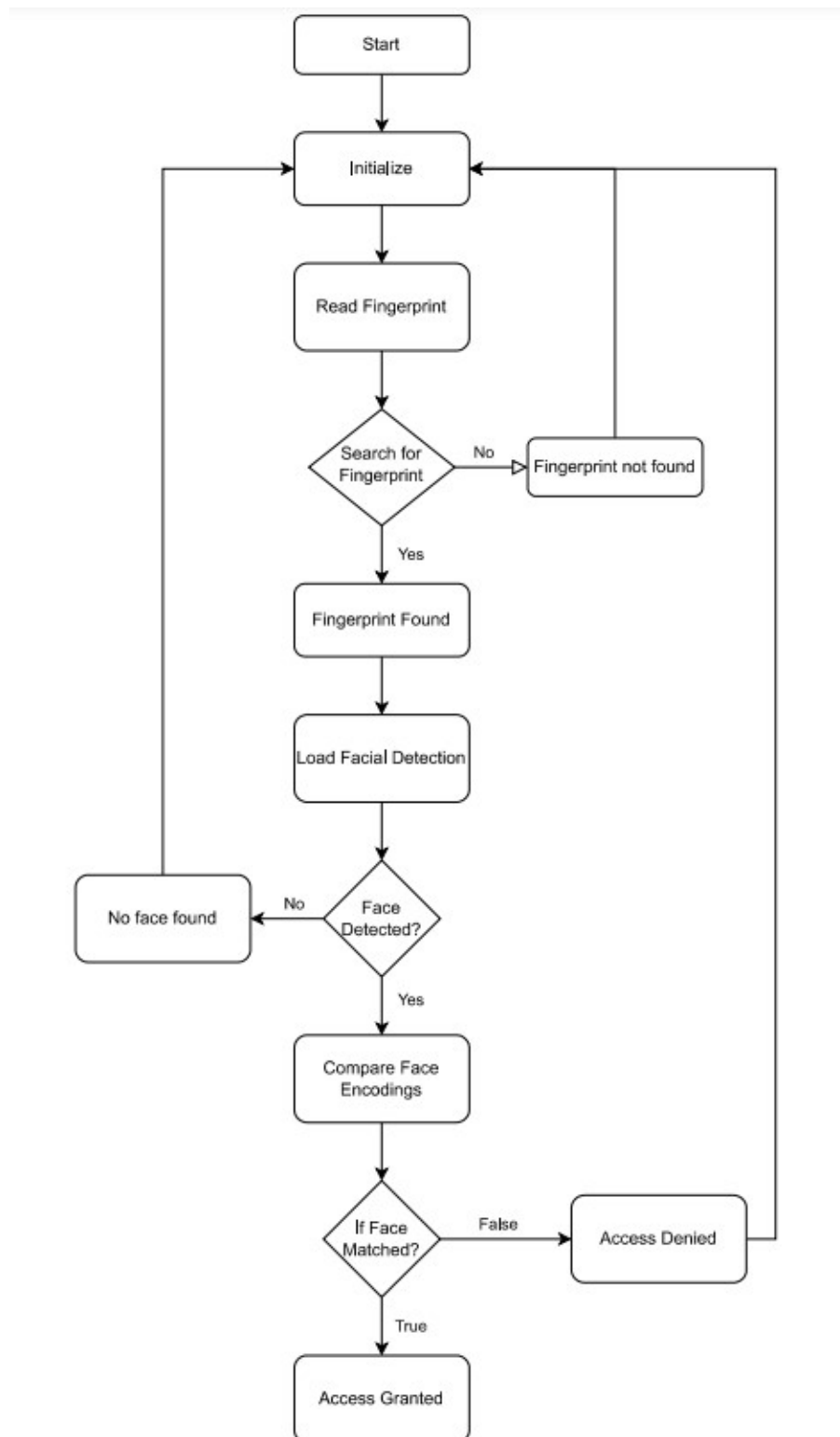


Fig. 3.2: Flow chart

3.1.3 Circuit Diagram

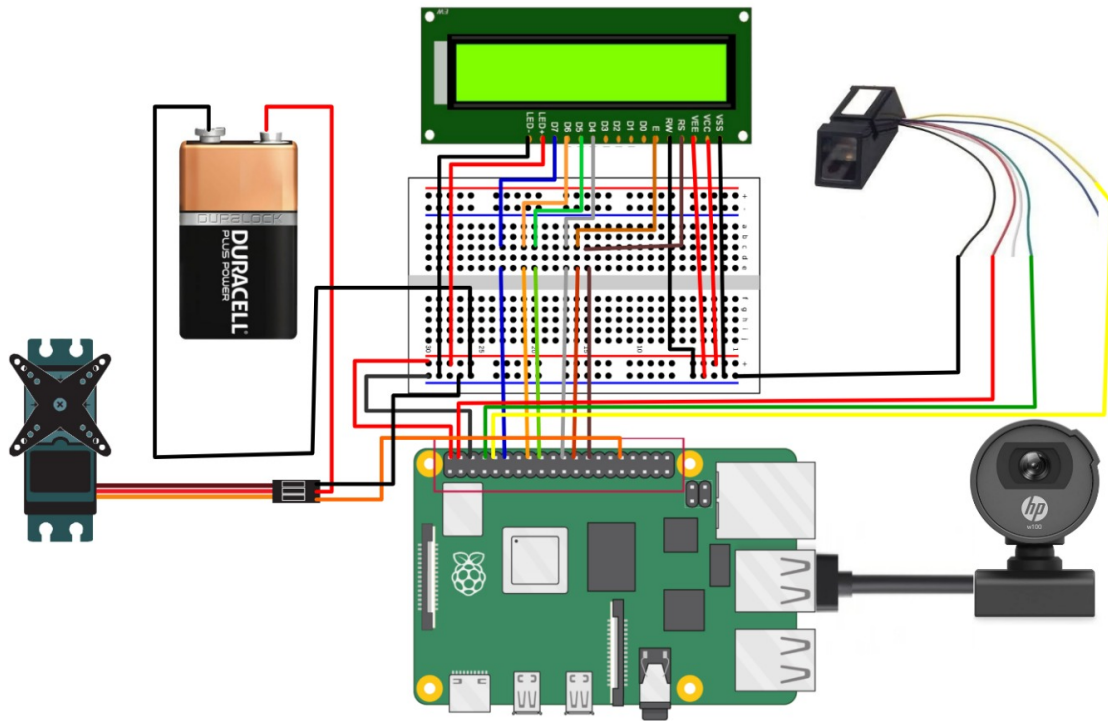


Fig. 3.3: Circuit diagram of the system

3.2 Components List

- (i) Raspberry Pi
- (ii) Fingerprint Module
- (iii) Servo Motor
- (iv) Webcam
- (v) LCD Display
- (vi) Power supply
- (vii) Breadboard
- (viii) Jumper wires

3.2.1 Raspberry Pi 4 Model B

The Raspberry Pi 4 Model B is a powerful single-board computer that offers improved performance and enhanced features compared to its predecessors. It is equipped with a Broadcom BCM2711 SoC, featuring a quad-core ARM Cortex-A72 CPU running at up to 1.5 GHz and a VideoCore VI GPU. In terms of connectivity, the Raspberry Pi 4 Model B includes Gigabit Ethernet, dual-band 2.4 GHz and 5 GHz IEEE 802.11ac wireless LAN, and Bluetooth 5.0.

The Raspberry Pi 4 Model B features two USB 3.0 ports and two USB 2.0 ports, providing ample options for connecting peripherals and external storage devices. It also has two micro HDMI ports, supporting up to 4K resolution, enabling users to connect multiple displays for high-quality visuals. Additionally, it includes a microSD card slot for primary storage and interfaces for display (DSI) and camera (CSI) connections.

With its 40-pin GPIO header, the Raspberry Pi 4 Model B allows for easy interfacing with external devices, sensors, and expansion boards, expanding its capabilities for various projects and applications.

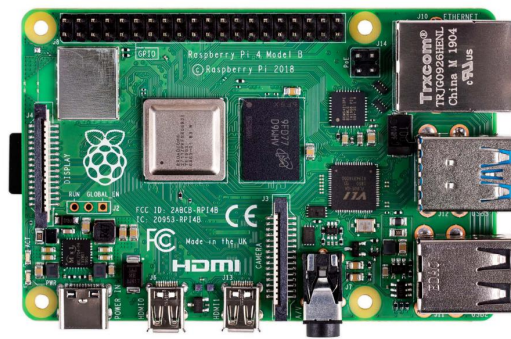


Fig. 3.4: Raspberry Pi 4 Model B

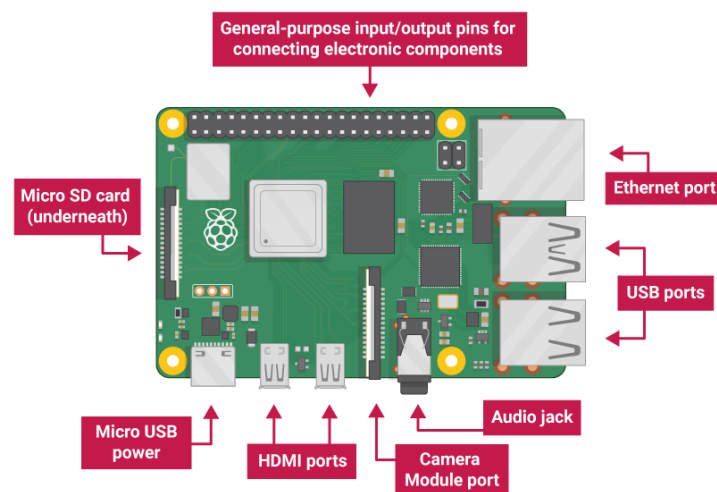


Fig. 3.5: Raspberry Pi 4 Model B Pheriperals

3.2.2 Fingerprint Module

R307 Fingerprint Module consists of optical fingerprint sensor, high speed DSP processor, high performance fingerprint alignment algorithm, high capacity FLASH chips and other hardware and software composition, stable performance, simple structure, with fingerprint entry, image processing, fingerprint matching, search and template storage and other functions. The R307 fingerprint module has two interface TTL UART and USB2.0, USB2.0 interface can be connected to the computer. RS232 interface is a TTL level, the default baud rate is 57600 , can be changed, refer to a communication protocol. Integrated image collecting and algorithm chip together are present in R307. The fingerprint reader can conduct secondary development and can be embedded

into a variety of end products. It has Low power consumption, low cost, small size, excellent performance, professional optical technology and precise module manufacturing techniques. It also has good image processing capabilities and can successfully capture an image up to resolution 500 dpi.



Fig. 3.6: Fingerprint Module

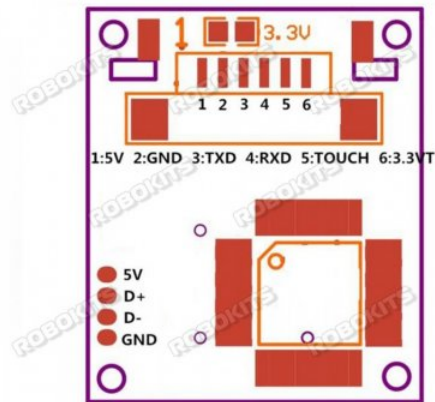


Fig. 3.7: Fingerprint Module Peripherals

3.2.3 Servo Motor

A servo motor is a precision motor that incorporates several key components to provide accurate control over angular position. At its core is a DC motor, which generates rotational motion. This motor is coupled with a gear train mechanism that reduces speed and increases torque output, enabling precise control. The servo motor also includes control circuitry, which interprets input signals and adjusts the motor's operation accordingly. To achieve precise positioning, a position feedback device such as a potentiometer or optical encoder is incorporated. This feedback device provides information about the motor's current position, allowing for precise adjustments. The servo motor also incorporates a driver circuit that translates control signals into appropriate voltage and current levels to drive the motor. With its robust mechanical structure and these essential components, the servo motor excels in applications requiring accurate positioning and motion control. Most of the Servo motors operate from 4.8V to 6.5V, the higher the voltage higher the torque we can achieve, but most commonly they are operated at +5V.



Fig. 3.8: Towerpro SG90 Motor

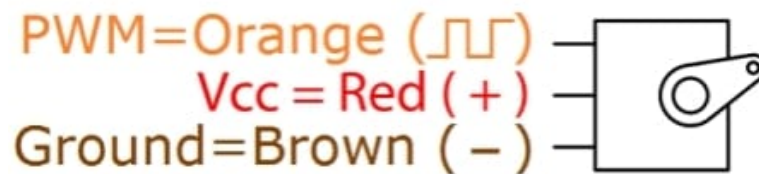


Fig. 3.9: Towerpro SG90 Motor Pinout

3.2.4 Webcam

A Webcam or Web Camera or PC camera is in the product range which is usually a rather low priced segment. Usually Webcam is attached though the USB interface, a few Webcam from some vendors often offers higher quality images using the FireWire or Proprietary interface. Most Webcams have only a limited resolution like 640 \times 480 pixels (VGA). This resolution is usually sufficient for intended uses. Furthermore Webcams are a cheaper substitute for the expensive CCD cameras in astrophotography too, but Webcam are also suitable for taking images through special instruments like a microscope. Some manufacturers also design Webcam for using it as network cameras, which directly get connected via Internet protocol or via Ethernet or WLAN. HP w100

480P HD Web Camera enables the user to experience good quality video calls and supports video quality up to 640x480 pixels. It comes with a 60° wide angle view for a stunning visual experience.



Fig. 3.10: hp w100 Webcam

3.2.5 LCD Display

An electronic device that is used to display data and the message is known as LCD 16x2. As the name suggests, it includes 16 columns 2 rows so it can display 32 characters ($16 \times 2 = 32$) in total every character will be made with 5×8 (40) Pixel Dots. So the total pixels within this LCD can be calculated as 32×40 otherwise 1280 pixels. The registers used in LCD are two types like data register command register. The register can be changed by using the RS pinout. If we set '0' then it is a command register and if it is '1' then it is a data register. The main function of the command register is to save instructions illustrated on LCD. The data register is used to save the data to exhibit on the LCD. Once we transmit data to LCD, then it shifts to the data register to process the data. If we fix the register value at one, then the data register will start working.

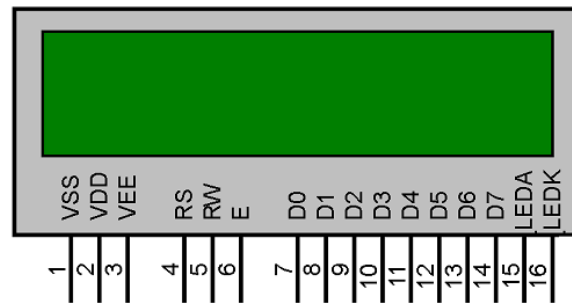


Fig. 3.11: 16x2 LCD Display Pinout

3.2.6 Power Supply

A 9V battery provide as an external power supply for the servo motor. The capacity of a 9V power supply battery determines its ability to store and deliver charge, typically measured in milliampere-hours (mAh) or ampere-hours (Ah). With its compact design and reliable voltage output, the 9V power supply battery serves as a convenient and portable power source for numerous applications, making it a popular choice among electronic enthusiasts and professionals alike.



Fig. 3.12: 9v Power Supply

3.2.7 Breadboard

A breadboard is an essential tool for electronics prototyping and circuit design. It consists of a plastic board with interconnected metal strips or tracks that allow for easy creation of temporary electrical connections. The board features terminal strips for power and ground connections and

a grid of contact points where components can be inserted. The metal tracks provide internal connections, enabling components to be easily connected without the need for soldering.



Fig. 3.13: Breadboard

3.2.8 Jumper Wire

Jumper wires are essential components in electronics and circuitry for creating temporary electrical connections. They consist of a conductive material, typically copper, encased in an insulating material such as plastic. Jumper wires are flexible, allowing them to be easily routed and connected between different points on a breadboard, circuit board, or electronic components. They come with connectors on each end, such as male pins or female sockets, enabling them to establish connections with other components or contact points.



Fig. 3.14: Jumper Wires

3.3 Estimated Cost

COMPONENTS	UNIT	PRICE	COST
Raspberry Pi 4 Model B	1	8254	8254
Webcam hp w100	1	400	400
Fingerprint Module R307	1	2400	2400
Towerpro SG90 Servo motor	1	175	175
External Power supply	1	45	45
Jumper Wires	45	5	225
16×2 LCD Display	1	170	170
Breadboard	1	75	75
Connecting Wires	1	25	25
TOTAL COST	RS.11769		

Fig. 3.15: Cost Estimation

3.4 Software

The Raspberry Pi OS is a Linux-based operating system specifically designed for Raspberry Pi devices. It provides essential drivers and software compatibility for the R307 fingerprint sensor and webcam. Python is a versatile programming language widely used for Raspberry Pi projects. It offers an extensive range of libraries and modules for image processing, computer vision, and database connectivity. OpenCV is a powerful open-source computer vision library that provides various functions and tools for image and video processing. It offers features such as image capture, preprocessing, feature extraction, and facial recognition algorithms. A fingerprint recognition algorithm based on minutiae extraction will be implemented using Python and OpenCV. This algorithm will analyze the captured fingerprint images from the R307 sensor, extract unique minutiae points, and perform matching or identification. OpenCV's facial recognition algorithms LBPH (Local Binary Patterns Histograms), will be used to detect and analyze faces captured by the webcam. The algorithm will extract facial features and perform matching or identification tasks.

Commands for Installing OpenCV :

- i. `sudo apt-get install python3-pip`
- ii. `pip install numpy`
- iii. `apt list python*opencv*`
- iv. `apt show python3-opencv`

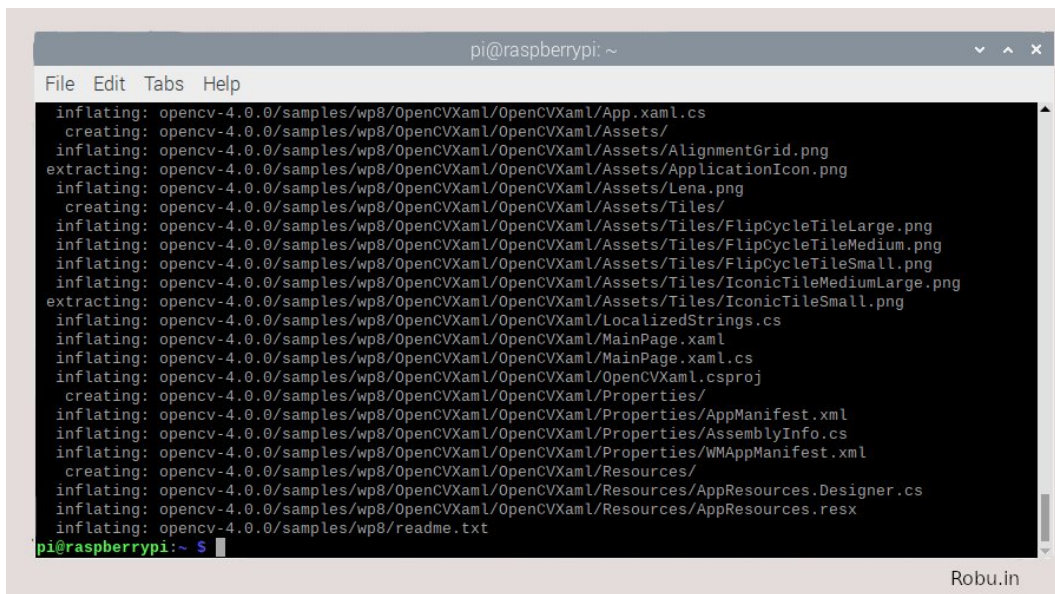


Fig. 3.16: Installing OpenCV

A MySQL database will be utilized to store registered fingerprint templates, associated facial data, and user information. Python's MySQL connector library will facilitate the interaction between the recognition system and the database, enabling storage, retrieval, and management of data. The system uses MySQL queries to access the database and retrieve or update the information.

Commands for Installing MySQL database :

1. sudo apt update
2. sudo apt upgrade
3. sudo apt install mariadb-server
4. sudo mysql-secure-installation
5. sudo mysql -u root -p

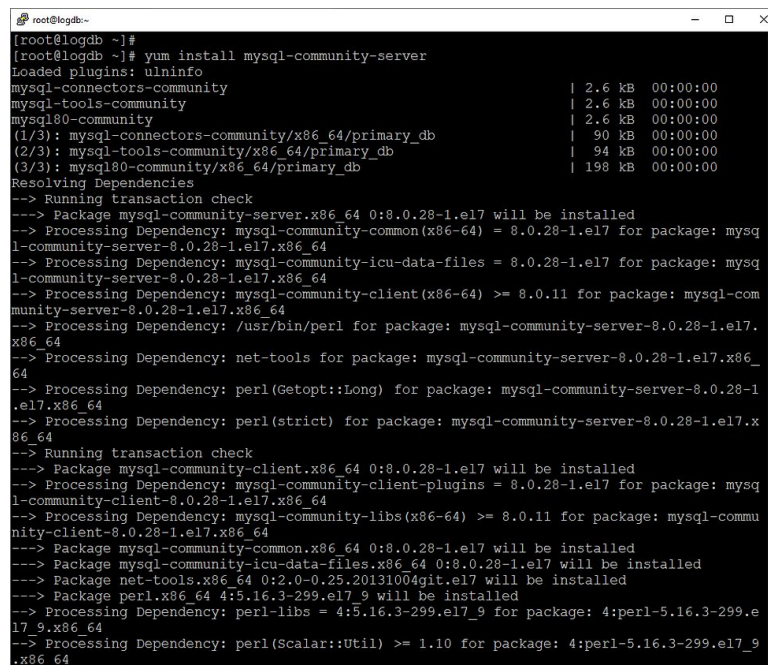
Creating a MySQL Database and User

1. sudo mysql -u root -p
2. CREATE DATABASE exampleadb;

3. CREATE USER 'exampleuser'@'localhost' IDENTIFIED BY 'pimylifeup';
4. GRANT ALL PRIVILEGES ON exampledb.* TO 'exampleuser'@'localhost';
5. FLUSH PRIVILEGES;

Installing the PHP MySQL Connector

1. sudo apt install php-mysql



```

[root@logdb ~]#
[root@logdb ~]# yum install mysql-community-server
Loaded plugins: ulninfo
mysql-connectors-community | 2.6 kB 00:00:00
mysql-tools-community | 2.6 kB 00:00:00
mysql80-community | 2.6 kB 00:00:00
(1/3): mysql-connectors-community/x86_64/primary_db | 90 kB 00:00:00
(2/3): mysql-tools-community/x86_64/primary_db | 94 kB 00:00:00
(3/3): mysql80-community/x86_64/primary_db | 193 kB 00:00:00
Resolving Dependencies
--> Running transaction check
--> Package mysql-community-server.x86_64 0:8.0.28-1.el7 will be installed
--> Processing Dependency: mysql-community-common(x86-64) = 8.0.28-1.el7 for package: mysql-community-server-8.0.28-1.el7.x86_64
--> Processing Dependency: mysql-community-icu-data-files = 8.0.28-1.el7 for package: mysql-community-server-8.0.28-1.el7.x86_64
--> Processing Dependency: mysql-community-client(x86-64) >= 8.0.11 for package: mysql-community-server-8.0.28-1.el7.x86_64
--> Processing Dependency: /usr/bin/perl for package: mysql-community-server-8.0.28-1.el7.x86_64
--> Processing Dependency: net-tools for package: mysql-community-server-8.0.28-1.el7.x86_64
--> Processing Dependency: perl(Getopt::Long) for package: mysql-community-server-8.0.28-1.el7.x86_64
--> Processing Dependency: perl(strict) for package: mysql-community-server-8.0.28-1.el7.x86_64
--> Running transaction check
--> Package mysql-community-client.x86_64 0:8.0.28-1.el7 will be installed
--> Processing Dependency: mysql-community-client-plugins = 8.0.28-1.el7 for package: mysql-community-client-8.0.28-1.el7.x86_64
--> Processing Dependency: mysql-community-libs(x86-64) >= 8.0.11 for package: mysql-community-client-8.0.28-1.el7.x86_64
--> Package mysql-community-common.x86_64 0:8.0.28-1.el7 will be installed
--> Package mysql-community-icu-data-files.x86_64 0:8.0.28-1.el7 will be installed
--> Package net-tools.x86_64 0:2.0-0.25.20131004git.el7 will be installed
--> Package perl.x86_64 4:5.16.3-299.el7_9 will be installed
--> Processing Dependency: perl-libs = 4:5.16.3-299.el7_9 for package: 4:perl-5.16.3-299.el7_9.x86_64
--> Processing Dependency: perl(Scalar::Util) >= 1.10 for package: 4:perl-5.16.3-299.el7_9.x86_64

```

Fig. 3.17: Installing MySQL Database

Overall, the combination of these software technologies provides a powerful and versatile platform for developing the face and fingerprint recognition system.

3.5 Implementation

We have designed the work with the use of several components. The below image shows the interfacing of all components of our system.

Installing the necessary software: The Raspberry Pi OS is installed and made up-to-date. Install Python and the OpenCV library to facilitate image processing and computer vision tasks.

Next, install the MySQL database server and the MySQL Connector library for Python to enable interaction with the database. Finally, execute the vault system by running the Python scripts on the Raspberry Pi.

Connecting the hardware components: Connect the R307 fingerprint sensor to the Raspberry Pi by establishing the required power, ground, and serial communication connections. Connect the webcam to one of the USB ports on the Raspberry Pi. The servo motor is connected to GPIO pins via an external supply.

Creating the database: The system requires a database to store user biometrics. Within this database, create a table to store user id, fingerprint and facial data. Define the appropriate table structures, including columns for storing fingerprint and facial templates, user IDs, and other relevant information.

Developing the face and fingerprint recognition algorithms: Develop the recognition system by writing Python scripts that capture fingerprint and facial data, process images using OpenCV, implement the fingerprint and facial recognition algorithms, and communicate with the MySQL database using the MySQL Connector library. And finally to provide servo motor with the control signals upon successful authentication.

User interface: The user can interact with the system through commands displayed via LCD at times of successful/unsuccessful authentication. The system is connected to an external monitor, mouse and a keyboard at the time of enrollment (Raspbian OS).

Integrating the components: The final step is to integrate the various components of the system, including the face and fingerprint recognition algorithms, the user interface, and the database.

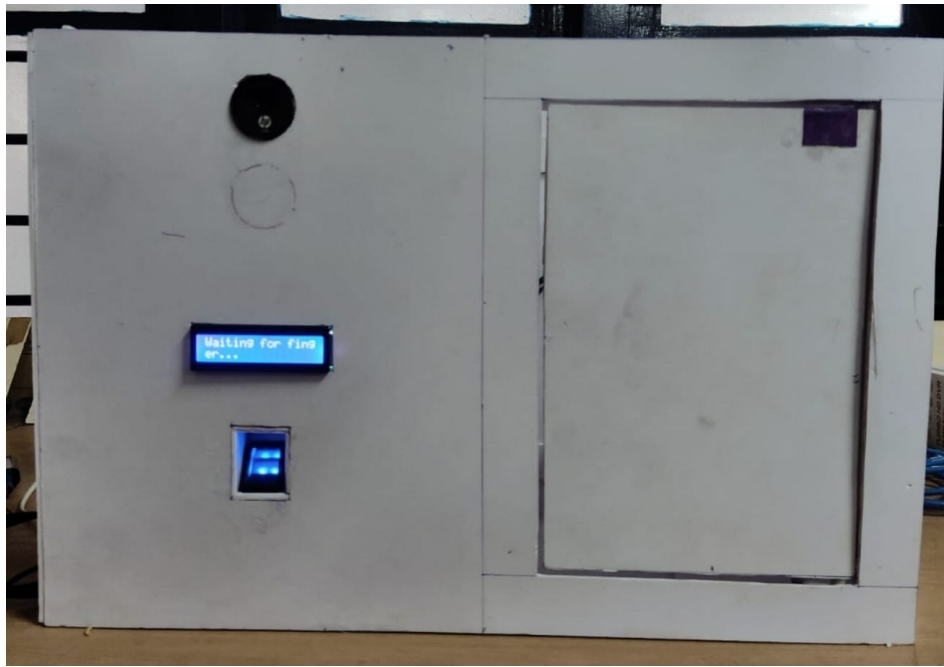


Fig. 3.18: Full Hardware setup (front view)



Fig. 3.19: Full Hardware setup (top view)

3.6 Work Plan

The work plan describes the about of work done by each members in a given interval of time. Our work was equally divided and everyone played a equal role the work plan had 4 phases phase 1, phase 2, phase 3, phase 4

Phase 1: The first phase consisted of brainstorming, project idea submission, research on the topic, power point preparation and our 1st presentation.

Phase 2: The second phase consisted of interaction with our project guide to discuss the constraints of our project according to that the components where selected and a brief cost estimation and optimum conditions of the suitable variant were researched. It also consisted of submission of report upto 2 chapter and a presentation was held in that phase also.

Phase 3: During the third phase the power point modifications were done and then presented after that each selected components where selected and tested and the values of each where briefly analysed, During this phase we were asked to submit the report so its preparation was also done. Here All the technical works and preparation of the system body was also done, everyone contributed in something and did there best.

Phase 4: The final phase or the 4th phase consisted of final report submission, final presentation and other works like managing the model, left over works etc.

Chapter 4

RESULT

The Electronic Vault with 2 factor Authentication based on Raspberry Pi, utilizing the R307 fingerprint sensor, webcam, and MySQL database, delivers highly accurate and efficient user identification and authentication along with an added advantage of remote access. The result of our project is depicted in the form a table below:

Input	Output
Placing the finger	Reading Fingerprint
If the pressed fingerprint data exists	Fingerprint found
If the pressed fingerprint data doesn't exist	Fingerprint not found
Face scanning	Loading face detector
If the scanned face data exists	Face detected
If the scanned face data doesn't exist	Face not detected
If both the user data gets matched	Access granted

Fig. 4.1: Final output

Chapter 5

CONCLUSION

In conclusion, the locker system based on Raspberry Pi, utilizing fingerprint and facial recognition with the R307 sensor, webcam, and MySQL database, offers a highly secure and efficient solution for access control and user authentication. By combining the power of Raspberry Pi with advanced biometric recognition technologies, the system ensures reliable and accurate identification of authorized users. The integration of the R307 sensor and webcam enables seamless capture of fingerprint and facial data, while the MySQL database provides secure storage and retrieval of user information and providence to remote access. This locker system enhances security measures by eliminating the need for traditional keys or passwords, reducing the risk of unauthorized access. It offers convenience and ease of use, allowing authorized users to access lockers with a simple scan of their fingerprint or face. With its robust performance and reliable recognition capabilities, the locker system based on Raspberry Pi provides a cutting-edge solution for secure storage and access control in various settings, such as offices, gyms, and personal storage facilities.

5.1 Future Scope

1. **Multi-factor Authentication:** The system can be expanded to include additional authentication factors, such as voice recognition or iris scanning, for even stronger security measures. Implementing multi-factor authentication can further enhance the system's robustness and make it even more resistant to unauthorized access.

2. **Cloud Integration:** Integrating the locker system with cloud-based storage solutions can

provide added convenience and accessibility. Users could remotely access and manage their locker contents, allowing for greater flexibility and convenience in storing and retrieving personal belongings.

3. **Mobile Application Integration:** Developing a mobile application that interfaces with the locker system would enable users to control and monitor their lockers using their smartphones. This would provide a seamless and user-friendly experience, allowing users to easily manage locker access and receive notifications on their mobile devices.

4. **Advanced Analytics:** Implementing advanced analytics algorithms on the collected data can provide valuable insights and patterns. For example, the system could analyze usage patterns and trends to optimize locker allocation and improve overall system efficiency.

5. **Artificial Intelligence and Machine Learning:** Leveraging AI and machine learning techniques can enhance the system's recognition capabilities. By training the system on a large dataset of fingerprints and facial images, it can continuously improve its accuracy and adapt to varying environmental conditions and user profiles.

6. **Integration with IoT Devices:** Integrating the locker system with Internet of Things (IoT) devices can provide additional functionalities. For example, integrating with smart home devices could enable users to unlock their lockers through voice commands or control access remotely.

REFERENCES

- [1] Gian Luca Marcialis, Fabio Roli, Luca Didaci, "Personal identity verification by serial fusion of fingerprint and face matchers", *Pattern Recognition* 42 (2009) 2807-2817.
- [2] G. Zhao and M. Pietikainen, "Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions," *Pattern Analysis and Machine Intelligence*, 29(6): 915-928, 2007.
- [3] Jun Zhou, Guangda Su, Chunhong Jiang, Yafeng Deng, and Congcong Li, "A face and fingerprint identity authentication system based on multi-route detection Neurocomputing, Elsevier Science Publishers, Volume 70, Issue 4-6, Pages: 922-931, 2007.
- [4] Chitresh Saraswat Amit Kumar. "An Efficient Automatic Attendance System using Fingerprint Verification Technique Chitresh Saraswat et al (UCSE) *International Journal on Computer Science and Engineering* Vol. 02, No. 02, 2010, 264-269
- [5] Manvjeet Kaur, Mukhwinder Singh, Akshay Girdhar, and Parvinder S. Sandhu, "Fingerprint Verification System using Minutiae Extraction Technique", *World Academy of Science, Engineering and Technology* 46 2008
- [6] Sanqiang Zhao and Yongsheng Gao, "Establishing Point Correspondence using Multidirectional Binary Pattern for Face Recognition", 978-1-4244-2175-6/08/2008 IEEE
- [7] <https://www.techiesms.com/product/r307-optical-fingerprint-reader-sensor-module/>
- [8] <https://www.amazon.in/HP-Webcam-Wide-Angle-Calling-Microsoft/dp/B08FTFXNNB>
- [9] <https://www.thingbits.in/products/standard-lcd-16x2-display>

[10] <https://circuitdigest.com/article/16x2-lcd-display-module-pinout-datasheet>

[11] <https://core-electronics.com.au/guides/face-identify-raspberry-pi/>

[12] <https://pimylifeup.com/raspberry-pi-mysql/>

Appendix A

PROGRAM CODE

A.1 DATA ENROLLMENT

```
# Establish a connection to the MySQL database
db = pymysql.connect(host='localhost', user='karthik', password='karthik',
                     database='authentication')

curs = db.cursor()

# Create an instance of PyFingerprint class for fingerprint scanner
communication

f = PyFingerprint('/dev/ttyS0', 57600, 0xFFFFFFFF, 0x00000000)

# Verify the fingerprint sensor password
if not f.verifyPassword():
    raise ValueError('The given fingerprint sensor password is wrong!')

print('Waiting for finger...')

# Wait for the finger to be read
```

```
while f.readImage() == False:
    pass

print('Remove finger...')
time.sleep(2)

# Convert the finger image to characteristics and search for a template
f.convertImage(0x01)
result = f.searchTemplate()
positionNumber = result[0]

if positionNumber >= 0:
    print('Fingerprint already exists!')
    print('Template position: ' + str(positionNumber))
    temp = f.downloadCharacteristics()
    print(temp)

# Prompt the user to enter the ID and name
iD = input('Enter ID: ')
name = input('Enter your name: ')

# Convert the characteristics to a SHA-256 hash
chara = str(f.downloadCharacteristics()).encode('utf-8')
fin = hashlib.sha256(chara).hexdigest()

# Store the fingerprint template
f.storeTemplate()
result = f.searchTemplate()
positionNumber1 = result[0]
print('Enrolled data position: ' + str(positionNumber1))
```

```
print('Finger enrolled successfully!')
time.sleep(2)

# Define the path for storing the facial recognition dataset
downloads_path = os.path.expanduser('~/.facial_rec/dataset')

# Create a folder for the user if it doesn't exist
folder_path = os.path.join(downloads_path, iD)
if not os.path.exists(folder_path):
    os.makedirs(folder_path)
    print(f'Created folder: {folder_path}')
else:
    print(f'Folder already exists: {folder_path}')

# Open the video capture device
cam = cv2.VideoCapture(0)

# Create a named window for displaying the video feed
cv2.namedWindow("Press space to take a photo", cv2.WINDOW_NORMAL)
cv2.resizeWindow("Press space to take a photo", 500, 300)

count = 0

# Start capturing video frames
while True:
    ret, frame = cam.read()
    if not ret:
        print("Failed to grab frame")
        break
```

```
cv2.imshow("Press space to take a photo", frame)

k = cv2.waitKey(1)
if k % 256 == 27:
    # ESC pressed, exit the loop
    print("Escape hit, closing...")
    break
elif k % 256 == 32:
    # SPACE pressed, save the frame as an image
    file_path = os.path.join(folder_path, f'img' + str(count) + '.jpg')
    cv2.imwrite(file_path, frame)
    print("{} written!".format(file_path))
    count += 1

# Release the video capture device and close the window
cam.release()
cv2.destroyAllWindows()

# Process facial recognition for the images in the folder
folder_path = os.path.join("dataset", iD)

print("[INFO] Start processing faces...")
imagePaths = list(paths.list_images(folder_path))

# Initialize the list of known encodings and known IDs
knownEncodings = []
knownNames = []

# Loop over the image paths
for (i, imagePath) in enumerate(imagePaths):
```



```
print("[INFO] Processing image {}/{}".format(i + 1, len(imagePaths)))

# Extract the person ID from the image path
ID = imagePath.split(os.path.sep)[-2]

# Load the input image and convert it from RGB to dlib ordering (RGB)
image = cv2.imread(imagePath)
rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Detect the (x, y)-coordinates of the bounding boxes
#Corresponding to each face in the input image
boxes = face_recognition.face_locations(rgb, model="hog")

# Compute the facial embedding for the face
encodings = face_recognition.face_encodings(rgb, boxes)

# Loop over the encodings
for encoding in encodings:
    # Add each encoding + ID to our set of known IDs and encodings
    knownEncodings.append(encoding)
    knownNames.append(ID)

# Dump the facial encodings + IDs to disk
print("[INFO] Serializing encodings...")
data = {"encodings": knownEncodings, "ID": knownNames}
f = open("encodings.pickle", "wb")
f.write(pickle.dumps(data))
f.close()

# Load the serialized encodings and insert them into the database
```

```
with open('encodings.pickle', 'rb') as h:
    data = pickle.load(h)
data_bin = pickle.dumps(data)
query = "INSERT INTO data11 (id, name, fing_encod, temp_pos, img_encod)
VALUES (%s, %s, %s, %s, %s)"
curs.execute(query, (iD, name, fin, positionNumber1, data_bin))
db.commit()
db.close()
```

A.2 DELETING DATA

```
# Initialize a variable
pos = 0

# Establish a connection to the MySQL database
db = pymysql.connect(host='localhost', user='karthik', password='karthik',
    database='authentication')
curs = db.cursor()

# Create an instance of PyFingerprint class for fingerprint scanner comm.
f = PyFingerprint('/dev/ttyS0', 57600, 0xFFFFFFFF, 0x00000000)

while True:
    # Prompt the user to enter the ID of the user to be deleted
    iD = input("Enter ID to be deleted: ")
```

```
# Execute a SQL query to check if the user with the specified
ID exists in the database
curs.execute("SELECT * FROM data11 WHERE id=%s", iD)
row = curs.fetchone()

if row is None:
    print('User not found')
else:
    break

pos = row[3] # Retrieve the position of the user's fingerprint template
#from the database
pos = int(pos) # Convert the position to an integer

# Delete the fingerprint template from the fingerprint scanner
f.deleteTemplate(pos)

# Construct a SQL query to delete the user from the database
#based on the ID
q = "DELETE FROM data11 WHERE id = %s"
curs.execute(q, iD)

# Commit the changes to the database
db.commit()

# Close the database connection
db.close()

# Delete the folder with the given ID
downloads_path = os.path.expanduser('~/.facial_rec/dataset')
```

```
folder_path = os.path.join(downloads_path, iD)

if os.path.exists(folder_path):
    shutil.rmtree(folder_path) # Delete the folder and its contents
    print(f'Deleted ID: {iD}')
else:
    print(f'{iD} does not exist')

print('Deleted successfully')
```

A.3 AUTHENTICATION

```
import hashlib
import time
from pyfingerprint.pyfingerprint import PyFingerprint
from RPi import GPIO
import pymysql
import cv2
import os
from imutils import paths
import face_recognition
from imutils.video import VideoStream
from imutils.video import FPS
import imutils
import pickle
import datetime
import RPi.GPIO as GPIO
from time import sleep
```

```
GPIO.setwarnings(False)
from RPLCD.gpio import CharLCD

# Initialize the LCD display
lcd = CharLCD(pin_rs=7, pin_e=8, pins_data=[25, 24, 23, 18],
              numbering_mode=GPIO.BCM, cols=16, rows=2, dotsize=8,
              charmap='A02', auto_linebreaks=True)

# Initialize the fingerprint scanner
fing = PyFingerprint('/dev/ttyS0', 57600, 0xFFFFFFFF, 0x00000000)

# Start the video stream
vs = VideoStream(0).start()

time.sleep(2.0)

# Initialize the servo motor
GPIO.setmode(GPIO.BCM)
GPIO.setup(12, GPIO.OUT)
pwm = GPIO.PWM(12, 50)
pwm.start(0)

# Function to set the angle of the servo motor
def setangle(angle):
    duty = angle / 18 + 2
    GPIO.output(12, True)
    pwm.ChangeDutyCycle(duty)
    sleep(1)
    pwm.ChangeDutyCycle(0)
```

```
# Verify the fingerprint sensor password
if not fing.verifyPassword():
    raise ValueError('[INFO]The given fingerprint sensor is wrong !')

while True:
    # Establish a connection to the MySQL database
    db = pymysql.connect(host='localhost', user='karthik',
        password='karthik', database='authentication')
    curs = db.cursor()
    now = datetime.datetime.now()

    today = now.strftime("%d-%m-%y")

    # Display initial message on the LCD
    lcd.clear()
    lcd.write_string('ELECTRONIC VAULT      2FA')
    time.sleep(5)
    lcd.clear()
    lcd.write_string('Waiting for finger...')
    print('[INFO]Waiting for finger...')

    # Wait for the finger to be read
    while fing.readImage() == False:
        pass

    # Convert the finger image to characteristics
    # And Search for a template
    fing.convertImage(0x01)
```

```
result = fing.searchTemplate()
positionNumber = result[0]

if positionNumber == -1:
    lcd.clear()
    lcd.write_string('Fingerprint not found')
    print("[INFO]Fingerprint not found")
    time.sleep(2)
    vs.stop()
else:
    lcd.clear()
    lcd.write_string('Fingerprint found')
    print("[INFO]Fingerprint found")
    time.sleep(2)

# Load the fingerprint template
#convert characteristics to SHA-256 hash
fing.loadTemplate(positionNumber, 0x01)
chara = str(fing.downloadCharacteristics(0x01)).encode('utf-8')
find = hashlib.sha256(chara).hexdigest()

# Query the database with the fingerprint hash
query = "SELECT * FROM data11 WHERE fing_encod = %s"
curs.execute(query, find)
db.commit()
row = curs.fetchone()

iD = row[0]
name = row[1]
```

```
print("Fingerprint found for ID : ", row[0])
print("Name : ", row[1])

# Display ID and name on the LCD
lcd.clear()
lcd.cursor_pos = (0, 0)
lcd.write_string('ID : {}'.format(id))
lcd.cursor_pos = (1, 0)
lcd.write_string('Name : {}'.format(name))

time.sleep(2)

face = row[4]
data = pickle.loads(face)

# Save the facial encodings to a pickle file
with open('data_enc.pickle', 'wb') as f:
    pickle.dump(data, f)
encodingsP = "data_enc.pickle"

lcd.clear()
lcd.write_string('loading face detector...')
print("[INFO] loading face detector...")

# Load the facial encodings from the pickle file
data = pickle.loads(open(encodingsP, "rb").read())

time.sleep(2.0)

while True:
```



```
        frame = vs.read()
        frame = imutils.resize(frame, width=500)
        boxes = face_recognition.face_locations(frame)

    if len(boxes) == 0:
        print('No face found\n')
        lcd.clear()
        lcd.write_string('No face found')
        time.sleep(2)
        break

    encodings = face_recognition.face_encodings(frame, boxes)

    for encoding in encodings:
        matches =face_recognition.compare_faces(data["encodings"],
                                                encoding)

    if True in matches:
        print("Face Matched for ID : ", iD)
        lcd.clear()
        lcd.write_string('Face matched for ID : {}'.format(iD))
        time.sleep(2)
        print("Access granted\n")
        lcd.clear()
        lcd.write_string('Access granted')
        time.sleep(2)

    # Insert the access record into the database
    sql = "INSERT INTO record1 (id, name, date, time)
          VALUES (%s, %s, %s, %s)"
```

```
curs.execute(sql, (iD, name, today, now.time()))
db.commit()

pwm.start(0)

setangle(0)

setangle(90)

time.sleep(5)

else:
    lcd.clear()
    lcd.write_string('Face not match')
    print("Face not match")
    time.sleep(2)
    lcd.clear()
    lcd.write_string('Access denied')
    print("Access denied\n")
    time.sleep(2)

break

setangle(0)
cv2.destroyAllWindows()
curs.close()
db.close()
f.close()
```