



**NANYANG TECHNOLOGICAL UNIVERSITY**

**SCSE23-0312: IOT - Development of Home  
Automation system**

Keith Lee Mun Wei

Project Supervisor: A/P Lau Chiew Tong

School of Computer Science and Engineering

Academic Year 2023/2024

## **NANYANG TECHNOLOGICAL UNIVERSITY**

### **SCSE23-0312: IOT - Development of Home Automation system**

Submitted in Partial Fulfilment of the Requirements

for the Degree of Bachelor of Engineering in Computer Science  
of the Nanyang Technological University

by

Keith Lee

School of Computer Science and Engineering

Academic Year 2023/2024

# Abstract

The realm of home automation has become increasingly prevalent with technological advancements and a growing user base. This field, spanning from security applications to remote monitoring, offers a diverse array of solutions for users even when they are away from home.

This project centres on the development of a home security system. This involves the implementation of both system activation and cameras to offer security information, ensuring assurance regarding home activities. The execution of this system is facilitated through an application, enabling users to remotely control and monitor their home security measures anywhere they want.

# Acknowledgement

I offer my sincere gratitude to Associate Professor (A/P) Lau Chiew Tong, my supervisor, for providing unwavering guidance and invaluable advice during the entire course of my project and for the opportunity to contribute to the project as it has allowed me to explore and gain more knowledge in the field of IoT.

I would like to thank Nanyang Technological University and my FYP supervisor for granting me the opportunity to learn more about Home Automation Systems and to be able to develop it as well.

I would also like to credit the immense contributions of the open-source community and Ai tools such as ChatGPT which I used to help me with my project.

# Table of Contents

Abstract.....	iv
Acknowledgement .....	v
Chapter 1: Introduction .....	ix
1.1    Background .....	ix
1.2 Objective and scope .....	ix
Chapter 2: Literature review .....	xi
2.1 Existing products and comparisons .....	xi
1.    Yew F11 Facial Recognition Digital Lock .....	xi
2.    Samsung Digital Door Lock.....	xi
3.    Bosch FU780 Digital Lock .....	xii
Chapter 3: Project Overview .....	xv
3.1 Brief Project Description .....	xv
3.2 Main Hardware Components.....	xv
3.2.1 Raspberry pi 4 .....	xv
3.2.2 Raspberry pi camera V2 .....	xv
3.2.3 Servo Motor .....	xvi
3.2.4 Ultrasonic sensor (HC-SR04) .....	xvi
3.2.5 Led bulb.....	xvii
3.2.6 Breadboard .....	xviii
Chapter 4: Implementation .....	xviii
4.1 Application .....	xviii
4.1.1    Login Page .....	xviii
4.1.2    Home Page .....	xix
4.1.3    Camera View Page .....	xxii
4.2    Sensor and Facial Recognition System.....	xxiii
4.2.1    Training the model .....	xxiii
4.2.2    Ultrasonic sensor code.....	xxiv
4.2.3    Facial Recognition file .....	xxiv
4.2.4    Servo motor file .....	xxvi
4.3    Google's Realtime Firebase Database.....	xxvi

4.3.1	Firestore .....	xxvi
4.3.2	Storage.....	xxvii
4.3.3	Authentication.....	xxvii
4.4	Web communication.....	xxvii
4.4.1	Streaming Handler Class .....	xxviii
4.4.2	Streaming Object Class .....	xxx
4.4.3	StreamingServer Class.....	xxx
4.4.4	Mobile Application Web View .....	xxxi
Chapter 5: Conclusion .....		xxxii
Chapter 6: Further improvements .....		xxxiii
6.1	Low light conditions .....	xxxiii
6.2	Scalability .....	xxxiii
6.3	Rechargeable battery.....	xxxiii
References .....		xxxiii

# Table of Figures

Figure 4 Yew F11 Digital Lock	xi
Figure 1 <i>Samsung Digital Door Lock</i>	xii
Figure 3 Bosch FU780 Digital Lock	xii
Figure 2 Philips Easykey 702 Digital Lock	xiii
Figure 5 Raspberry Pi 4 (Rpi)	xv
Figure 6 Raspberry pi camera V2	xv
Figure 7 Servo Motor	xvi
Figure 8 Ultrasonic sensor	xvi
Figure 9 Led	xvii
Figure 10 Breadboard	xviii
Figure 11 Login page	xix
Figure 12 Home page	xx
Figure 13 Selected Member dialog	xx
Figure 14 Add member/user dialog	xxi
Figure 15 Add administrator	xxi
Figure 16 Camera View page	xxii
Figure 17 alertTextView dialog	xxiii
Figure 18 face_recognition functions code	xxiii
Figure 19 encodings loop code	xxiv
Figure 20 ultrasonic sensor code	xxiv
Figure 21 process_frame() function code	xxv
Figure 22 servo motor code	xxvi
Figure 23 Example database of members in Firestore	xxvi
Figure 24 Example Database of images in Firebase Storage	xxvii
Figure 25 Example database of administrators	xxvii
Figure 26 Simplified diagram of facial recognition system	xxviii
Figure 27 Streaming Handler Class code	xxix
Figure 28 Streaming Object Class code	xxx
Figure 29 Streaming Server Class code	xxx
Figure 30 Face Detection and Streaming with Raspberry Pi Camera	xxxi
Figure 31 onCreateView() of Camera View page	xxxi
Figure 32 Android code for startRepeatingTask()	xxxii



# Chapter 1: Introduction

## 1.1 Background

Over the years, technology has seamlessly integrated into our daily lives, bringing forth remarkable breakthroughs in various areas such as cloud storage, drones, and transportation tracking systems. Amidst these thriving technologies, people are increasingly seeking solutions that offer convenience, and home automation has emerged as one such widely popular technology.

The inception of home automation can be traced back to 1998, gaining rapid recognition by the early 2000s [1]. Initially, home automation existed without integration with the Internet; however, the concept of a smart home intrigued individuals due to its novelty and perceived convenience. Automated devices, though unfamiliar at the time, captivated tech-savvy users.

The introduction of the Internet of Things (IoT) further expanded the horizons of technology. From tracking lost keys to implementing smart lighting systems on streets, the possibilities within the IoT realm are limitless, continuously shaping a highly technologically advanced society that seeks to enhance people's lives.

The convergence of IoT with home automation empowers individuals to control their devices via internet connections, particularly appealing to the tech-savvy younger generation for whom convenience is a paramount concern. Presently, home automation entails the remote control of internet-connected devices from virtually anywhere through a mobile device with internet connectivity. This application proves relevant for monitoring and security purposes, providing individuals with a means to oversee the safety of their homes.

## 1.2 Objective and scope

This project aims to design and implement a more cost-effective digital door lock. This project will take form in a digital facial recognition door lock, and it is designed to have a low energy consumption and an integrated supply of energy. The IoT-enabled home system consists of raspberry pi, a camera, and a servomotor. An advantage of an integrated supply of energy removes the inconvenience of replacing batteries.

Furthermore, the system will be designed to seamlessly integrate with smartphones through a distinctive mobile application, providing users with a user-friendly interface for control and monitoring. The mobile application will offer an intuitive user interface, facilitating the management of real time facial recognition feature. Users will have easy and efficient control over the various components through the application.



# Chapter 2: Literature review

## 2.1 Existing products and comparisons

The exploration of digital door locks has become increasingly vital in the contemporary landscape, where security, convenience, and smart home integration are paramount considerations.

As the demand for advanced security solutions continues to rise, the literature review focuses on 4 popular digital door lock brands and models, by price: Yew, Samsung, Bosch, Phillips [2]. Each brand offers distinct features catering to the diverse needs of consumers in the digital security market. The prices indicated are the minimum online rates available at the time this was written.

### 1. Yew F11 Facial Recognition Digital Lock:



*Figure 1 Yew F11 Digital Lock*

The Yew F1 is a digital lock that has facial recognition technology, complemented by PIN codes, anti-peep passwords, and an intrusion alarm. Operating on battery power, the Yew F11 aims to offer a balance of security and convenience through facial recognition. The Yew F11 uses 4200mAh Lithium Li-ion Rechargeable Battery and lasts up to 2 months on a single charge. This lock is priced at \$449 [9].

### 2. Samsung Digital Door Lock SHP-DP609:



*Figure 2 Samsung Digital Door Lock*

The SHP-DP609 by Samsung is highlighted as a top pick due to its exceptional combination of security, convenience, and smart home integration. The SHP-DP609 is commended for seamlessly merging cutting-edge technology with robust security features, offering advanced encryption, Smart Keyless Entry, and an Intrusion Alarm. It also has Wi-Fi IoT capabilities that enable remote control and monitoring, making it a versatile and trustworthy option for users seeking both security and convenience. The SHP-DP609 uses 8 AA batteries and can last up to 10 months. It can be shorter or longer depending on its usage [3]. It is priced at \$888 [4].

### **3. Bosch FU780 Digital Lock:**



*Figure 3 Bosch FU780 Digital Lock*

The Bosch FU780 has features such as PIN code access, RFID card access, and an intrusion alarm. Its battery-operated system ensures a reliable power source. The Bosch FU780 uses 4 AA batteries and can last up to 12 months [7]. It is priced at \$1199 [8]

#### 4. Philips Easykey 702:



*Figure 4 Philips Easykey 702 Digital Lock*

The Philips EasyKey 702 facial recognition door lock utilizes advanced 3D facial scanning technology, allowing users to unlock doors without physical contact. This lock employs dual cameras to create a three-dimensional scan of the user's face, like human vision. One standout feature is its built-in light, which activates in low-light conditions, ensuring accurate facial scans day or night. This precise facial recognition system minimizes the risk of unauthorized access, providing users with peace of mind regarding their home security. It is powered by 8 AA batteries [5]. With the Wi-Fi is continuously connected, the lithium battery life can last for 2 months. It is priced at \$1399 [6].

There is a general trend observed whereby the more expensive the price of the door lock, the longer it lasts. The exception being *Philips EasyKey 702*, this could be due to the dual cameras and built in light which might consume more power compared to the other locks.

If the digital lock is priced appropriately, it could have a relatively long lifespan. Nonetheless, there remains a concern about the batteries eventually depleting. Not much has been explored in providing an integrated power supply to the lock itself to boost its life span instead of relying on batteries.

In summary, this literature review highlights the unique features and characteristics of digital door locks from Samsung, Phillips, Bosch, and Yew. Each brand brings its distinct approach to security, authentication, and user convenience. However, none of these brands and model addresses the

inconvenience of changing batteries once they are depleted. Furthermore, the pricing of these locks is not particularly budget friendly.

# Chapter 3: Project Overview

## 3.1 Brief Project Description

This project will serve as a proof of concept that charging the digital lock while in use is feasible and can be explored more. Low-cost components will be used for this project. Below are the descriptions of these components and their justification for the project.

## 3.2 Main Hardware Components

### 3.2.1 Raspberry pi 4



*Figure 5 Raspberry Pi 4 (Rpi)*

The Raspberry Pi (Rpi) is a credit card-sized single-board computer developed by the Raspberry Pi Foundation, known for its compact size, affordability, and versatility. It has Wi-fi capabilities and GPIO pins that allow for motor control [10]. Hence it is suitable for this project.

### 3.2.2 Raspberry pi camera V2



*Figure 6 Raspberry pi camera V2*

The Raspberry pi camera V2 has a Sony IMX219 8-megapixel sensor allowing it to take high-definition videos and photos [11]. The high-definition camera will not compromise the facial recognition function of the digital lock.

### 3.2.3 Servo Motor



*Figure 7 Servo Motor*

Servo Motors are electronic devices and rotary or linear actuators that rotate and push parts of a machine with precision [12]. Though the raspberry pi does not have a dedicated clock system to generate PWM signals, the servo motors can still be operated using software generated PWM signals [13]. Only 1 servo motor is needed to serve as a mechanism to unlock the door lock.

### 3.2.4 Ultrasonic sensor (HC-SR04)

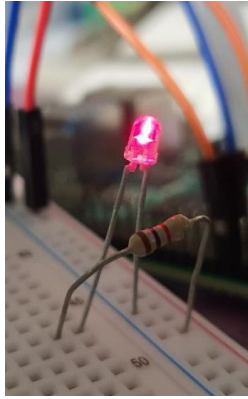


*Figure 8 Ultrasonic sensor*

The HC-SR04 ultrasonic sensor offers a low-cost solution for distance measurement. Employing ultrasonic waves at 40 kHz, it calculates distance by measuring the round-trip time of the sound reflected from an object [14]. It has a range of 2 cm to 400 cm and a respectable accuracy of 3 mm. The sensor will serve as a proximity detector for when a person walks towards the camera.



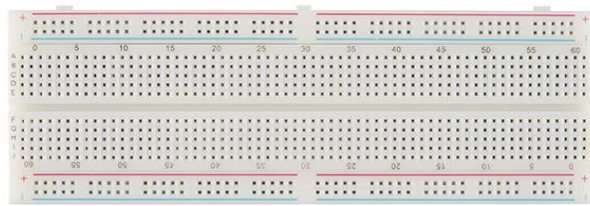
### 3.2.5 Led bulb



*Figure 9 Led*

The Led bulb will be lit to signal that the door lock is unlocked.

### 3.2.6 Breadboard



*Figure 10 Breadboard*

Bread board will be used to wire up the circuits for the different components.

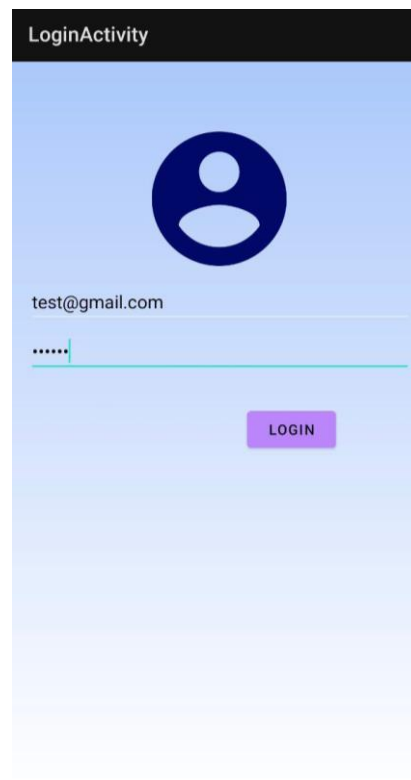
The most expensive component is the Rpi which costs USD35 [10]. The total cost of all the components amount to close to \$75. Which is a fraction of the cost of the facial recognition locks mentioned in the literature review.

## Chapter 4: Implementation

### 4.1 Application

An Android Studio mobile application has been developed to integrate with Firebase and Fire Storage, enabling real-time access to data. The mobile application enables users to monitor individuals at their doorstep and decide whether to grant them entry. Additionally, the application includes a feature to remotely control the house lights. The application is connected to the same network as the Rpi and they communicate via SSH (Secure Shell).

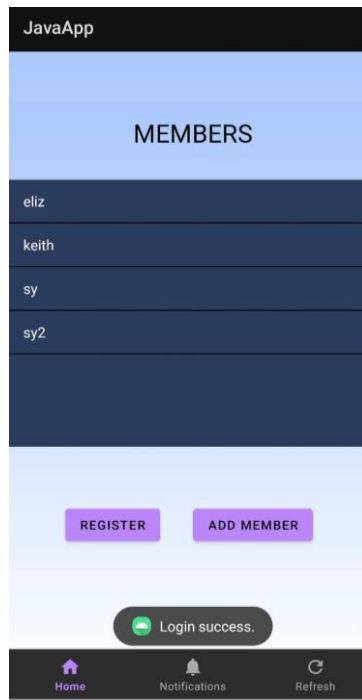
#### 4.1.1 Login Page



*Figure 11 Login page*

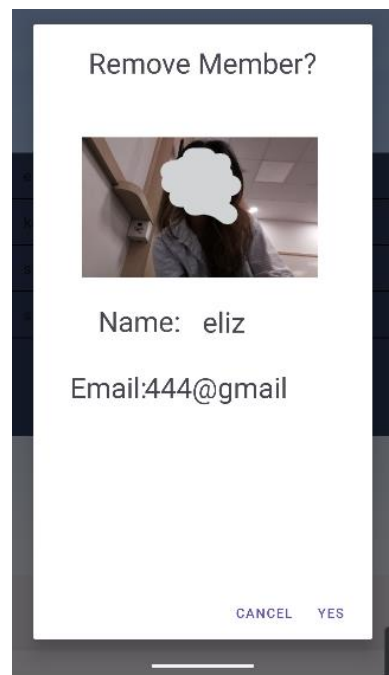
The login page is tailored to facilitate the sign-in process for administrators. Administrators are granted the ability to manage the list of members who are allowed to enter the house, they are individuals with their facial data stored in the database. This functionality allows administrators to efficiently add or remove members as needed, ensuring accurate and up-to-date access control for the household.

#### **4.1.2 Home Page**



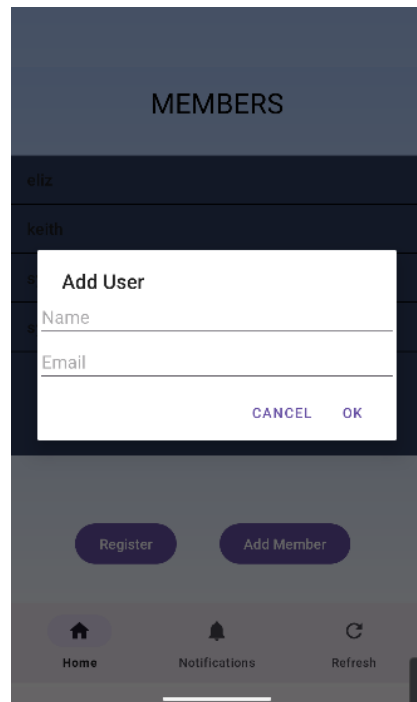
*Figure 12 Home page*

The home page has the list of all the house members that exists in the firebase database. The list is scrollable should there be a lot of members. Administrators can add members by pressing the "Add members" button.



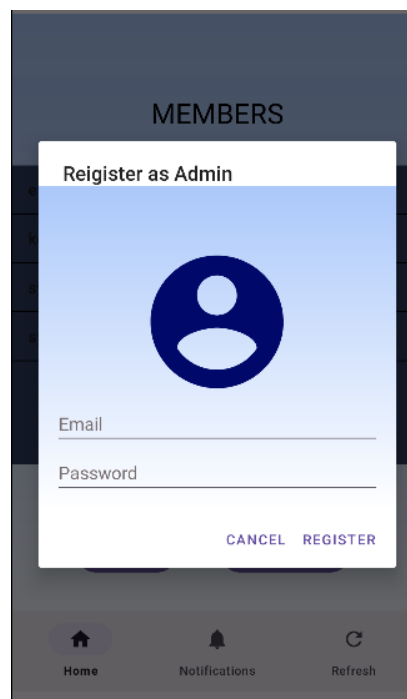
*Figure 13 Selected Member dialog*

If a member in the list is selected, a dialog will appear. The administrator will then have the option to remove the selected member from the list. Both the member's details and their face will be removed should the administrator clicks on "YES".



*Figure 14 Add member/user dialog*

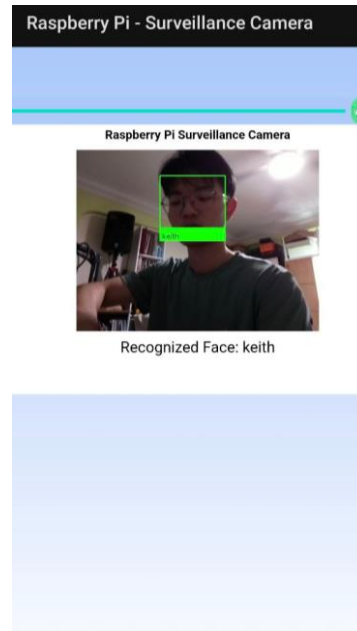
A dialog will pop up and administrators can input the member's name and email. Once the member's particulars are submitted, a prompt will appear to ask for the newly added member to face door lock camera. An image of the member's face will be taken and saved into the firebase storage database.



*Figure 15 Add administrator*

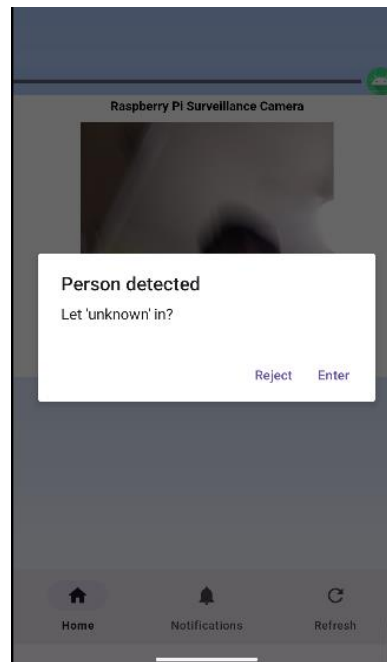
Administrators can also add other administrators as well by pressing the "Register" button. A dialog will pop up to ask for the new administrator's email and password.

### 4.1.3 Camera View Page



*Figure 16 Camera View page*

The camera view page contains the stream for the camera view in real time. If the face is recognised by the camera, the member's name will appear on the page and the app will send a command to the Rpi, unlocking the door.



*Figure 17 alertTextView dialog*

If the face is not in the fire database storage, the face will be labelled as unknown. A dialog, “alertView”, will pop up and prompt the administrator if the unknown person is allowed to enter. If administrator presses “Enter” the door will be unlocked.

## 4.2 Sensor and Facial Recognition System

The facial recognition system uses the Open-Source Computer Vision Library (OpenCV) python library. It allows for basic image processing operations such as reading and writing images and accessing pixel values and image properties. The “face\_recognition” python library is used as well. It is powerful and user-friendly tool that allows for accurate face detection and face recognition.

### 4.2.1 Training the model

```
boxes = face_recognition.face_locations(rgb, model="cnn")
# compute the facial embedding for the face
encodings = face_recognition.face_encodings(rgb, boxes)
```

*Figure 18 face\_recognition functions code*

To train the model, a loop is used to iterate every image that is stored in Firebase Storage. In each iteration, each image will be processed. The functions, “face\_locations” and “face\_encodings”, used are from the “face\_recognition” library. “face\_locations” detects the faces in the frame using the Convolutional Neural Networks (CNN) model and “face\_encodings” encodes the detected faces.

The “face\_recognition” library supports the models, Convolutional Neural Networks (CNN) and Histogram of Oriented Gradients (HOG). While the CNN model generally provides a higher accuracy compared to HOG model, it is generally more computationally expensive. In the case of a facial recognition door lock where security is a priority, the CNN model is chosen.

```
# loop over the encodings
for encoding in encodings:
    # add each encoding + name to our set of known names and
    # encodings
    knownEncodings.append(encoding)
    knownNames.append(fullName)
```

*Figure 19 encodings loop code*

After the image is processed, the encodings will be looped, and the name of each image will be stored along with the corresponding encodings.

#### **4.2.2 Ultrasonic sensor code**

```
ultrasonic = DistanceSensor(echo=17, trigger=4, threshold_distance=0.5)

def call_recognition():
    # call the recognition script
    subprocess.run(["python", "web.py"])
ultrasonic.when_in_range = call_recognition
```

*Figure 20 ultrasonic sensor code*

The ultrasonic sensor will continuously detect the proximity of any person in front of the door. If the person is within the range of 0.5m away from the door, the facial recognition code will be called.

#### **4.2.3 Facial Recognition file**



```

def process_frame(self, image):
    rgb_image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    face_names = []
    small_rgb_image = cv2.resize(rgb_image, (0, 0), fx=0.25, fy=0.25)
    try:
        face_locations = face_recognition.face_locations(small_rgb_image)
        face_encodings = face_recognition.face_encodings(small_rgb_image, face_locations)
    except Exception as e:
        print("error is e")
    threshold = 0.5
    name = "Unknown"
    if not face_locations:
        face_names.append(name)
        self.face_name = name
    for face_encoding in face_encodings:
        face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
        matches_within_threshold = np.where(face_distances <= threshold)[0]
        if len(matches_within_threshold) > 0:
            best_match_index = matches_within_threshold[np.argmin(face_distances[matches_within_threshold])]
            id,name = known_face_names[best_match_index].split('_',1)
        else:
            name = "Unknown"
        face_names.append(name)
        self.face_name = name
    for (top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
        top *= 4
        right *= 4
        bottom *= 4
        left *= 4
        # Box around the face
        cv2.rectangle(image, (left, top), (right, bottom), (0, 255, 0), 2)
        # label below the face
        cv2.rectangle(image, (left, bottom - 35), (right, bottom), (0, 255, 0), cv2.FILLED)
        font = cv2.FONT_HERSHEY_SIMPLEX
        cv2.putText(image, name, (left + 6, bottom - 6), font, 0.6, (0, 0, 10), 1)
    if name != "Unknown":
        self.face_recognized = True
    return image

```

*Figure 21 process\_frame() function code*

Firstly, the “face\_recognition” library is used to detect and encode the face from each frame in real time by invoking the functions “face\_locations()” and “face\_encodings()”

Next, the Euclidean distance between the newly detected face and each known face from the database is then calculated using “face\_distance()” [16]. If the distance falls below a specified threshold, the detected face is considered a match to a known face. The code selects the best match within the threshold and extracts the corresponding name known face encodings array. If no matches are found within the threshold, the detected face is labelled as "Unknown." This process allows for the identification and verification of real-time faces, which makes it suitable for a facial recognition door lock. The threshold is set to 0.5 for this project. Adjusting the threshold parameter can impact the facial recognition’s accuracy, if the threshold is too high, the best matched face will be identified, an unknown face might be labelled as known. If it is too low, a known face might be labelled as unknown.

Lastly, a rectangular frame is added to the detected face along with its corresponding name for a more obvious and user-friendly result (fig,16).

#### 4.2.4 Servo motor file

```
servo = Servo(18)
val = -1
try:
    while True:
        servo.value = 0.3
        sleep(3)
        break
```

Figure 22 servo motor code

The servo motor file contains a simple command that activates the servo motor. The servo motor is connected to pin 18 of the Rpi.

## 4.3 Google's Realtime Firebase Database

### 4.3.1 Firebase Firestore

Firebase Firestore, by Google Firebase, is a database service for storing and synchronizing data. It offers real-time data synchronization and scalability [17]. The live updates allow the mobile application to always keep track of the members.

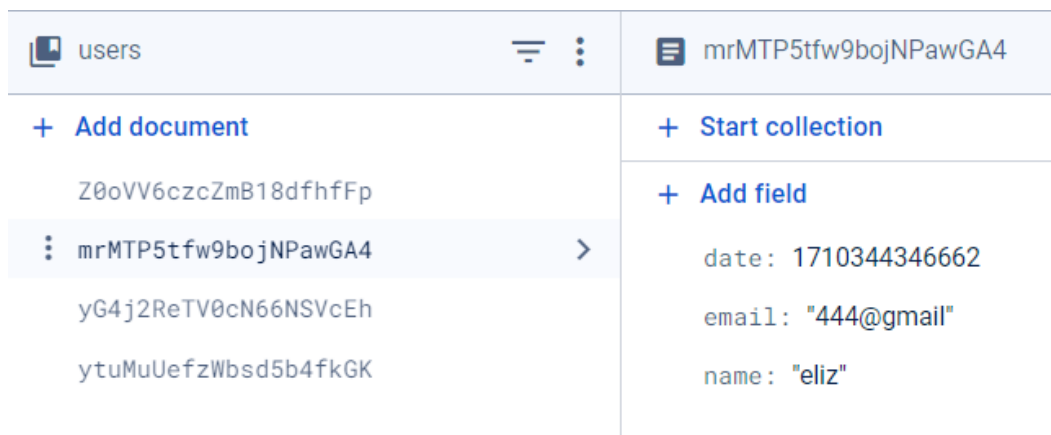






Figure 23 Example database of members in Firestore

When a member is added through the app, the member's name and email along with the date added will be stored in Firebase Fire store. Each member has a unique ID which is automatically generated.



### 4.3.2 Firebase Storage

Name
 Z0oVV6czcZmB18dfhfFp_sy2.jpg
 mrMTP5tfw9bojNPawGA4_eliz.jpg
 yG4j2ReTV0cN66NSVcEh_keith.jpg
 ytuMuUefzWbsd5b4fkGK_sy.jpg

*Figure 24 Example Database of images in Firebase Storage*

Firebase Storage, by Google Firebase, is designed for storing and serving user-generated content like images, videos, audio, and other files [17]. It supports operations like uploading and downloading content, suitable for processing. When a member is added, a picture of the member's face will be taken, after which it will be uploaded to Firebase Storage. The name of the image is the unique ID of the member and his or her corresponding name, this is done to prevent confusion within the system when there are members with identical names.

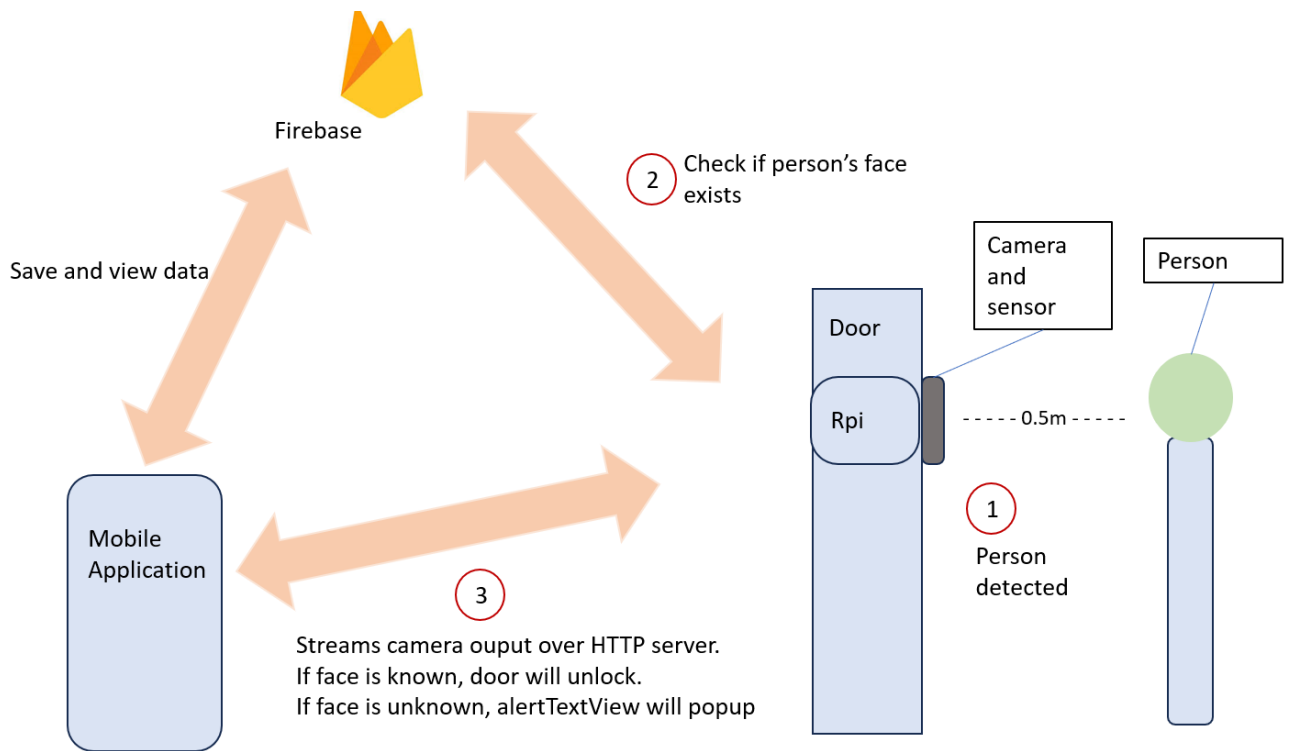
### 4.3.3 Firebase Authentication

Identifier	Providers	Created ↓	Signed In	User UID
test2@gmail.com		Mar 23, 2024	Mar 23, 2024	8IOJ33jtd6eTrDmvsTTjbqUPW...
test@gmail.com		Mar 23, 2024	Mar 23, 2024	9coTwGBWhNVXTydbhYPJX5...

*Figure 25 Example database of administrators*

Firebase Authentication offers a variety of authentication methods, including email and password and popular third-party providers like Google and Facebook, making it versatile for different application needs [18]. The mobile app for this facial recognition project uses email and password verification for secure login and easy registering.

## 4.4 Web communication



*Figure 26 Simplified diagram of facial recognition system*

A simple flow of the communication system is as explained:

1. A person is detected by the ultrasonic sensor at 0.5 metres away
2. The Rpi will retrieve the images from Firebase.
3. The Rpi will then compare the encodings of the faces from the Firebase with the face detected.
4. If the face detected is labelled "unknown", the "alertTextView" dialog will show (fig.17)
5. If the administrator presses "Enter" (fig. 17), the app will send a command to the Rpi to unlock the door.
6. If face is known, the door will unlock.

#### **4.4.1 Streaming Handler Class**

```

class StreamingHandler(server.BaseHTTPRequestHandler):
    def do_GET(self):
        global recognised
        endThread = True
        recognised = "Unknown"
        if self.path == '/':
            self.send_response(301)
            self.send_header('Location', '/index.html')
            self.end_headers()
        elif self.path == '/index.html':
            content = PAGE.encode('utf-8')
            self.send_response(200)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(content))
            self.end_headers()
            self.wfile.write(content)
        if self.path == '/get-recognized-face':
            self.send_response(200)
            self.send_header('Content-type', 'application/json')
            self.end_headers()
            self.wfile.write(json.dumps({"name": output.face_name}).encode())
        elif self.path == '/stream.mjpg':
            self.send_response(200)
            self.send_header('Content-Type', 'multipart/x-mixed-replace; boundary=FRAME')
            self.end_headers()
            try:
                while True:
                    with output.condition:
                        output.condition.wait()
                        frame = output.frame
                    self.wfile.write(b'--FRAME\r\n')
                    self.send_header('Content-Type', 'image/jpeg')
                    self.send_header('Content-Length', len(frame))
                    self.end_headers()
                    self.wfile.write(frame)
                    self.wfile.write(b'\r\n')
                    if output.face_detected and endThread:
                        Timer(1, shutdown_server).start()
                        endThread = False
            except Exception as e:
                print(f"Stream stopped: {e}")
        else:
            self.send_error(404)
            self.end_headers()

```

*Figure 27 Streaming Handler Class code*

The StreamingHandler class facilitates the communication between the web server and the application. It manages all HTTP GET requests to various endpoints, which includes redirection to the main HTML page (/index.html), retrieval of recognized face names (/get-recognized-face), and streaming of MJPEG video (/stream.mjpg). The streaming functionality enables live video display with facial recognition processing, allowing for real-time

recognition updates. The function `shutdown_server` is used to stop the server upon detecting a face to release the camera resource.

#### 4.4.2 Streaming Object Class

```
class StreamingOutput(object):
    def __init__(self):
        self.frame = None
        self.buffer = io.BytesIO()
        self.condition = Condition()
        self.face_recognized = False
        self.face_name = "Unknown"

    def write(self, buf):
        if buf.startswith(b'\xff\xd8'):
            self.buffer.truncate()
            with self.condition:
                self.frame = self.buffer.getvalue()
                if len(self.frame) > 0:
                    self.condition.notify_all()
                    data = np.frombuffer(self.frame, dtype=np.uint8)
                    if data.size > 0:
                        image = cv2.imdecode(data, cv2.IMREAD_COLOR)
                        if image is not None:
                            image = self.process_frame(image)
                            _, buffer = cv2.imencode('.jpg', image)
                            self.frame = buffer.tobytes()
            self.buffer.seek(0)
        return self.buffer.write(buf)
```

*Figure 28 Streaming Object Class code*

The `StreamingOutput` class is in charge of processing incoming video frames, ensuring efficient handling and recognition of faces with the function `process_frames()`. Upon receiving a new frame, the class decodes it using OpenCV, checks for the presence of a face, and updates the `face_detected` flag.

#### 4.4.3 StreamingServer Class

```
class StreamingServer(socketserver.ThreadingMixIn, server.HTTPServer):
    allow_reuse_address = True
    daemon_threads = True
```

*Figure 29 Streaming Server Class code*

```

with picamera.PiCamera(resolution='640x480', framerate=24) as camera:
    output = facepart.StreamingOutput()
    camera.start_recording(output, format='mjpeg')
    time.sleep(1)
    try:
        address = ('', 8000)
        server = StreamingServer(address, StreamingHandler)
        Timer(10, shutdown_server_enable_ultrasonic).start()
        server.serve_forever()

```

*Figure 30 Face Detection and Streaming with Raspberry Pi Camera*

When the StreamingServer class is instantiated above, it will create a HTTP server that is capable of handling streaming data.

The camera output will be streamed over the custom HTTP server created by the StreamingHandler. The mobile application will retrieve the stream via the HTTP URL and display it in the “Camera View Page” (fig.16).

It also has a shut down condition where if no face is detected for a set amount of time. In this case 10 seconds. The function “shutdown\_server\_enable\_ultrasonic()” is called to shut down the server and the ultrasonic sensor will be enabled again. This is to allow effective energy utilisation.

#### 4.4.4 Mobile Application Web View

```

public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
    View view = inflater.inflate(R.layout.fragment_notifications, container, attachToRoot: false);
    TextView alertTextView = view.findViewById(R.id.alertTextView);
    try{
        initializeViews(view);
        setupWebView();
    }catch(Exception e){
        Log.e(tag: "webb", msg: "webb: ", e);
    }
}

```

*Figure 31 onCreateView() of Camera View page*

In the android application, the web view is initialised in the “onCreateView()” method.

```

@Override
public void onCreateView(@NonNull View view, @Nullable Bundle savedInstanceState) {
    super.onCreateView(view, savedInstanceState);
    startRepeatingTask(view.findViewById(R.id.alertTextView));
}

1 usage
private void startRepeatingTask(Textview alertTextView) {
    final Runnable updateFaceNameRunnable = new Runnable() {
        @Override
        public void run() {
            getFaceName(new FaceNameCallback() {
                1 usage
                @Override
                public void onFaceNameReceived(String faceName) {
                    if (isAdded() && !faceName.equals(prevName)) {
                        if (!faceName.equals("Unknown") && !faceName.equals("null")) {
                            showAlert(alertTextView, faceName);
                        }
                        prevName = faceName;
                    }
                }
            });
            handler.postDelayed(this, delayMillis: 2000); // Schedule the next execution
        }
    };
    handler.post(updateFaceNameRunnable); // Kick off the first execution
}

```

*Figure 32 Android code for startRepeatingTask()*

To get the name of the face detected from the web in real time, the “startRepeatingTask()” method is used to initiate a recurring task that updates a dialog popup called “alertTextView” (fig.17) with the name of the detected face. A handler is used to schedule the runnable to run every 2 seconds such that the “alertTextView” stays updated with the latest detected face name.

To prevent the dialog from continuously popping up whenever the same unknown face remains within the frame, a simple check is used to verify if the name of the currently detected face matches the name of the previously identified face.

## Chapter 5: Conclusion

This project concluded that it is possible to create a facial recognition door lock by utilizing Raspberry Pi and a mobile application. The project aimed to create a facial recognition system that is affordable and improves on the initial problem of battery depletion. It is successful in demonstrating that a facial recognition door lock can have an integrated power supply whilst still being budget friendly.



# Chapter 6: Further improvements

While the project is successful in developing the facial recognition system, further improvements can still be made to enhance this home system.

## 6.1 Low light conditions

In low light conditions, the Raspberry Pi camera (Figure 6) might not capture a clear facial image. This can hinder the image clarity can impede image processing, potentially leading to the facial recognition system's inability to accurately identify the face. Hence, a lighting device can be added to the Rpi to illuminate the face

## 6.2 Scalability

The scalability of this home automation system can be improved by allowing users to add more home devices by themselves. The system could also support a wider range of device types such as fans or air conditions.

## 6.3 Rechargeable battery

With the system in need of a constant power supply, it is not very efficient. Furthermore, some houses may not have a power supply available near their door. A way to resolve this is to have a rechargeable battery back. The Raspberry Pi can operate using a battery pack, and to ensure continuous operation as the battery nears depletion, an additional servo motor could be incorporated. This servo motor would activate the power, allowing the Raspberry Pi to remain operational while the battery pack is recharging.

# References

[1] D. Hendricks (22 April, 2014), "The history of Smart Homes - IOT Evolution World," The history of Smart Homes, <https://www.iotevolutionworld.com/m2m/articles/376816-history-smart-homes.htm> (accessed Jan. 15, 2024).

[2] J. Mike (11 Dec, 2023), "12 Best Digital Locks in Singapore 2024: Top smart locks," Singapore, <https://productnation.co/sg/5971/best-digital-lock-singapore/> (accessed Jan. 15, 2024).

[3] samsungddladmin, "Samsung Digital Door Lock FAQ," *SamsungDigitalLife / Samsung Digital Door Locks / Audio Video Intercom / Samsung Smart Things*. <https://www.samsungdigitallife.com/samsung-digital-door-lock-faq/> (accessed Jan.16, 2024).

[4] "MY DIGITAL LOCK Selling Samsung DR708 Digital Lock Unlock HDB Door and Condo Laminate Main Door using WI-FI only in Singapore \$1150 Call YISHUN, BUKIT BATOK,

MACPHERSON AND TAMPINES, 98440884,” *My Digital Lock*.  
<https://www.mydigitallock.com.sg/shp-dp-609/> (accessed Mar. 23, 2024).

[5] Phillips, “Buy the philips 7000 series Smart video door lock ddl702iagcw/97 smart video door lock,” Philips, [https://www.philips.com.my/c-p/DDL702IAGCW\\_97/7000-series-smart-video-door-lock#:~:text=DDL702%2D1HWS'%20s%20C%2D,mins%2C%20enhanced%20security%20is%20as](https://www.philips.com.my/c-p/DDL702IAGCW_97/7000-series-smart-video-door-lock#:~:text=DDL702%2D1HWS'%20s%20C%2D,mins%2C%20enhanced%20security%20is%20as) sured. (accessed Mar. 23, 2024).

[6] “Philips Digital Lock with Facial Recognition | Face Locks In Singapore,” *HDB Laminate Main Door, Metal Gate and Bedroom Door Supplier in Singapore*.  
[https://www.laminatedoor.com.sg/product/face-recognition-digital-lock/?gad\\_source=1&gclid=Cj0KCQjw-\\_mvBhDwARIsAA-Q0Q6K8JjJWHydN3CGTLTzCXJy4F5sDuSXcwt\\_ywLHA16VBYN2ravegMaAib9EALw\\_wcB](https://www.laminatedoor.com.sg/product/face-recognition-digital-lock/?gad_source=1&gclid=Cj0KCQjw-_mvBhDwARIsAA-Q0Q6K8JjJWHydN3CGTLTzCXJy4F5sDuSXcwt_ywLHA16VBYN2ravegMaAib9EALw_wcB) (accessed Mar. 23, 2024).

[7] “Bosch FU 780,” *Smart Digital Lock At Reddot*. <https://smartdigitallockreddot.sg/product/bosch-fu-780/> (accessed Jan. 16, 2024).

[8] “Bosch,” *Smart Digital Lock At Reddot*. <https://smartdigitallockreddot.sg/digital-door-lock/bosch/> (accessed Mar. 23, 2024).

[9] “Yew F11 Face Recognition Door Digital Lock - Yew Digital.”  
<https://yewdigitallock.com.sg/product/yew-f11-face-recognition-door-digital-lock/> (accessed Jan. 16, 2024).

[10] Raspberry Pi Foundation, “Raspberry Pi — Teach, Learn, and Make with Raspberry Pi,” *Raspberry Pi*, 2019. <https://www.raspberrypi.org/>

[11] R. P. Ltd, “Buy a Raspberry Pi Camera Module 2,” *Raspberry Pi*.  
<https://www.raspberrypi.com/products/camera-module-v2/>

[12] “The Purpose of Servo Motors,” *Fuji Electric Global*.  
[https://www.fujielectric.com/about/column/detail/servo\\_01.html](https://www.fujielectric.com/about/column/detail/servo_01.html)

[13] *Raspberrypi.org*, 2024. <https://projects.raspberrypi.org/en/projects/grandpa-scarer/3> (accessed Jan. 29, 2024).

[14] LastMinuteEngineers, “How HC-SR04 Ultrasonic Sensor Works & How to Interface It With Arduino,” *Last Minute Engineers*, Mar. 29, 2019. <https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/>

[15] BILAL\_AI, “HOG vs. CNN ‘Unveiling the Power of Image Classification,’” *Medium*, Oct. 06, 2023. <https://medium.com/@entrepreneurbilal10/hog-vs-cnn-unveiling-the-power-of-image-classification-a3d1585d76e8#:~:text=While%20CNNs%20outperform%20HOG%20in> (accessed Mar. 23, 2024).

[16] “face\_recognition package — Face Recognition 1.4.0 documentation,” *face-recognition.readthedocs.io*. [https://face-recognition.readthedocs.io/en/latest/face\\_recognition.html#:~:text=Given%20a%20list%20of%20face](https://face-recognition.readthedocs.io/en/latest/face_recognition.html#:~:text=Given%20a%20list%20of%20face) (accessed Mar. 24, 2024).

[17] “Cloud Storage for Firebase,” *Firebase*.

<https://firebase.google.com/docs/storage#:~:text=Cloud%20Firestore%20is%20a%20flexible>

[18] Google, “Firebase Authentication | Firebase,” *Firebase*, 2019.

<https://firebase.google.com/docs/auth>

