VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

– – – – – – – – – – –

CAPSTONE PROJECT

ENHANCE THE SECURITY OF SMART HOME APPLICATIONS BASED ON ARTIFICIAL INTELLIGENCE AND THE INTERNET OF THINGS

MAJOR: COMPUTER ENGINEERING

THESIS COMMITTEE: **Computer Engineering Project Council 2**
SUPERVISOR: **Ph.D. LE TRONG NHAN**
REVIEWER: **Assoc. Prof. TRAN NGOC THINH**

STUDENT 1: **HUYNH HUU HANH** **1910161**
STUDENT 2: **TRUONG HOANG NAM** **1910358**

Ho Chi Minh City, May 2023

ĐẠI HỌC QUỐC GIA TP.HCM

TRƯỜNG ĐẠI HỌC BÁCH KHOA

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

KHOA:KH & KT Máy tính____
BỘ MÔN:KHMT_____

NHIỆM VỤ LUẬN ÁN TỐT NGHIỆP

Chú ý: Sinh viên phải dán tờ này vào trang nhất của bản thuyết trình

HỌ VÀ TÊN: HUỲNH HỮU HẠNH _____ MSSV: 1910161 _____
HỌ VÀ TÊN: TRƯƠNG HOÀNG NAM_____ MSSV: 1910358 _____
NGÀNH: KĨ THUẬT MÁY TÍNH_____ LỚP:_____

1. Đầu đề luận án:

Nâng cao tính bảo mật cho nhà thông minh dựa trên trí tuệ nhân tạo và kết nối vạn vật

(Enhance the security of smart home applications based on Artificial Intelligence and the Internet of Things)

2. Nhiệm vụ (yêu cầu về nội dung và số liệu ban đầu):

Tìm hiểu và hiện thực kĩ thuật nhận dạng giọng và khuôn mặt của người nói, hỗ trợ cho việc xác thực cho việc điều khiển các thiết bị trong nhà thông minh.

Hiện thực hệ thống quản lý các thiết bị và phân quyền điều khiển các thiết bị tới người sử dụng. Giao diện tương tác thân thiện với người sử dụng.

Tìm hiểu và tích hợp hệ thống trợ lý ảo trong nhà thông minh. Hệ thống có thể trả lời các câu hỏi đơn giản từ phía người sử dụng.

3. Ngày giao nhiệm vụ luận án: 01/02/2023

4. Ngày hoàn thành nhiệm vụ: 30/05/2023

5. Họ tên giảng viên hướng dẫn: **Phần hướng dẫn:**

1) TS. LÊ TRỌNG NHÂN — Toàn bộ

Nội dung và yêu cầu LVTN đã được thông qua Bộ môn.

Ngày ..... tháng ..... năm .....

CHỦ NHIỆM BỘ MÔN
(Ký và ghi rõ họ tên)

GIẢNG VIÊN HƯỚNG DẪN CHÍNH
(Ký và ghi rõ họ tên)

TS. Lê Trọng Nhân

PHẦN DÀNH CHO KHOA, BỘ MÔN:

Người duyệt (chấm sơ bộ):_____
Đơn vị: _____
Ngày bảo vệ: _____
Điểm tổng kết: _____
Nơi lưu trữ luận án: _____

ĐẠI HỌC QUỐC GIA TP.HCM

TRƯỜNG ĐẠI HỌC BÁCH KHOA

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

Ngày 13 tháng 06 năm 2023

PHIẾU ĐÁNH GIÁ LUẬN VĂN/ ĐỒ ÁN TỐT NGHIỆP

(Dành cho người hướng dẫn)

1. Họ và tên SV: Huỳnh Hữu Hạnh
MSSV: 1910161 Ngành (chuyên ngành):Kỹ thuật Máy tính
Trương Hoàng Nam
MSSV: 1910358 Ngành (chuyên ngành):Kỹ thuật Máy tính

2. Đề tài: **Nâng cao tính bảo mật cho nhà thông minh dựa trên trí tuệ nhân tạo và kết nối vạn vật**

3. Họ tên người hướng dẫn: TS. Lê Trọng Nhân

4. Tổng quát về bản thuyết minh:
Số trang: 99 Số chương: 5
Số bảng số liệu: 30 Số hình vẽ: 54
Số tài liệu tham khảo: 30 Phần mềm tính toán:
Hiện vật (sản phẩm)

5. Những ưu điểm chính của LV/ ĐATN:
Sinh viên cơ bản đã hiện thực được các tính năng của một ngôi nhà thông minh dựa trên nền tảng kết nối vạn vật, để điều khiển thiết bị và giám sát một số cảm biến cơ bản. Trợ lý ảo RASA đã được tích hợp trong đề tài để tăng tính tương tác giữa người dùng và hệ thống.

Quy trình cài đặt phân quyền cho người dùng khi điều khiển thiết bị bằng giọng nói được thiết kế khoa học, bao gồm hệ cơ sử dữ liệu và ứng dụng trên thiết bị di động. Việc học một giọng nói được hiện thực hợp lý trong đề tài. Đây là tính năng chính để nâng cao tính bảo mật trong việc điều khiển thiết bị bằng giọng nói.

Luận văn được trình bày bằng tiếng Anh.

6. Những thiếu sót chính của LV/ĐATN:
Xác suất nhận dạng giọng chủ chưa cao.

7. Đề nghị: Được bảo vệ ☑ Bổ sung thêm để bảo vệ ○ Không được bảo vệ ○

8. Các câu hỏi SV phải trả lời trước Hội đồng:
a. Sinh viên hãy trình bày các rủi ro khi giọng người chủ bị thay đổi do các yếu tố ngoại cảnh (buồn ngủ, đau họng, v.v…) ?

9. Đánh giá chung (bằng chữ: Xuất sắc, Giỏi, Khá, TB): Xuất sắc Điểm : 9.7/10

Ký tên (ghi rõ họ tên)

Lê Trọng Nhân

TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KH & KT MÁY TÍNH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

Ngày tháng năm

PHIẾU ĐÁNH GIÁ LUẬN VĂN/ ĐỒ ÁN TỐT NGHIỆP

(Dành cho người ~~hướng dẫn~~/phản biện)

1. Họ và tên SV: Huỳnh Hữu Hạnh
MSSV: 1910161 Ngành (chuyên ngành):Kỹ thuật Máy tính
Trương Hoàng Nam
MSSV: 1910358 Ngành (chuyên ngành):Kỹ thuật Máy tính

2. Đề tài: **Nâng cao tính bảo mật cho nhà thông minh dựa trên trí tuệ nhân tạo và kết nối vạn vật (Enhance the security of smart home applications based on Artificial Intelligence and the Internet of Things)**

3. Họ tên người ~~hướng dẫn~~/phản biện: PGS.TS. Trần Ngọc Thịnh
4. Tổng quát về bản thuyết minh:
Số trang: 99 Số chương: 5
Số bảng số liệu: 30 Số hình vẽ: 54
Số tài liệu tham khảo: 30 Phần mềm tính toán:
Hiện vật (sản phẩm)
5. Những ưu điểm chính của LV/ ĐATN:
Luận án này đề xuất giải pháp tích hợp nhận dạng khuôn mặt & giọng nói của người dùng, giúp xác thực, phân quyền điều khiển các thiết bị trong nhà thông minh.
Với nhận dạng khuôn mặt, các tác giả sử dụng OpenCV và DLib. Với tính năng nhận dạng giọng nói, các tác giả sử dụng kỹ thuật speech-to-text của Google kết hợp RASA Open Source, một khung xử lý ngôn ngữ tự nhiên (NLP), để trích xuất ý định từ giọng nói của người dùng.
Đối với Nhận dạng người nói, các tác giả sử dụng MFCC kết hợp huấn luyện trên mô hình GMM.

Hệ thống được triển khai trên nền tảng Next Unit of Computing (NUC) của Intel, với nhiều thành phần phần mềm phổ biến khác. Các thử nghiệm bao gồm 10 mẫu đầu vào để nhận dạng ý định và nhận dạng giọng nói, 60 cách phát biểu để nhận dạng người nói. Kết quả là khá tốt. Phần mềm giao diện trên thiết bị di động thân thiện và dễ dùng.
6. Những thiếu sót chính của LV/ĐATN:
Tập dữ liệu để huấn luyện và thử nghiệm các lệnh điều khiển trong nhà còn khá ít và đơn giản, khó đánh giá được tính đúng đắn, ổn định của hệ thống.

Định dạng của luận án cần phải được sửa đổi. Các kết quả nên được đưa vào Chương 4 hoặc một chương riêng biệt thay vì trong Chương 5 với phần kết luận.

7. Đề nghị: Được bảo vệ ☑ Bổ sung thêm để bảo vệ □ Không được bảo vệ □
8. Các câu hỏi SV phải trả lời trước Hội đồng:
a. Thời gian đáp ứng của các lệnh khá chậm lên đến vài giây, tác giả có thể giải thích và cách khắc phục nếu có thể.

9. Đánh giá chung (bằng chữ: Xuất sắc, Giỏi, Khá, TB): *Giỏi* Điểm : 9/10

Ký tên (ghi rõ họ tên)

Trần Ngọc Thịnh

COMMITMENT

We pledge that this project is based on our supervisors' ideas and knowledge. All studies and data have not been published. The references, numbers, and statistics are reliable and honest. The group completed the project requirements set by the faculty of computer science and engineering - department of Computer Engineering.

Sincerely,

Huynh Huu Hanh
Truong Hoang Nam

ACKNOWLEDGEMENT

We have been through almost four years of university programs. This project is the final milestone to re-evaluate the entire learning process. We could not have undertaken and gone through without the encouragement, confidence, and advice brought to us by friends and family, by my supervisors and fellow students, and by people, we met throughout that journey.

First and foremost, we would express our most profound appreciation to our project supervisors, Ph.D. Le Trong Nhan. He has been there, providing his heartfelt support and guidance at all times. He has given us invaluable guidance, inspiration, and suggestions in our quests for knowledge during our university time. Without his assistance and dedicated involvement in every step throughout the process, this thesis would have never been accomplished.

We sincerely thank the teachers who occupy the Faculty of Computer Science and Engineering in particular and the Ho Chi Minh City University of Technology in general, who have constantly been imparting knowledge in the past four years. Their support, encouragement, and credible ideas have been great contributors to the completion of the project.

Last but not least, it would be inappropriate if we omit to thank our friends and family. Our late parents' unconditional love and blessings, and the care of friends and acquaintances who never let things get dull have all made a tremendous contribution to helping us reach this stage in our life. We thank them for putting up with us in difficult moments when we felt stumped and for goading us on to reach for our passions.

Finally, we would like to wish you good health and success in your noble life.

ABSTRACT

Smart home technology generally refers to any suite of devices, appliances, or systems that connect to a common network that can be independently and remotely controlled. The smart homes of tomorrow promise to increase comfort, aid elderly and disabled people, and help inhabitants save energy. Unfortunately, smart homes today are far from this vision – people who already live in such a home struggle with complicated user interfaces, inflexible home configurations, and difficult installation procedures. With the above concerns, we recommend **Enhance the security of smart home applications based on Artificial Intelligence and the Internet of Things**.

The goal of this project is to implement a system to control the devices automatically. The system is secure by the face recognition method. As for controlling home appliances, we implement a virtual assistant. This virtual assistant helps users with voice control to control home appliances, such as turning on/off the fan or turning on/off the lights. In addition, it can also help you to complete some tasks in daily life, e.g., scheduling, and asking for time. This application of artificial intelligence will help improve the user experience, making it easier to manipulate devices such as the elderly and children.

We collect data from sensors such as temperature, and humidity from which to provide solutions to users or alarms in dangerous cases. We also implement a mobile application, so that users can interact with house appliances and observe how often the device is used so that they can adjust their lifestyle accordingly.

Keywords: Face Recognition, Speaker Verification, Virtual Assistant, User-friendly Interface, Security.

Table of contents

Tables

List of figures

CHAPTER 1

INTRODUCTION

This chapter provides an overview of the project. The first section briefly overviews the combination of the Internet of Things (IoT) and Artificial Intelligence (AI) in daily life. After that, we introduce smart homes, their advantages, and their disadvantages. Then, we outline the scope and aim of the project. Finally, we describe the structure of the project.

1.1 Introduction

The Internet of Things and Artificial Intelligence are two of the most important technological advancements in the current 4.0 technology revolution. The combination of IoT and AI promises to provide humans with numerous new meanings in the future. The AI system supports the rapid and precise provisioning of connected devices via IoT to give real-world problem solutions.

The Internet of Things is a system that connects physical devices. This contributes to the development of a flexible network of activities, as well as the development of a process in production and business operations. Applications include IoT in agriculture, industry, manufacturing, aquaculture, and so on.

Artificial Intelligence is intelligence-perceiving, synthesizing, and inferring information demonstrated by machines, as opposed to intelligence displayed by animals and humans. Example tasks in which this is done include speech recognition, computer vision, translation between natural languages, as well as other mappings of inputs.

Artificial Intelligence of Things (AIoT) is a system of networked devices capable of collecting and learning human response patterns. AIoT technologies allow previously non-smart items to become smart by connecting them to the internet through various embedded devices, communication protocols, sensor networks, internet protocols, and applications. It is clear that the two technologies above are extremely superior. Many applications have been developed and will be developed in the future to assist individuals in developing and enhancing business productivity.

Now, let's look at some of the applied research IoT and AI in everyday life that we've been striving towards:

Healthcare Industry: There are many AIoT-based healthcare services utilized in the medical industry, which can be classified as electronic health and telecare networks, diagnosis, prevention, rehabilitation, and monitoring devices. Essentially, it may increase access to clinical services thus alleviating pressure on healthcare facilities, and it can empower individuals to have more autonomy of their own well-being at all times. Besides, When applying IoT technologies and artificial intelligence, this application will help the healthcare industry to take care of human health with many important steps. Thanks to that, diseases are detected quickly and health is continuously monitored.

Self-Driving Car: On the self-driving car system, there is a sensor camera to help recognize and collect images such as objects, traffic signs on the road or heat sensors, radar so on. This is linked to the IoT system to create a collection of variables. Along with AI technology is the central agency that controls the vehicle on the road. Or take actions such as stopping in front of signs and maintaining a safe distance from cars. In addition, it can stop in time to avoid collisions.

Smart Home: The trend of leading in the 4.0 technology revolution is smart home technology [1]. AIoT enables devices to be tightly linked to collect and exchange data. This data will be used by AI technology to make decisions. With smart home technology, the AIoT system will connect devices, robots, and machines to create a unified network.

AIoT technology is widespread in a changing technology world. Recognizing the tremendous potential of AIoT technology, our team developed a solution to meet housing demands. Provide an excellent customer experience with the smart home system.

1.1.1 Smart Home [2]

Smart home is a system where appliances and devices can be automatically controlled remotely from anywhere with an internet connection using a mobile or other networked device. Devices in a smart home are interconnected through the internet, allowing the user to control functions such as security access to the home, temperature, lighting, and a home theater remotely.

Smart homes supply homeowners with convenience, saving energy, security, and ease of use. Here are the detailed benefits of smart homes:

Convenience: Homeowners can control appliances, thermostats, lighting, and other features all using a smartphone or tablet, rather than controlling them using different devices. Users can get notifications and updates on issues in their homes. For instance, smart doorbells allow homeowners to see and communicate with people who come to their doors even when they are not at home. Users can set and control the internal temperature, lighting, and appliances as well.

Saving Energy: Smart home can help you use energy efficiently during peak demand times. You can adjust the temperature inside your house and save a lot of energy by turning the system off whenever you are away. And saving energy can also save you money.

Security: The ability to detect fires and intrusion is a major benefit of having a smart home. Since, smart home has two-factor authentication, as well as gas sensors to prevent dangerous situations [3].

Besides the benefits, it still has challenges that need to be overcome:

Cost: Compare to traditional homes, smart homes require a higher investment. This problem is one of the common disadvantages of smart homes. Several companies provide smart home systems, but they are quite expensive.

Security: May pose some security risks as products are connected to networks and can be hacked. No internet system is 100% secure; there is always a loophole. Since smart homes are either integrated with a Wifi network, it's not hard for hackers to creep into your home network.

Internet Dependency: The basic requirement of every smart technology is the internet, without which it isn't easy to control your smart home. If there is no internet connection for any reason, things may even take a bad turn leading to complications even in simple tasks.

1.1.2 Advanced Technology for Smart Home

Artificial Intelligence and Internet of Things enhance the user experience and add value to smart homes by making life more secure, comfortable, convenient, and safe.

Artificial Intelligence: AI encompasses the ability to connect multiple IoT devices, coupled with superior processing and learning abilities, and use them to pre-empt human behavior. AI-powered smart home devices can interact with each other and acquire new data that assists in learning human habits. Data collected is used to predict the behavior of users and develop situational awareness, i.e., understand user preferences and change parameters accordingly. Prominent AI applications in smart homes include Face Recognition and Voice Assistant.

Face recognition adds more actionable information to system alerts. Just showing your face to the camera is enough to unlock the door or adjust the room's temperature.

Besides, with virtual assistants, such as Alexa, Siri, and Google Assistant. Most people are familiar with at least some aspects of AI functionality. Artificial intelligence continues to improve, understanding commands better and responding in more relevant ways. Integration with voice controls will enable the user to save time and ease off certain tedious tasks. Controlling devices and home appliances through voice should be made a priority as providing user-friendly services always yields productive results for the business.

Internet of Things: IoT enables connected devices, vehicles, buildings, and

other objects – which are embedded with software, sensors, and the internet – to communicate with each other and can either be controlled remotely or can transfer the information to a remote user with AI. These connected devices can monitor the status of any device connected under the same network and provide real-time data with the help of AI.

1.1.3 Proposed System

To build this system, we need to combine many different technologies, ranging from hardware to software. However, the three most important technologies needed to build this system, include Face Recognition, Voice Assistant, Internet of Things, and Mobile Application.

Face Recognition: We use face recognition for the security system. You have set up your smart home security system and granted access to the people you want to allow into your home, no need for you to answer the doorbell, or to respond to a request to enter. That means you need to create profiles for friends, relatives, and others you want the system to identify. A face recognition system works by taking an image of a face and predicting whether the face matches another face stored in a database. You can also set up a blacklisted person trying to unlock your smart door, the system will send you an emergency alert. This feature is more suitable in the context of Covid 19, as biometric security such as a fingerprint is restricted.

Voice Assistant: We build a voice assistant. This assistant includes speaker verification and intent classification that help you manipulate the devices in your home without touching them, setting a timer, or scheduling some daily tasks in your life. Besides, it also helps to verify who is speaking, each person in the system will have the right to manipulate the device differently. If the operation is not authorized, it will not happen.

Internet of Things: IoT is the core that plays an essential role in implementing the system. We use several devices such as sensors, lights, doors, and a server to collect and analyze data. This system then uses this data to improve infrastructure, user experience, services, and more. It also allows the system to be scaled more efficiently, adding and setting up devices quickly and straightforwardly.

Mobile Application: We build a mobile app for users to easily monitor sensor values, control devices, register new users, as well as help homeowners manage the permissions and members in the house. This means that a user has different device control permissions and this permission will be decided by the host.

1.2 Scope and Objectives

The aim of the project is to implement and enhance the security of a smart device controller system. The system can recognize the user's face and voice, then control the devices by voice. It consists of the Authentication Phase, Speaker Verification Phase, and Voice Command Phase. We detail each phase with illustrations below in order to better understand our process.

Authentication Phase: At this phase, the user authenticates the identity to be able to enter the house. The system uses face recognition to recognize whether the face is already in the database. If not, the user is not able to enter the house. In case, if the system recognizer in adverse circumstances such as lack of light, or the user wears a mask, the system will ask the user to verify with a pin code to be able to enter the system.

Speaker Verification Phase: Once a user interacts with our assistant, the system authenticates who you are. That is, each user has permission to manipulate different devices. If the user interacts with devices that do not allow the operation, the operation will not happen.

Voice Command Phase: If the system has verified whose voice it is, the system will convert the voice to text, then the assistant will process what you want to do, and then give commands to the devices and finally, the assistant will respond to you.

Besides, we use Adafruit IO to observe the change in temperature, humidity, and the state of home devices.

1.3 Project Structure

The project consists of five chapters: Introduction, Background, System Design, Implementation, and Conclusion.

Chapter 1. Introduction
Provide details about the introduction, proposed system, as well as the scope and objectives of the project.

Chapter 2. Background
Introduce the techniques and technologies to be used for the system., such as Face recognition, Voice recognition, Speaker verification, and Rasa open source.

Chapter 3. System Design
Provide overview architecture, conceptual database design, and mobile use-case design.

Chapter 4. Implementation
Describe how to implement the modules in the system.

Chapter 5. Conclusion
Summarize the achievements of this project and the plan for future development.

CHAPTER 2

BACKGROUND

After we have defined the scope and objectives of the system. In this chapter, we present the relevant knowledge used to implement the system. Firstly, we introduce face recognition, its principles, as well as the frameworks that are widely used. In addition, we also introduce some voice recognition techniques and Rasa assistant; point out advantages and disadvantages, and choose the most suitable technique for the project.

2.1 Face Recognition

This section focuses on the principles behind methods currently used for face recognition, which have a wide variety of uses from bio-metrics, surveillance, and forensics.

After a brief description of how faces can be detected in images and a basic introduction to OpenCV for face detection. After that, we continue to research face recognition principles and the face recognition library which was built using *dlib's state-of-the-art* face recognition built with deep learning. The model has an accuracy of **99.38%** on the Labeled Faces in the Wild benchmark [5].

2.1.1 Face Detection

Face detection is an artificial intelligence (AI) based computer technology used to find and identify human faces in digital images. Face detection technology can be applied to various fields including security, bio-metrics, law enforcement, entertainment, and personal safety. It provides surveillance

and tracking of people in real time.

2.1.1.1 Principles of Face Detection

Face detection is a simple feature used to detect the presence of human faces without comparing the differences between faces. In general, the system must undertake the following analysis steps:

The first step to face recognition is, unsurprisingly, to recognize the face. How do you identify a face? Well, a face generally has two eyebrows, two eyes, a nose, a mouth, and a jawline. So essentially like this:

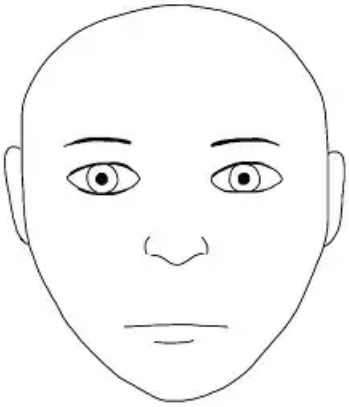

Figure 2.1: Face of Human

The problem is that most images have too much stuff going on. Consider Benedict Cumberbatch's face. By just eyeballing it, we can make a generic face fit over Cumberbatch.

Figure 2.2: Face of Benedict Cumberbatch

The problem is we got too much stuff going on in the picture. Sure we can eyeball it, but it would be difficult to specify a computer to match the face.

We really care just about the edges, and edges are defined by changes in gradient. We can use the sobel operator to calculate the vertical and horizontal derivatives. Basically, it's looking for any drastic change in the surrounding pixels, which is basically an edge. If one pixel is white, and the neighbor is black, then the derivative (amount of change) between the two cells would be high.

But using polar coordinates on each pixel provides too granular of a view to be used for comparison. We need to pool the polar coordinates, kind of like performing a survey of the surrounding coordinates to get a general sense of the directions each group of cells is pointing. That's called the Histogram of Oriented Gradients (HOG).

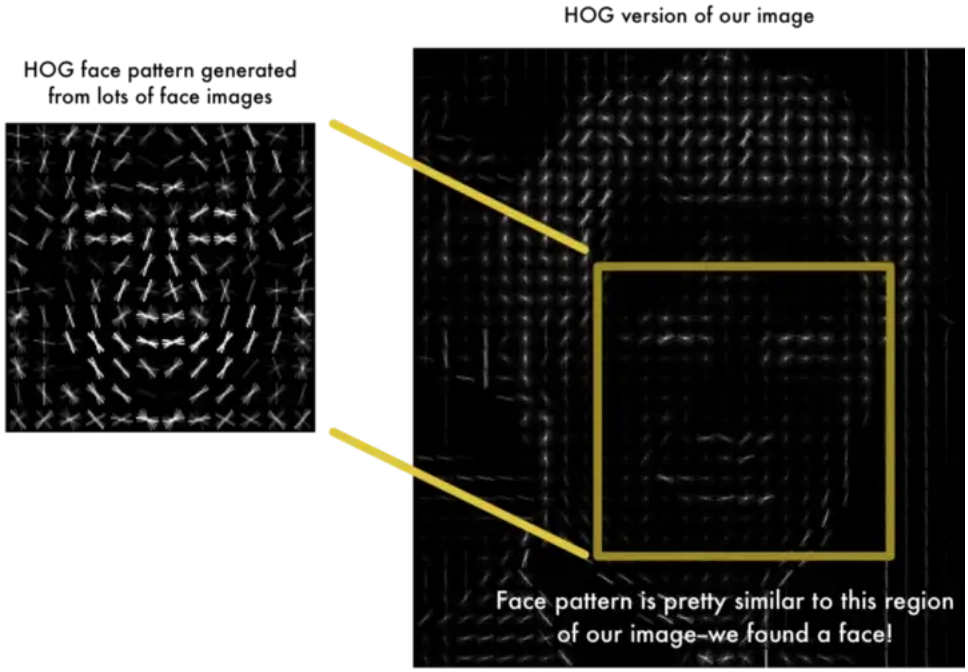

Figure 2.3: Face of Benedict Cumberbatch after using HOG

Well, maybe our generic face is less than ideal, but luckily there are better distillations that we can use. We can just use a sliding window to see if we get a reasonable match.

After detecting the face, we can get a lot of useful information, including the location of the landmarks (e.g. nose, eyes, jawline). And it also allows us to distill the essence of a face into a vector. Comparing vectors of different faces is essentially how facial recognition works.

2.1.1.2 Face Detection using OpenCV

You can perform fast, accurate face detection with OpenCV using a pre-trained deep-learning face detector model shipped with the library.

OpenCV provides the trainer as well as the detector. We can train the classifier for any object like cars, planes, and buildings by using OpenCV. There are two primary states of the cascade image classifier first one is training and the other is detection.

OpenCV provides two applications to train cascade classifiers *haartraining* and *traincascade*. These two applications store the classifier in different file formats.

For training, we need a set of samples. There are two types of samples:

- **Negative sample**: It is related to non-object images.
- **Positive sample**: It is a related image with detected objects.

A set of negative samples must be prepared manually, whereas the collection of positive samples is created using the *createsamples* utility.

2.1.2 Face Recognition

Face recognition is a type of biometric technology that uses data to verify the presence of a human being's face in a digital capture. There are two main uses for face recognition software: recognition and authentication.

2.1.2.1 Principles of Face Recognition

Face recognition is a long-standing issue that has been thoroughly researched for over 30 years. In general, Face recognition will process as following steps.

Step 1: Finding all the faces

Face detection went mainstream in the early 2000s when Paul Viola and Michael Jones invented a way to detect faces that were fast enough to run on cheap cameras [4].

However, much more reliable solutions exist now. We're going to use a method invented in 2005 called Histogram of Oriented Gradients — or just HOG for short.

To find faces in an image, we'll start by making our image black and white because we don't need color data to find faces. Using this technique, we can now easily find faces in any image.

Step 2: Posing and projecting faces

Whew, we isolated the faces in our image. But now we have to deal with the problem that faces turned different directions look totally different to a computer.

To account for this, we will try to warp each picture so that the eyes and lips are always in the sample place in the image. This will make it a lot easier for us to compare faces in the next steps.

To do this, we are going to use an algorithm called face landmark estimation. There are lots of ways to do this, but we are going to use the approach invented in 2014 by Vahid Kazemi and Josephine Sullivan [4].

The basic idea is we will come up with 68 specific points (called landmarks) that exist on every face — the top of the chin, the outside edge of each eye, the inner edge of each eyebrow, etc. Then we will train a machine learning algorithm to be able to find these 68 specific points on any face.

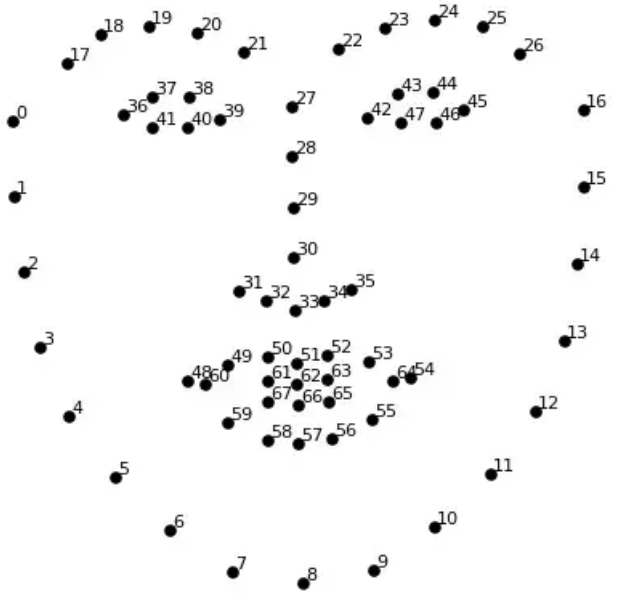

Figure 2.4: The 68 landmarks we locate on every face

Step 3: Encoding faces

The simplest approach to face recognition is to directly compare the unknown face we found in Step 2 with all the pictures we have of people that have already been tagged. When we find a previously tagged face that looks very similar to our unknown face, it must be the same person.

There's actually a huge problem with that approach. A site like Facebook with billions of users and a trillion photos can't possibly loop through every previous-tagged face to compare it to every newly uploaded picture. That would take way too long. They need to be able to recognize faces in milliseconds, not hours.

The solution is to train a Deep Convolutional Neural Network. But instead of training the network to recognize pictures of objects like we did last time, we are going to train it to generate 128 measurements for each face.

The training process works by looking at 3 face images at a time:

1. Load a training face image of a known person
2. Load another picture of the same known person
3. Load a picture of a totally different person

Encoding our face image

This process of training a convolutional neural network to output face embeddings requires a lot of data and computer power. Even with an expensive NVidia Telsa video card, it takes about 24 hours of continuous training to get good accuracy.

But once the network has been trained, it can generate measurements for any face, even ones it has never seen before. So this step only needs to be done once. Lucky for us, the fine folks at OpenFace already did this and they published several trained networks which we can directly use.

So all we need to do ourselves is run our face images through their pre-trained network to get the 128 measurements for each face. Here are the measurements for our test image.

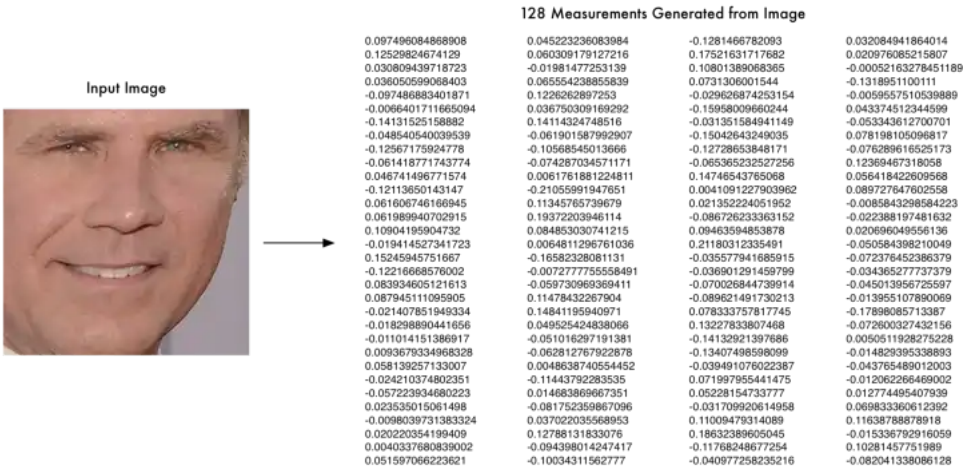

Figure 2.5: 128 measurements generated from image

Step 4: Finding the person's name from the encoding

This last step is actually the easiest step in the whole process. All we have to do is find the person in our database of known people who has the closest measurements to our test image.

You can do that by using any basic machine-learning classification algorithm. No fancy deep-learning tricks are needed. We'll use a simple linear SVM classifier, but lots of classification algorithms could work.

All we need to do is train a classifier that can take in the measurements from a new test image and tells which known person is the closest match. Running this classifier takes milliseconds. The result of the classifier is the name of the person.

2.1.2.2 Using face recognition open source

Python offers some great free libraries for face recognition. They offer great wrappers for C++ libraries like dlib. Using *face_ recognition*, we can identify the face locations of an image.

We essentially transformed someone's face into a numerical representation of points (landmarks). The logical next step to face recognition would be to compare these landmarks and calculate some kind of landmark distance. But these are just landmarks that make sense to humans. They may not be that distinguishing, or there may exist some other landmarks that are more distinguishing. So while this technique is useful in capturing different parts

of a person's face, it may not be best suited to tell us definitively whether two photos contain the same person.

Modern machine learning techniques do a similar process of converting a face into a numerical representation. Saying you have a large value in point 61 of your 128-point facial vector representation may not mean anything to us. After we have a numerical representation of the face, we can now use Euclidean distance to measure how different the vector is from some previous vector.

There is no magical distance that we could use to tell us that two people are the same. We can only measure distances and test their accuracy. Packaged facial recognition services often provide distance as confidence. The dlib model has a purported 99.38% accuracy on the standard LFW face recognition benchmark using a distance of 0.6 [5].

2.2 Speech Recognition

In our system, speech recognition plays a role to control devices in the house without physically touching them. Besides, the owner can talk with the Rasa bot about aspects of life such as: asking about the time, asking about the weather, making an appointment, etc. The voice bot will be presented more clearly in the following section.

Speech recognition, also known as automatic speech recognition (ASR), is a capability that enables a program to process human speech into a written format.

The following section will introduce some popular models and toolkits in this field, compare them, and apply the most suitable one for our Speech Recognition module.

2.2.1 Open Source Speech-to-Text Libraries

2.2.1.1 DeepSpeech

DeepSpeech is an open-source Speech-To-Text engine, using a model trained by machine learning techniques based on Baidu's Deep Speech research paper. It also implies machine learning technology using Google's

TensorFlow framework to fulfill its mission.

The original DeepSpeech paper from Baidu popularized the concept of "end-to-end" speech recognition models. The goal of "end-to-end" models was to simplify the speech recognition pipeline into a single model. In addition, the theory introduced by the Baidu research paper was that training large deep learning models, on large amounts of data, would yield better performance than classical speech recognition models.

Today, the Mozilla DeepSpeech library offers pre-trained speech recognition models that you can build with, as well as tools to train your own DeepSpeech models.

2.2.1.2 Kaldi

Kaldi is an open-source toolkit for speech recognition written in C++ and licensed under the Apache License v2.0. The toolkit is already pretty old (around 11 years old) but is still constantly updated and further developed by a pretty large community.

Kaldi is written mainly in C/C++, but the toolkit is wrapped with Bash and Python scripts. Kaldi is best tested on Debian and Red Hat Linux but will run on any Linux distribution, or on Cygwin or Mac OsX. The tools compile on commonly used Unix-like systems and on Microsoft Windows.

2.2.1.3 Google Speech-to-Text API

Google Speech-to-Text accurately converts speech into text with an API powered by the best of Google's AI research and technology.

It is an automation tool for transcribing speech into text with its implemented model on Google Server with many notable benefits:

- Support global user base with Speech-to-Text's extensive language in over 63 languages and variants.
- Leverage Google's most advanced deep learning neural network algorithms for automatic speech recognition (ASR).
- Experiment with, create and manage custom resources with the Speech-to-Text UI.

- Deploy ASR wherever you need it, whether in the cloud with the API or on-premises with Speech-to-Text On-Prem.

However, since Google only gives users 60 minutes of free transcription and supports transcribing files already in a Google Cloud Bucket, the free credits won't get you very far.

2.3 Speaker Verification

The objective of speaker verification is the authentication of a claimed identity from measurements on the voice signal. Applications of speaker verification include entry control to restricted premises, access to privileged information, funds transfer, credit card authorization, voice banking, and similar transactions.

This method is divided into text-dependent and text-independent methods. In text-dependent methods, the speaker verification system has prior knowledge about the text to be spoken and the user is expected to speak this text. However, in a text-independent system, the system has no prior knowledge about the text to be spoken and the user is not expected to be cooperative. Text-dependent systems achieve high speaker verification performance from relatively short utterances, while text-independent systems require long utterances to train reliable models and achieve good performance.

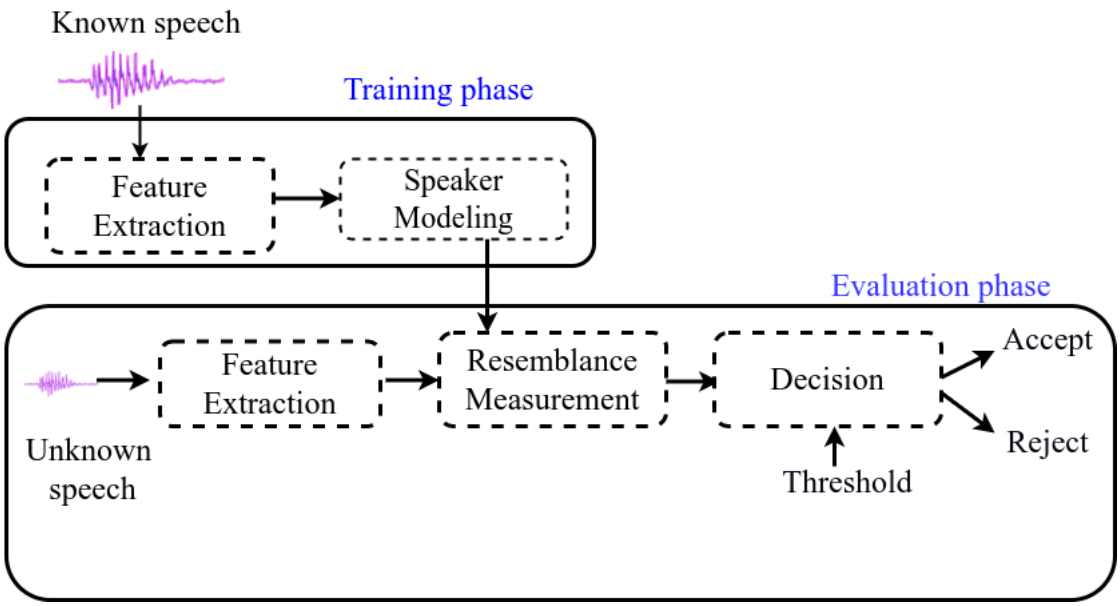

Figure 2.6: Speaker Verification System

As it is shown in the above block diagram of a basic speaker verification system, a speaker verification system involves two main phases: the training phase in which the target speakers are enrolled, and the testing phase in which a decision about the identity of the speaker is taken. From a training point of view, speaker models can be classified into generative and discriminative. Generative models such as Gaussian Mixture Model (GMM) estimate the feature distribution within each speaker. Discriminative models such as Support Vector Machines and Deep Neural Networks (DNN), in contrast, model the boundary between speakers.

2.3.1 Gaussian Mixture Models

The classical machine learning models to train an audio dataset somehow revolves around Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM). A GMM (Gaussian mixture model) can be thought of as a single-state HMM (Hidden Markov Model). In other words, a state in an HMM can be thought to have a mixture of distributions, with the probability of belonging to a distribution being represented by the emission probability (aka observation probability); each state in the HMM can have a unique set of emission probabilities. Therefore, each state in an HMM can be thought of as a GMM (with emission probabilities representing the probability of association to a distribution).

Other methods include computing the *out-of-sample*, *log-likelihood* upon the addition of each state, and the number of states maximizing the log-likelihood is chosen as the reasonable number of states to work with. Though, GMM is preferably used because it is more reliable than HMM, although HMM possesses more accuracy. Secondly, GMM yields output much faster than HMM. It consumes optimal resources compared to HMM. Most of the time either of them is used independently or with a DNN (deep neural network). However, sometimes they both can be used together in combination. Choosing any of the above-prescribed approaches is Arts-more-than-Science.

2.3.2 Feature Extraction Techniques

The next step is to extract features from these audio representations so that our algorithm can work on these features and perform the task it is designed for. Here's a visual representation of the categories of audio features that can be extracted.

After extracting these features, it is then sent to the machine learning model for further analysis. Audio feature extraction underpins a massive proportion of audio processing, music information retrieval, audio effect design, and audio synthesis. Design, analysis, synthesis, and evaluation often rely on audio features, but there is a large and diverse range of feature extraction tools presented to the community. An evaluation of existing audio feature extraction libraries was undertaken. Almost 7 + libraries were tried to perform the feature extraction task for retrieval of MFCC and delta MFCC.

2.3.3 Library of Feature Extraction

In this system, we have used the *python_ speech_ features* library. It fulfilled all of our requirements in feature engineering without causing any trouble. This library provides common speech features for ASR including MFCCs and filter-bank energies. You will need NumPy and Spicy to run these files. We have even achieved remarkable accuracy in identification tasks.

2.4 Rasa Open Source

In this system, we choose Rasa to develop the Voice assistant to make it easier for homeowners to interact with devices, improve their quality of life, as well as feel more comfortable in their living space.

Rasa is an open-source conversational AI platform that allows you to understand and hold conversations, and connect to messaging channels and third-party systems through a set of APIs. It supplies the building blocks for creating virtual (digital) assistants or chatbots.

Rasa provides the infrastructure & tools necessary for building the very best assistants - ones that meaningfully transform how customers communicate

with businesses.

2.4.1 NLU Training Data

NLU training data stores structured information about user messages. The goal of NLU (Natural Language Understanding) is to extract structured information from user messages.

NLU training data consists of example user utterances categorized by intent. To make it easier to use your intents, give them names that relate to what the user wants to accomplish with that intent, keep them in lowercase, and avoid spaces and special characters.

This usually includes the devices *intent* and any *entities* their message contains. You can add extra information such as *regular expressions* and *lookup tables* to your training data to help the model identify intents and entities correctly.

Intent

We group the examples according to the idea or the goal the message is expressing, which is also called the intent. In the code block on the right, we have added an intent called greet, which contains example messages like *"Hi"*, *"Hey"*, and *"Good morning"*. Intents and their examples are used as training data for the assistant's Natural Language Understanding (NLU) model.

```
nlu:
- intent: greet
  examples: |
    - Hi
    - Hey!
    - Hallo
    - Good day
    - Good morning

- intent: subscribe
examples: |
    - I want to get the newsletter
    - Can you send me the newsletter?
    - Can you sign me up for the newsletter?

- intent: inform
  examples: |
    - My email is example@example.com
    - random@example.com
    - Please send it to anything@example.com
    - Email is something@example.com
```

Figure 2.7: Intents in Rasa

Entities

Entities are structured pieces of information inside a user message. For entity extraction to work, you need to either specify training data to train an ML model or you need to define regular expressions to extract entities using the *RegexEntityExtractor* based on a character pattern.

When deciding which entities you need to extract, think about what information your assistant needs for its user goals.

Entities are annotated in training examples with the entity's name. In addition to the entity name, you can annotate an entity with synonyms, roles, or groups.

In the example above, our purpose is to extract the intent *inquire_time* and *control_device*. Here, we have entities about *place*, *device*, and *command*. The place entity with two values *"Việt Nam"*, and *"London"*. The device entity with two values *"đèn"*, and *"quạt"*. And the last one - command entity with three values *"tắt"*, *"bật"*, and *"mở"*.

Regular Expressions

You can use regular expressions to improve intent classification and entity extraction using the *RegexFeaturizer* components.

The format for defining a regular expression is as **Figure 2.9**:

```
nlu:
- regex: account_number
  examples: |
    - \d{10,12}
- intent: inform
  examples: |
    - my account number is [1234567891](account_number)
    - This is my account number [1234567891](account_number)
```

Figure 2.8: Regular Expressions in Rasa

Here *account_number* is the name of the regular expression. When used as a feature for the *RegexFeaturizer* the name of the regular expression does not matter. When using the *RegexEntityExtractor*, the name of the regular expression should match the name of the entity you want to extract.

Lookup Table

Lookup tables are lists of words used to generate case-insensitive regular expression patterns.

When you supply a lookup table in your training data, the contents of that table are combined into one large regular expression. This regex is used to check each training example to see if it contains matches for entries in the lookup table.

Lookup table regexes are processed identically to the regular expressions directly specified in the training data and can be used either with the *RegexFeaturizer* or with the *RegexEntityExtractor*. The name of the lookup table is subject to the same constraints as the name of a regex feature.

2.4.2 Pipeline Component

Tokenizers

Tokenization is breaking the raw text into small chunks. Tokenization breaks the raw text into words, and sentences called tokens. These tokens help in understanding the context or developing the model for the NLP.

The tokenization helps in interpreting the meaning of the text by analyzing the sequence of the words. Example: The text *"Chào buổi sáng"* can be tokenized by *WhitespaceTokenizer* into *"Chào"*, *"buổi"*, *"sáng"*.

Entity Extractors

Entity extractors extract entities, such as devices or places from the user message.

- **CRFEntityExtractor**: This component implements a conditional random field (CRF) to do named entity recognition. CRFs can be thought of as an undirected Markov chain where the time steps are words and the states are entity classes. Features of the words (capitalization, POS tagging, etc.) give probabilities to certain entity classes, as are transitions between neighboring entity tags: the most likely set of tags is then calculated and returned.

- **RegexEntityExtractor**: This component extract entities using the *lookup tables* and *regexes* defined in the training data. The component checks if the user message contains an entry of one of the lookup tables or matches one of the regexes. If a match is found, the value is extracted as an entity. This component only uses those regex features that have a name equal to one of the entities defined in the training data. Make sure to annotate at least one example per entity.

2.5 Conclusion

After researching the library and model of the technology that will be used in the system. Below will be a summary of each technology.

Face recognition: It is a quick and efficient verification system. It is faster and more convenient compared to other biometric technologies like fingerprints or retina scans. Facial recognition algorithms have near-perfect accuracy in ideal conditions. There is a higher success rate in controlled settings but generally a lower performance rate in the real world. It is difficult to accurately predict the success rate of this technology, as no single measure provides a complete picture.

Voice recognition: We use an alternative method for Speech-to-Text which is the Google Speech Recognition method. It is a free library for performing speech recognition, with support for several engines and APIs, online and offline. Although the performance is not compared with other libraries, it is enough for us to build a Voice Assistant for our system. In conclusion, we decide to use Google Speech Recognition to implement a speech recognition application. Based on our research, although Kaldi has the best performance among speech recognition libraries, the essential factor is that Kaldi still does not support embedded devices yet. Furthermore, Google Speech Recognition is suitable for low-resource platforms, flexible design, and a focus on practical application development.

Speaker verification: We have also studied, compared, and tried to incorporate different approaches and algorithms to find out the most efficient model for speaker verification. We believe the MFCC-GMM model is most appropriate based on parameters like identification accuracy,

computation time, false rejection, and false acceptance rate.

Rasa Open Source: Rasa has all these features: built-in enterprise-grade concurrency capabilities, rich functions covering all the needs of chatbots, rich documents and tutorials, and a huge global community. This is why we choose Rasa for our system.

CHAPTER 3

SYSTEM DESIGN

After researching about principles of AI techniques in the previous chapter. In this chapter, we design the architecture of our system which includes AI, mobile application, and IoT. We also start by outlining the functional and non-functional requirements of the system. Then, we introduce the hardware and software components used to implement the system. After that, we design the mobile application's flow. Finally, we explain how to build the database for our system.

3.1 Proposal Architecture

In this project, we build an AI system that users can be recognized and authenticated by Face Recognition. After that, users can interact with devices through Voice Assistant. This allows for a more natural way of interacting with the device, as well as providing an easier and quicker way to access information.

In addition, users can use a phone connected to the internet to view the information using our mobile application. This provides users with a comprehensive overview of their house, allowing them to monitor and control various aspects of their house remotely.

Figure 3.1 is the proposed architecture of our system, which includes a micro-controller, an AI system, and sensors. The micro-controller acts as the main controller, connecting all the other components together. The camera module is used for face recognition, while the microphone and speaker are used for voice recognition and speaker verification. Additionally, the Adafruit

server is used to store and retrieve data from the system. Finally, we have a mobile app so that users can interact and monitor the system's metrics and relative settings.

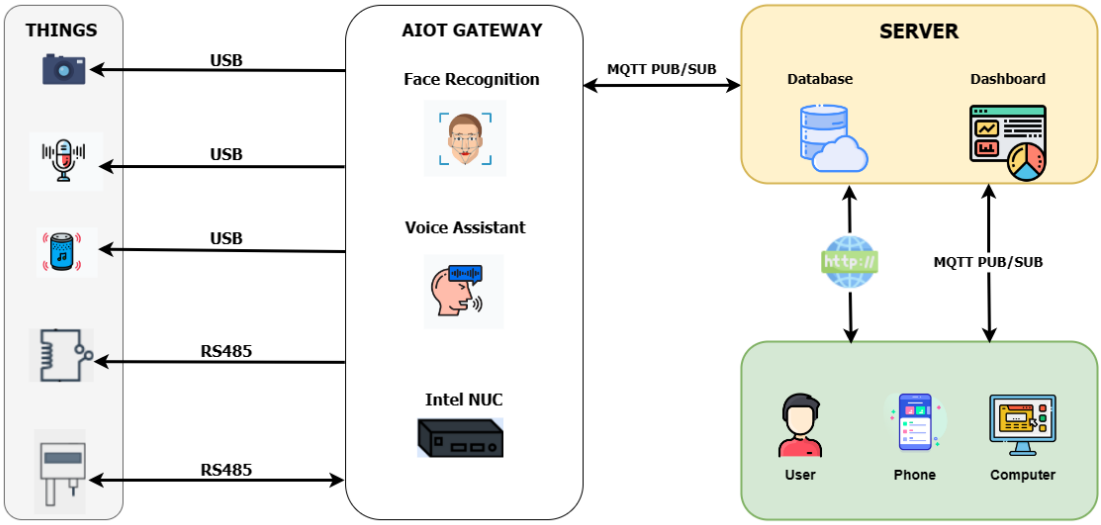

Figure 3.1: System architecture diagram

3.1.1 Functional Requirement

In this section, we present some basic functional requirements of the system:

1. **Identify users**: The system can recognize users by their faces and voices to increase security.
2. **Analyze user's requests**: The voice assistant can analyze the user's request based on their voice to provide a reasonable response.
3. **Train a new user**: A new user can enroll in the AI system by taking pictures of his/her face and recording his/her voice's records.
4. **Monitor sensor's values**: The system will display the sensor values in real time.

3.1.2 Non-functional Requirement

There are a few non-functional requirements of the system that need to be taken into account:

1. The system is capable of running 24/7 without any failures, making it one of the most reliable solutions available today.
2. The system can receive up to 50 simultaneous requests. This allows users to control multiple devices at once, making it easier and more efficient to manage their homes.
3. The system is easy to maintain and update. It provides users with the ability to upgrade features without having to manually change any settings. With this system, users can easily add new features or modify existing ones with just a few clicks of a button.
4. The system is able to learn from its users, making it more efficient over time.
5. It has the capability to recognize requests in less than 3 seconds, making it faster and more efficient than traditional methods of controlling devices.
6. User's data is never shared with third parties without explicit permission from the user. Additionally, all data collected is done so in accordance with applicable laws and regulations.

3.1.3 Use-case Diagram

Our system can control home appliances such as lights, doors, and smart locks by using voice commands from users. Besides, the homeowner has the authority to add a new user to the system and set permissions to control devices. Furthermore, the system can be controlled by using a mobile application. We can do many things of that faster, better, and more accurately in daily life.

We have designed a use-case diagram of the system which clarifies what function the system includes. That diagram illustrates how users interact with the system, what tasks are performed, and how the system responds. We also outline the various components of the system and their relationships.

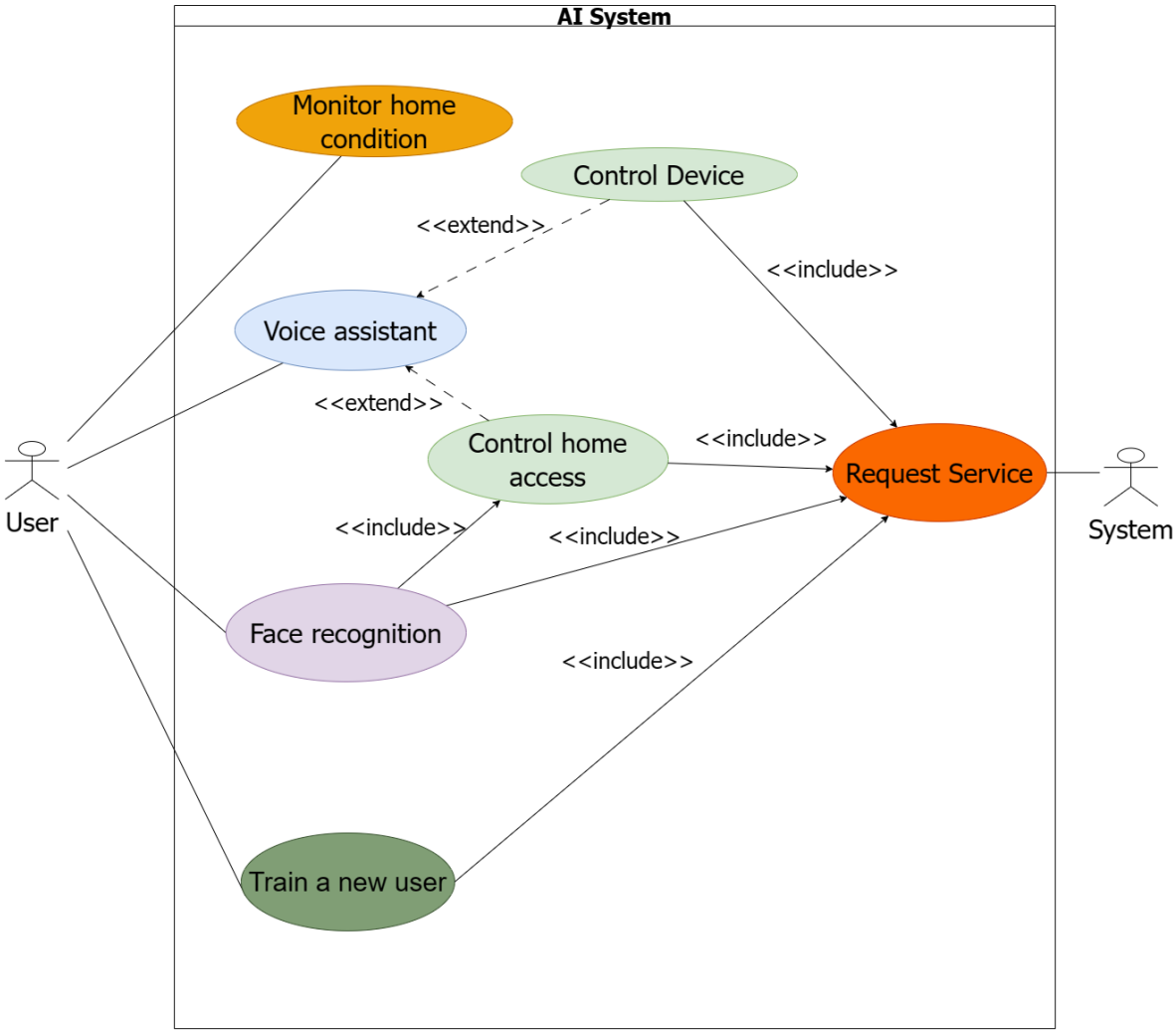

Figure 3.2: Use-case diagram of the AI system

3.1.4 Flowchart

Face recognition is one of the most important functions in our system. It enables users to quickly and easily identify who is trying to access their homes, without having to enter a code or use a key. Face recognition technology is becoming increasingly popular in smart homes, as it offers a secure way to control access to your property. It also eliminates the need for keys, which can be lost or stolen. Furthermore, it allows you to grant access to certain people while denying access to others. The flowchart below shows how face recognition in our system works:

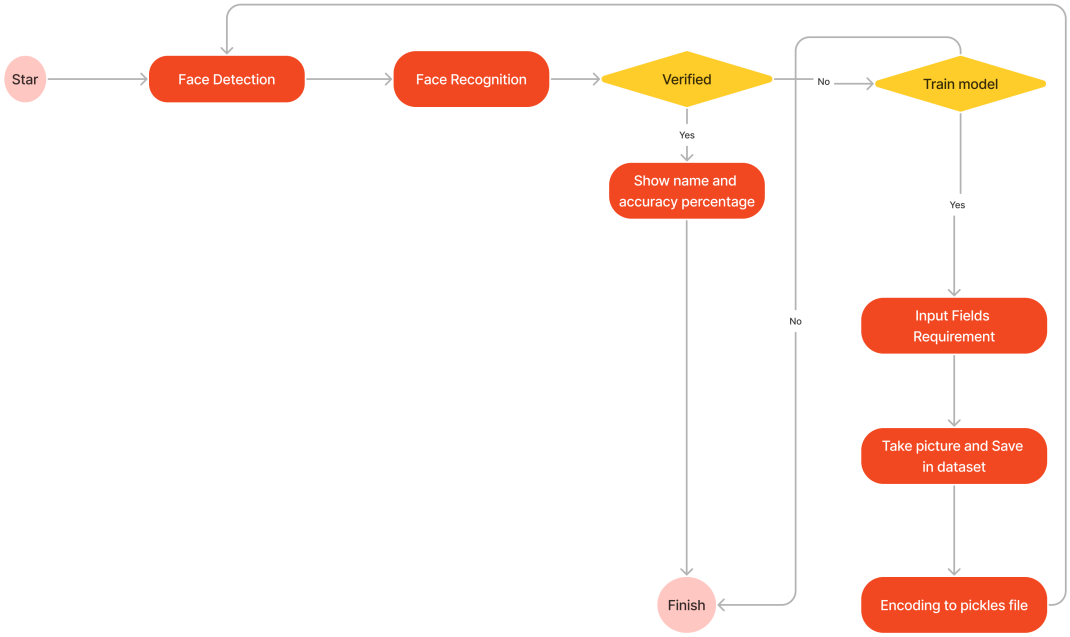

Figure 3.3: Face recognition system flowchart

In contrast to the face recognition system, the voice assistant is a great way to make your home more interesting. With voice assistant, you can control the lights, doors, and other devices in your home with just your voice. You can also ask the time, play music, and get answers to all sorts of inquiries. The flowchart below shows how the voice assistant in our system works:

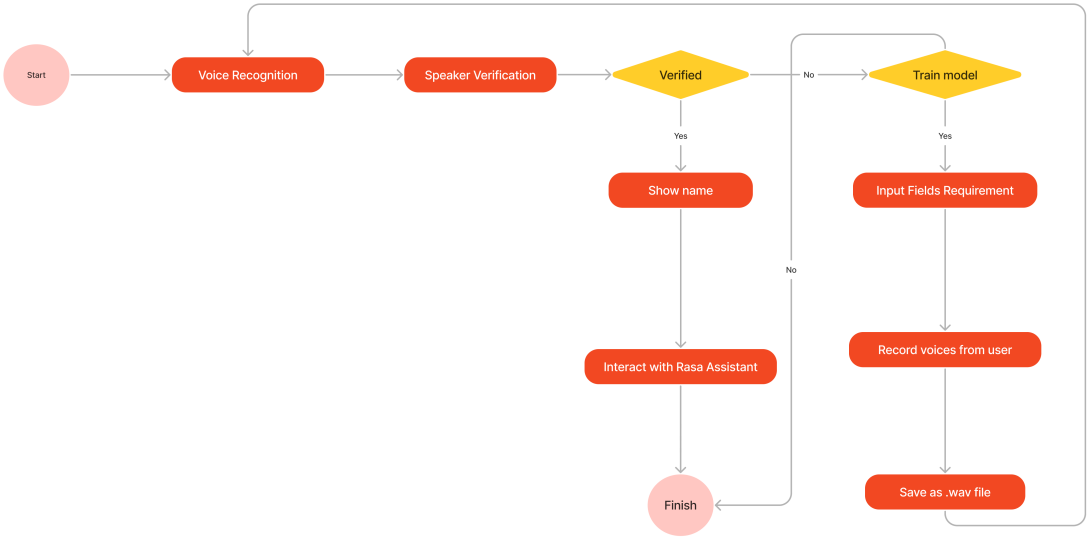

Figure 3.4: Voice assistant flowchart

3.2 Hardware Component

Hardware components play a significant role in our system since they execute our program and control the integrated AI model. Hence, below are the hardware components that we have used in our system.

3.2.1 Intel NUC [9]

The Next Unit of Computing (NUC) is a small form factor device or compute element that delivers a full desktop PC experience, a full gaming experience, or a full edge device experience. A NUC contains everything a standard PC does: processor, memory, SSD, LAN or Wi-Fi, and support for integrated and discrete graphics options. What began as a device associated with mini PCs has grown into a solution that defines what users are looking forward to next, hence the name.

Figure 3.5: Intel NUC

Our system uses sensors to collect data on various aspects of the home environment, such as temperature, humidity, and lighting levels. The data is then processed by the Intel NUC to provide personalized recommendations for the occupants of the home, such as adjusting the thermostat or turning on/off lights. The system also includes a user interface that allows the occupants to interact with the system and customize their preferences.

Intel Core CPUs, which offer high-performance computing capabilities, are found in NUCs. Moreover, a variety of connectivity options, including USB,

Ethernet, HDMI, and wireless, are included with the devices, making it simple to connect to different peripherals and networks.

Users may select the NUC that best suits their needs from a range of configurations that are offered. While some versions offer more performance and have cooling fans installed, others are built for lower power usage and have fan-less designs. This central computer will run our primary Python software, which uses advanced AI techniques, as well as manage other components in our house models, such as the lighting and fans. Also, this computer allows us to access the internet while working, which is extremely helpful for us to research and develop our system. **Table 3.1** shows the specification of Intel NUC.

GPU	Integrated Graphics
CPU	Intel Celeron Processor N3060 2.48GHz
Memory	8 GB 64-bit DDR3L-1333/1600 12.8 GB/s
Storage	64 GB
UPHY	1 USB 2.0, 1 USB 3.0
Power	6W
Price	$334 - $358

Table 3.1: Specification of Intel NUC

3.2.2 Yolo:Bit [10]

With Yolo:Bit, you can build a variety of projects, from controlling LED lights to complex IoT projects like making smart robots, building a spring automatic system, a smart home, ... or many other applications to solve problems in life. You can connect and load code for Yolo:Bit via USB cable or Bluetooth / Wi-Fi. Besides, Yolo:Bit also supports multiple programming languages such as Scratch and Python. **Figure 3.6** shows you an outlook of Yolo:Bit.

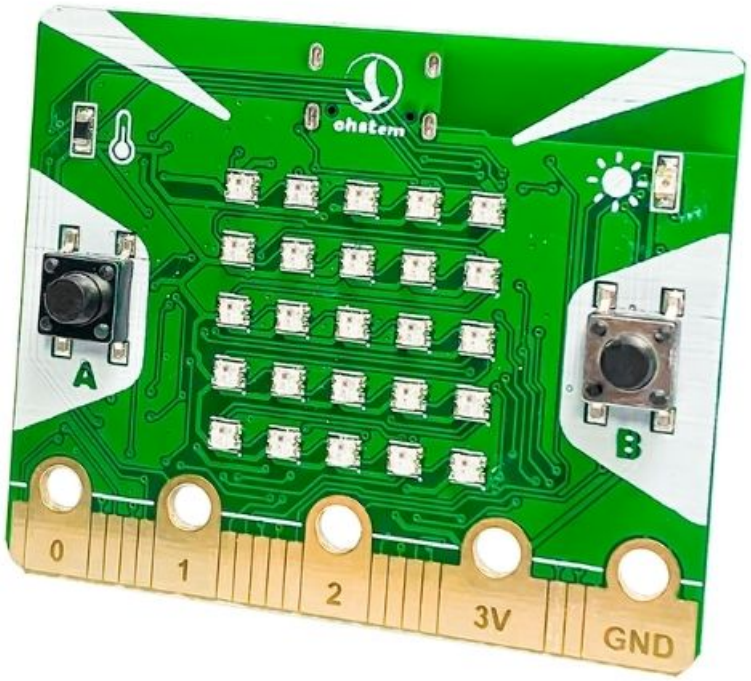

Figure 3.6: Yolo:Bit

In this project, we use Yolo:Bit to connect the DHT11 sensor. After that, we send temperature and humidity values to the Adafruit server. **Table 3.1** shows the specification of Yolo:Bit.

Power	3.7V - 6V
UPHY	USB Type C, Bluetooth
Flash Memory	4MB
RAM	8MB
Price	$18

Table 3.2: Specification of Yolo:Bit

3.2.3 Temperature & Humidity Sensor

In this system, we decided to use the DHT11 air temperature & humidity sensor because the price is relatively reasonable. Compared with other types of sensors, it has good quality, compact size, high durability, and stability. Below is a figure and some specifications.

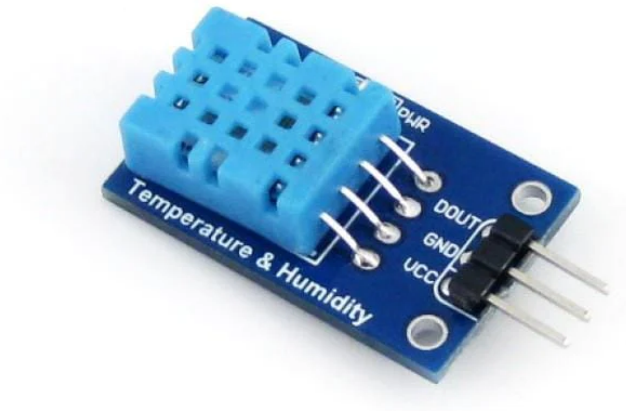

Figure 3.7: DHT11 sensor

Response time	⩽1s
Size	23mm*12mm*5mm
Humidity value	20% - 90%RH ± 5%RH
Temperature value	0 ∼ 50℃ ± 2℃
Power supply	DC 3 ∼ 5V
Frequency	1Hz
Price	$2

Table 3.3: Specification of DHT11 sensor

3.2.4 Relay

We use three SLA-12VDC-SL-A Power Relays to control the door and light. The NUC sends signals to a microcontroller, and from there, relays are triggered and transmit power to enable devices to operate. Below is a figure and some specifications.

Figure 3.8: DHT11 sensor

Maximum Amperage	30A
Power	1 HP / 120VAC
Coil Voltage	12 V
Current Type	AC
Number of Pins	4-Pin
Price	$3.63

Table 3.4: Specification of relay

3.2.5 Webcam

We use Logitech C922 Pro because it supports high resolution helping the face recognition system to capture precise images, providing high accuracy in processing and better identification. In addition, the device has a microphone which is a highlight for us to use this device because it makes the speech recognition process more accurate. Below is a figure and some specifications.

Figure 3.9: Logitech C922 Pro

Maximum Photo Resolution	1920×1080, 1920 x 1080, 2MP
Frame Rate	30fps
Connectivity	USB
Features	Autofocus, Built-in Microphone
Maximum Video Resolution	1280x720
Price	$54,97

Table 3.5: Specification of webcam

3.3 Software Component

These days, more and more software technologies help developers easier to build applications in a variety of fields. Therefore, we present the software components that we have utilized to implement the AI system and mobile application.

3.3.1 Python [11]

Python is a dynamic, interpreted (bytecode-compiled) language. There are no type declarations of variables, parameters, functions, or methods in the source code. This makes the code short and flexible, and you lose the compile-time type checking of the source code. Python tracks the types of all values at runtime and flags code that does not make sense as it runs.

Figure 3.10: Introduction to Machine Learning with Python

Python is consistent and is anchored on simplicity, which makes it most appropriate for machine learning. Due to its ability to run on multiple platforms without the need to change, developers prefer Python, unlike other programming languages. Python runs across different platforms, such as Windows, Linux, and macOS, thus requiring little or no changes. The ease of executability makes it easy to distribute software, allowing standalone software to be built and run using Python [12].

The Python programming language is a haven for most software developers looking for simplicity and consistency in their work. The Python code is concise and readable, which simplifies the presentation process. It allows developers to receive input from other developers in the community to help enhance the software or application.

Python frameworks and libraries offer a reliable environment that reduces software development time significantly. Python includes a modular machine learning library known as PyBrain, which provides easy-to-use algorithms for use in machine learning tasks. The best and most reliable coding solutions require a proper structure and tested environment, which is available in Python frameworks and libraries.

3.3.2 React Native [13]

React Native is an open-source framework for building Android and iOS applications using React and the app platform's native capabilities. With React Native, you use JavaScript to access your platform's APIs as well as to describe the appearance and behavior of your UI using React component.

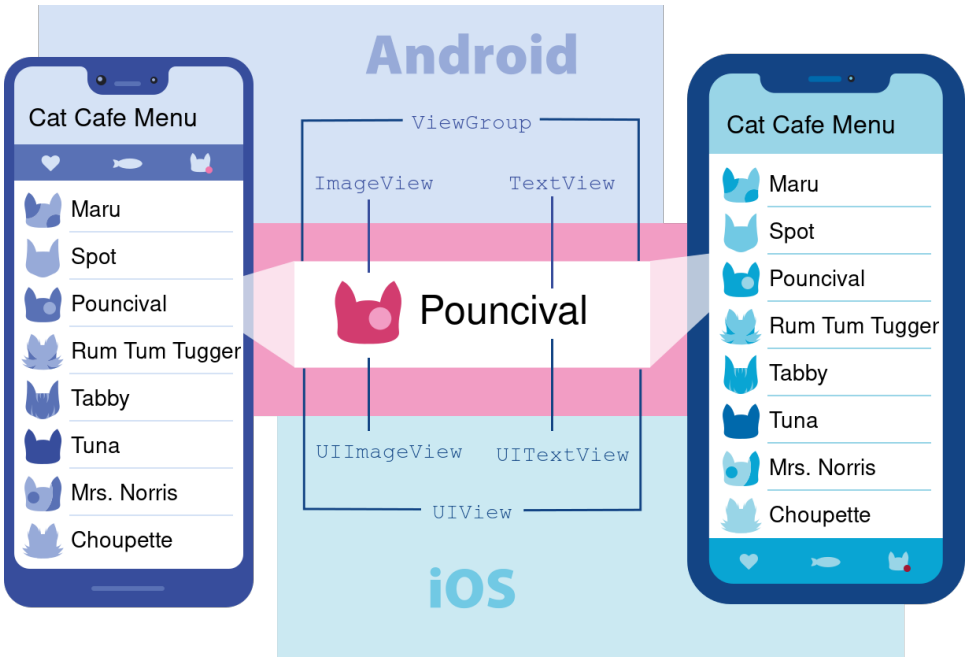

Figure 3.11: Just a sampling of the many views used in Android and iOS apps

In Android development, you write views in Kotlin or Java; in iOS development, you use Swift or Objective-C. With React Native, you can invoke these views with JavaScript using React components.

The framework works based on integrating three threads together:

- **JS thread (Javascript thread/ Main thread)**: Used by JS Engine, used to run JS bundle.
- **Native/UI thread**: Used to launch native modules, UI rendering processes, animations, and gesture handles.
- **Shadow thread**: Used to calculate the layout of the element before rendering to the screen.

Although there are limitations, being able to reuse code and save time when building, application development still brings a lot of benefits to both developers and users. Famous apps written in React Native include Facebook, Instagram, Skype, Walmart, Airbnb, SoundCloud Pulse, and Yeti Smart Home.

3.3.3 NodeJS [14]

NodeJS is an open-source and cross-platform JavaScript runtime environment. It is a popular tool for almost any kind of project. NodeJS runs the V8 JavaScript engine, the core of Google Chrome, outside the browser.

Figure 3.12: NodeJS

A NodeJS app runs in a single process, without creating a new thread for every request. When NodeJS performs an I/O action, such as reading data from the system or accessing the database system. Thanks to the non-blocking model, NodeJS will not block threads and waste CPU cycles NodeJS will continue to act when there is a response.

Some outstanding advantages of NodeJS:

- Use Javascript - an easy programming language to learn.
- Javascript ES6 supports functional programming to help apply patterns like listener or callback become simpler and easier to understand.
- Asynchronous event-driven IO, allowing multiple concurrent requests to be handled.
- There is a growing user community with active support.
- NodeJS handles thousands of connections continuously on a single server without any hassle of synchronous thread management.
- Has great scalability and can support microservices.

3.3.4 ExpressJS [15]

Express is the most popular Node web framework and is the underlying library for a number of other popular Node web frameworks. Express was initially released in November 2010 and is currently on major version 4 of the API.

Figure 3.13: ExpressJS

Express provides mechanisms to:

- Write handlers for requests with different HTTP verbs at different URL paths (routes).
- Integrate with "view"rendering engines in order to generate responses by inserting data into templates.
- Set common web application settings like the port to use for connecting, and the location of templates that are used for rendering the response.
- Add additional request processing "middleware"at any point within the request handling pipeline.

While Express itself is fairly minimalist, developers have created compatible middleware packages to address almost any web development problem. There are libraries to work with cookies, sessions, user logins, URL parameters, POST data, security headers, and many more.

3.3.5 MySQL [16]

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

Figure 3.14: MySQL

MySQL database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. Database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

MySQL databases are relational. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required, or optional, and "pointers" between different tables. The database enforces these rules so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

The MySQL Database Server is very fast, reliable, scalable, and easy to use. MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years.

The MySQL Database Software is a client/server system that consists of a multithreaded SQL server that supports different back ends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

MySQL Server has a practical set of features developed in close cooperation with our users. It is very likely that your favorite application or language supports the MySQL Database Server.

3.4 Database Design

In this project, we want to create a database that stores some necessary data to maintain in the system. The database needs to store in the table structure. Below is the structure of our database.

3.4.1 Database Structure

In terms of the smart home, we need to store its information, including users, devices, permissions, otps, schedules, and power consumption.

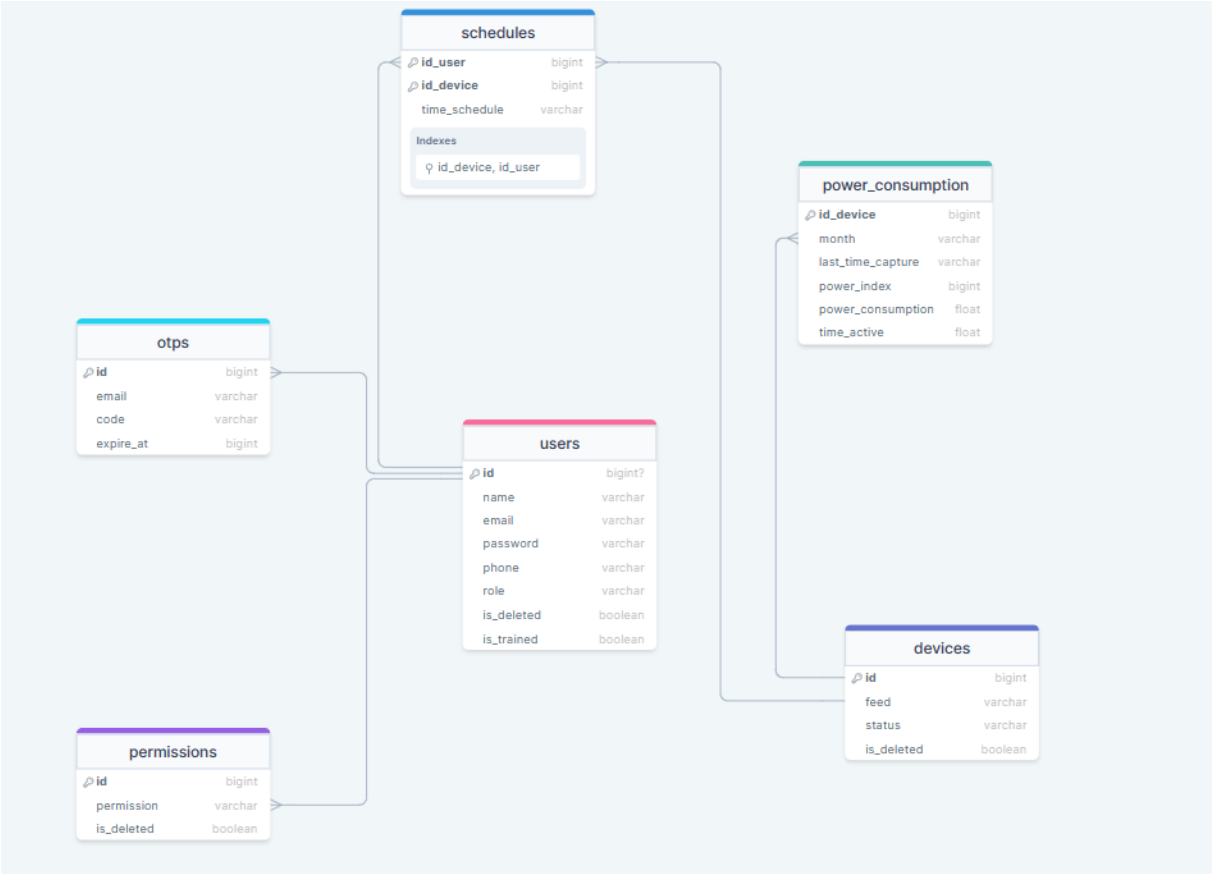

Figure 3.15: Database structure

3.4.2 Table

1. **User**

COLUMN	TYPE	LENGTH	NULLABLE
id	int	11	0
name	varchar	32	0
email	varchar	32	0
password	varchar	32	0
phone	varchar	32	0
role	varchar	None	0
is_deleted	boolean	None	0
is_trained	boolean	None	0

Table 3.6: User's schema in database

2. Device

COLUMN	TYPE	LENGTH	NULLABLE
id	int	11	0
feed	varchar	32	0
status	varchar	32	0
is_deleted	boolean	None	0

Table 3.7: Device's schema in database

3. Otp

COLUMN	TYPE	LENGTH	NULLABLE
id	int	11	0
email	varchar	32	0
code	varchar	32	0
expire_at	bigint	None	0

Table 3.8: Otp's schema in database

4. Permission

COLUMN	TYPE	LENGTH	NULLABLE
id	int	11	0
permission	varchar	32	0
is_deleted	boolean	None	0

Table 3.9: Permission's schema in database

5. Schedule

COLUMN	TYPE	LENGTH	NULLABLE
user_id	int	11	0
device_id	int	11	0
time_schedule	varchar	32	0

Table 3.10: Schedule's schema in database

6. Power Consumption

COLUMN	TYPE	LENGTH	NULLABLE
id_device	int	11	0
month	varchar	32	0
last_time_capture	varchar	32	0
time_active	float	None	0
power_index	bigint	None	0
power_consumption	float	None	0

Table 3.11: Power consumption's schema in database

3.4.3 Entity Relational Diagram

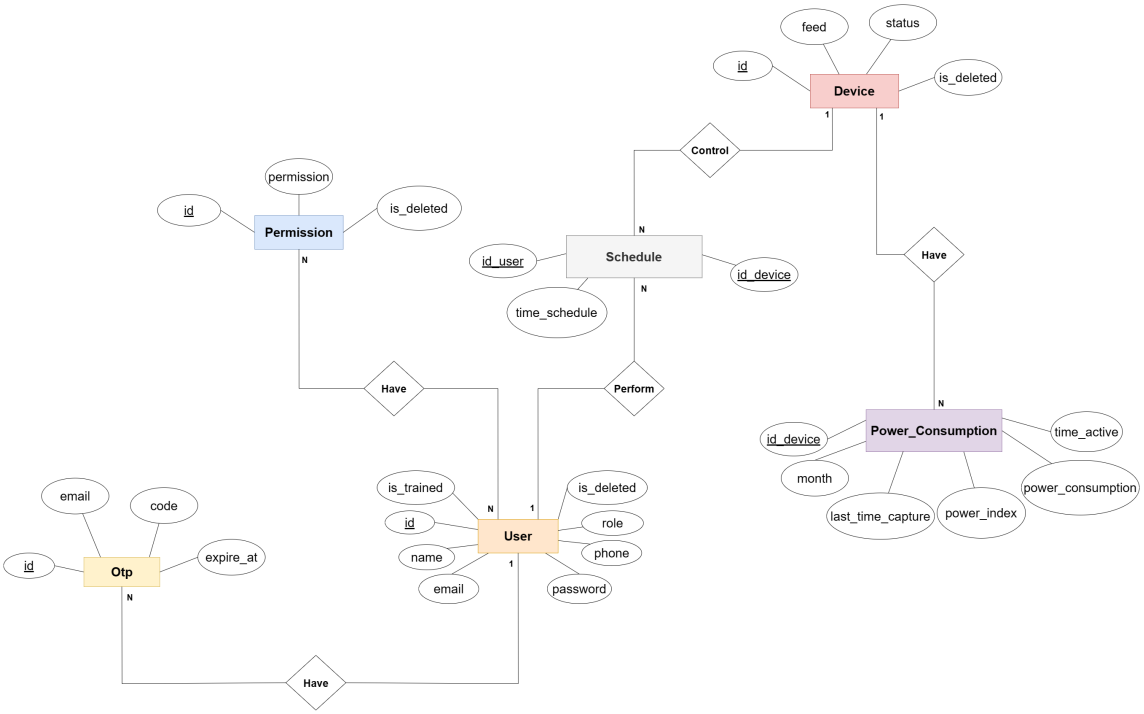

Figure 3.16: Entity Relation Diagram

3.5 Dashboard

Adafruit IO is a cloud service that is meant primarily for storing and then retrieving data. Adafruit IO supports different hardware like Raspberry PI, ESP2866, and Arduino [17].

Adafruit IO can display our data in real-time, online, make our project internet-connected, control motors, read sensor data, and more, and finally connect our project to other internet-enabled devices. Adafruit IO can handle and visualize multiple feeds of data.

In this system, we have designed a dashboard on Adafruit IO like below.

Figure 3.17: The dashboard on Adafruit IO

The layout for the dashboard consists of 3 parts:

Part 1 contains line graphs that have a history of the sensor's value and blocks showing the temperature and humidity.

Part 2 contains blocks that show the device's state (light and door) and a panel that announces the name of the person who just entered the house.

Part 3 will have a panel showing the recent events of the system.

3.6 Mobile Application

3.6.1 System Architecture

We build the application on reliable design patterns. Apply the client-server architecture, client-side includes the mobile application. On the server side including the HTTP to handle requests from users and API system from third parties. The server system's blocks are clearly divided into layers. The layers communicate with each other through the interface the application is easy to maintain layer by layer without affecting other components of the application. The system includes the following layers:

- **Security layer**: plays the role of a filter that authenticates requests coming from the client-side.
- **Rest controller layer**: provides endpoints for clients to communicate.
- **Business service layer**: is where the business-related logic of the platform resides. This layer intersects with the repository or service layers to handle requests from the user.
- **Repository**: acts as an interface for the layers above to communicate with the database.
- **3rd API service**: is a wrapper for calling 3rd party services.

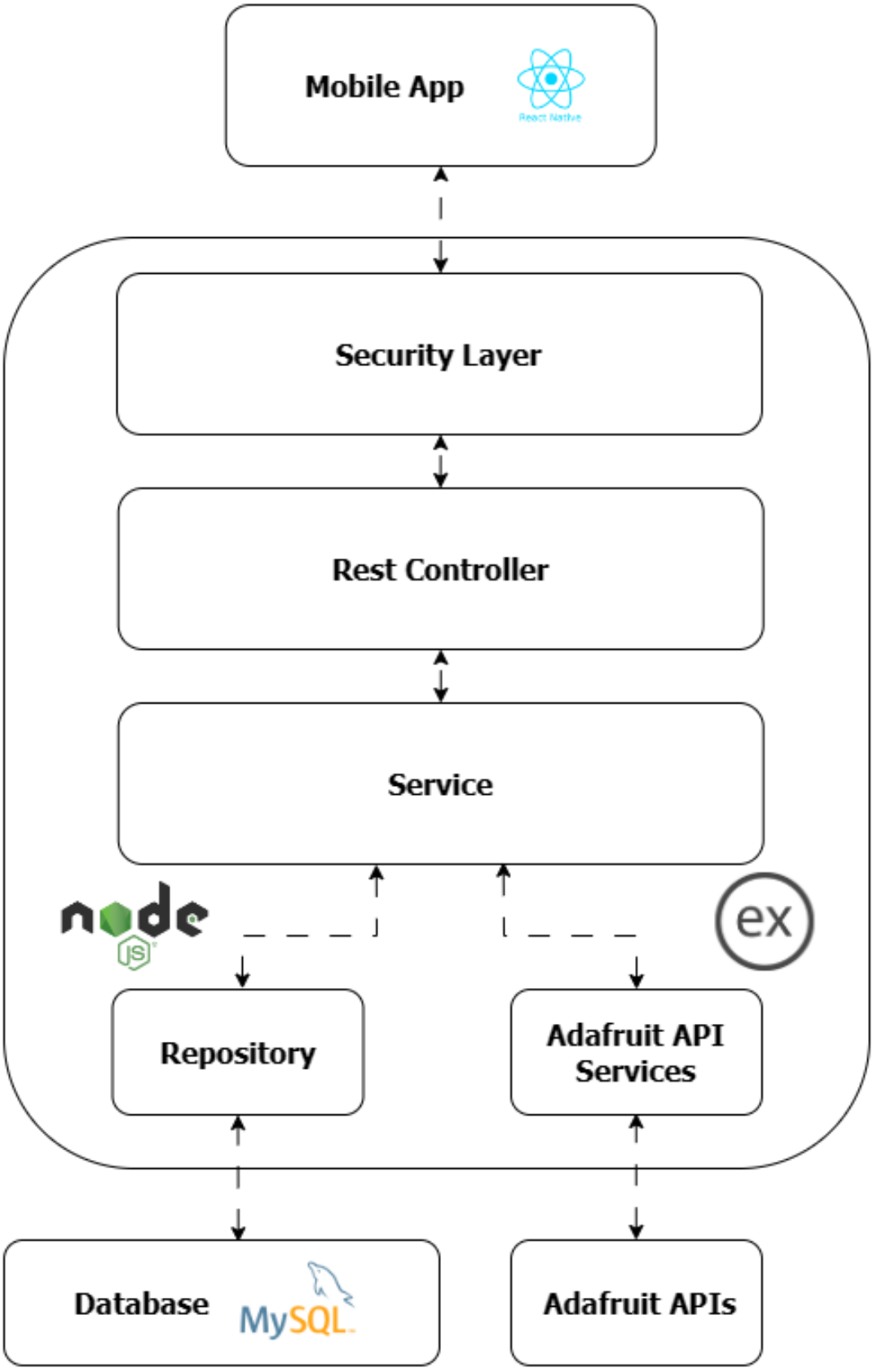

Figure 3.18: Mobile application architecture

3.6.2 Functional Requirement

When developing mobile applications, it is important to consider functional requirements :

- **Authentication**: Allows users to create and log in to their accounts using a secure authentication process, such as email and password or social media accounts.
- **Control devices**: Enables users to control their smart home devices, such as lights, thermostats, and security cameras, through the app. They can turn on/off devices, adjust settings, and monitor device status remotely.
- **Scheduling**: Allows users to set schedules for their smart home devices to turn on/off automatically at specific times or on specific days. This can help them save energy and make their home more convenient.
- **Manage users**: Allows the host to invite new members to the smart home network and sets different permission levels for each member. For example, the host can grant access to certain devices or limit control based on specific settings.
- **Tracking**: Provides users with real-time data on power consumption, temperature, and humidity levels in their homes. This data can help users optimize their energy usage and maintain a comfortable environment.

3.6.3 Non-functional Requirement

There are a few non-functional requirements of mobile applications that need to be taken into account:

- The app is capable of running 24/7 without any failures, making it one of the most reliable solutions available today.
- It works on multiple platforms like Android and iOS, making it easy for users to access their devices from any device.
- The app should be intuitive and easy to use, with a clear and consistent user interface that makes it easy to navigate and control smart home devices.

3.6.4 Use-case Diagram

- **Authentication**

Name	Authentication
Actor	Host and member
Description	Verifying the identity of a user who is trying to access application
Trigger	When member login to the application or create an account or reset password
Pre-condition	Member or Host in the home page of app
Post-condition App	The App validates the user's system role and allows whether member access the mobile app
Normal flow	1. The user (host, member) initiates the login process by entering their login credentials. 2. The system or application receives the user's credentials and verifies them against a database. 3. If the user's credentials are valid, the system or application grants the user access to the system or application. 4. Once the user is authenticated, they can continue using the system or application until they log out or their session expires.
Alternative flow	1.a: The user clicks the "Sign up" button to create an account with the valid token from the host. 3.a: If you don't remember your password, you may reset it by clicking "Forgot password".
Exception flow	4.a: The system checks for incorrect login information and show error field.

Table 3.12: Usecase - Authentication

- **Control devices**

Name	Control devices in smart home
Actor	Host and member
Description	A smart home provide users with convenient and intuitive control over their home devices
Trigger	The user click to "switch" button each rooms
Pre-condition	The user redirect to home or dashboard screen to adjust devices
Post-condition	Devices will adjust to some modifications made by user
Normal flow	1. The user initiates the control process by accessing the smart home application and redirect to room to control your devices. 2. The user click to "switch" button to change state of device. 3. The device may also send status updates back switch to the application or device to inform the user of the current state of the device.

Table 3.13: Usecase - Control devices

- **Schedule time for devices**

Name	Schedule time for devices of each room
Actor	Host and member
Description	The user can set up automated schedules or routines to control the device based on specific condition, Host
Trigger	The user click to "set schedule" button in the DetailDevice screen
Pre-condition	The user login to the app and redirect to DetailScreen
Post-condition	The device will be set schedule to change status on time
Normal flow	1. The user initiates the process by accessing the smart home application and device and selecting the device they want to schedule 2. The application presents the user with options for scheduling the device, setting the date and time for the device to turn on or off. 3. The user selects the desired date and time, and the application or device saves the schedule and sends a command to the device to adjust its settings accordingly. 4. The device receives the command and updates its settings to reflect the scheduled on/off times. 5. When the scheduled time arrives, the device turns on or off automatically according to the user's preferences.

Table 3.14: Usecase - Schedule time for devices

• Tracking power consumption, temperature, and humidity

Name	Tracking power consumption, temperature, and humidity
Actor	Host and member
Description	User can view the electricity consumption of each month and the line graph of temperature and humidity
Trigger	User clicks the "Statistic" icon on the "Navigation" bar
Pre-condition	The system is running, the user's device is connected to the Internet network and logged into the system.
Post-condition	The system displays the information successfully
Normal flow	1. User clicks the "Statistic" icon on the "Navigation" bar 2. User clicks the "Electricity" tab 3. The system displays the information successfully
Alternative flow	2.a: User clicks the "Temperature & Humidity" tab

Table 3.15: Usecase - Tracking power consumption, temperature, and humidity

• Manage permission for a member

Name	Manage permission for a member
Actor	Host
Description	Host manages the permissions of members in the system
Trigger	Host clicks the icons "Add" or "Remove" permission of each users
Pre-condition	The system is running, the host's device is connected to the Internet network and logged into the system. The host is on the "Manage Member" page
Post-condition	The system displays the member's permission after
Normal flow	1. Host clicks the "Select" button 2. The system displays a list of available options in the system 3. At this step, the host can perform the following actions: a). Add permissions for a member b). Remove added permissions for a member

Table 3.16: Usecase - Manage permission for a member

- **Invite a new member**

Name	Invite a new member
Actor	Host
Description	Host sends an invitation to a member to join the system
Trigger	Host click the “Add” button on the “Manage Member” page
Pre-condition	The system is running, the host’s device is connected to the Internet network and logged into the system. The host is on the “Add Member” page
Post-condition	Invited member receives an email containing the OTP code used to register an account
Normal flow	1. Host enters the member’s email and clicks the “Add” button 2. The system displays a pop-up to confirm the action 3. Host clicks the “Accept” button to agree 4. The system displays a successful status message of sending mail
Alternative flow	3.a: Host clicks the “Cancel” button to cancel
Exception flow	4.a: The system displays a failed status message of sending mail

Table 3.17: Usecase - Invite a new member

3.6.5 Sequence Diagram

- **Authentication**

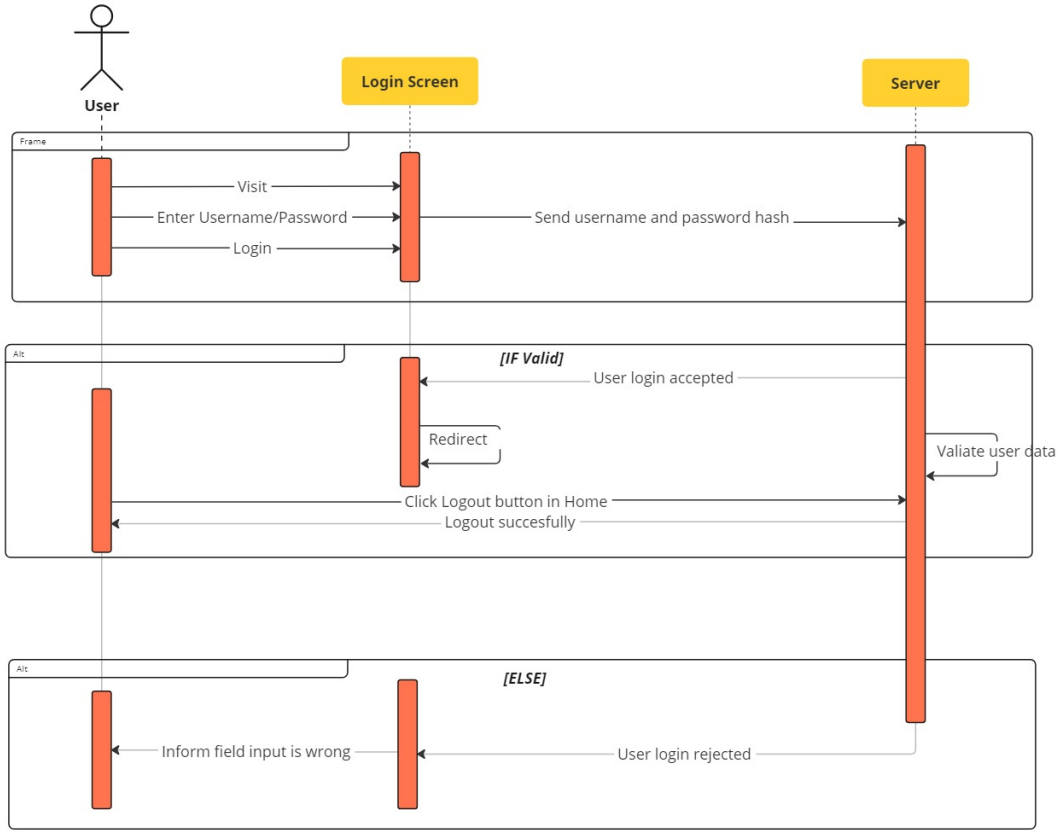

Figure 3.19: Sequence diagram of login

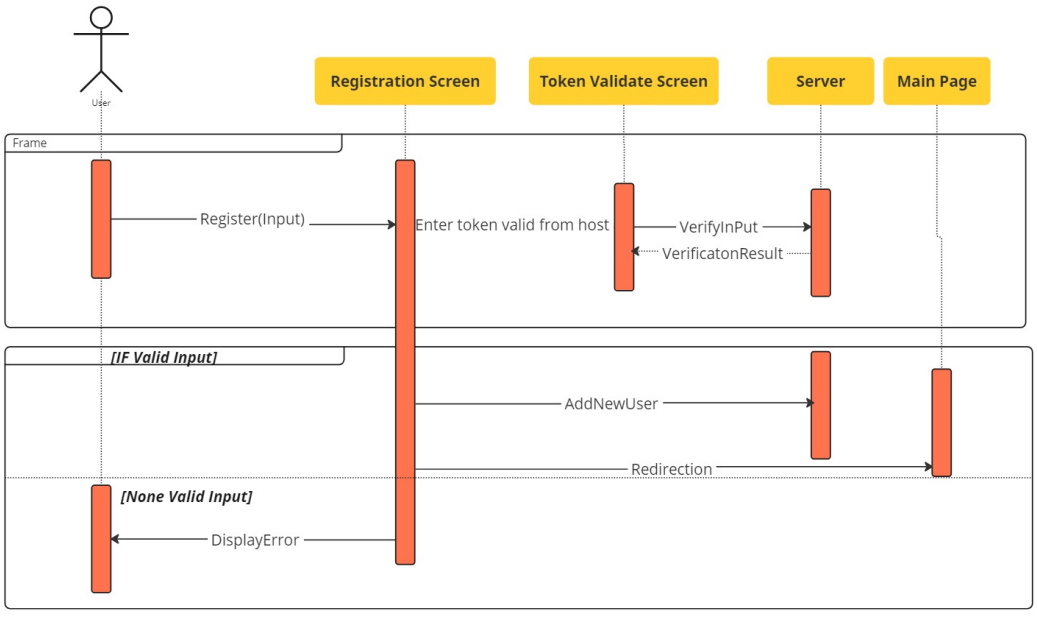

Figure 3.20: Sequence diagram of register

- **Control devices**

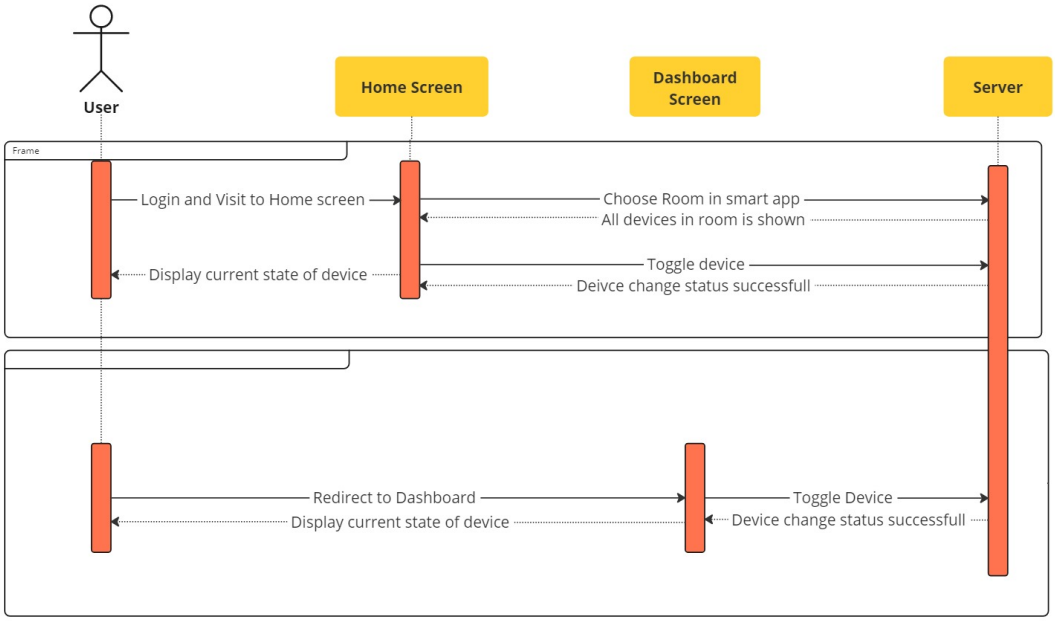

Figure 3.21: Sequence diagram of control devices

- **Schedule for devices**

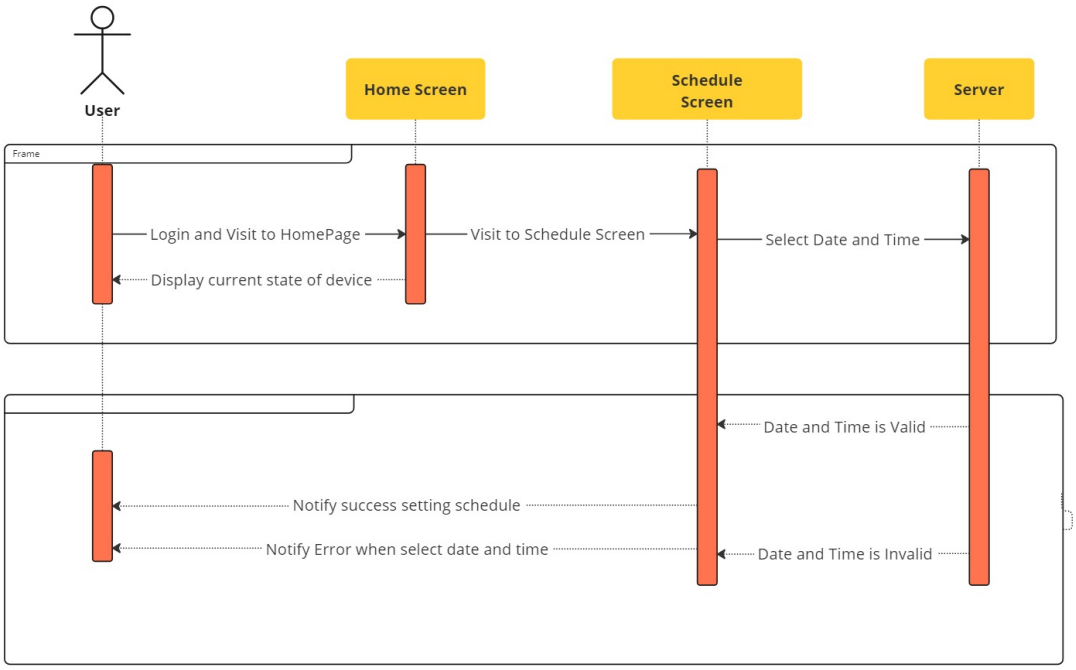

Figure 3.22: Sequence diagram of schedule for devices

- **Tracking electricity consumption, temperature, and humidity**

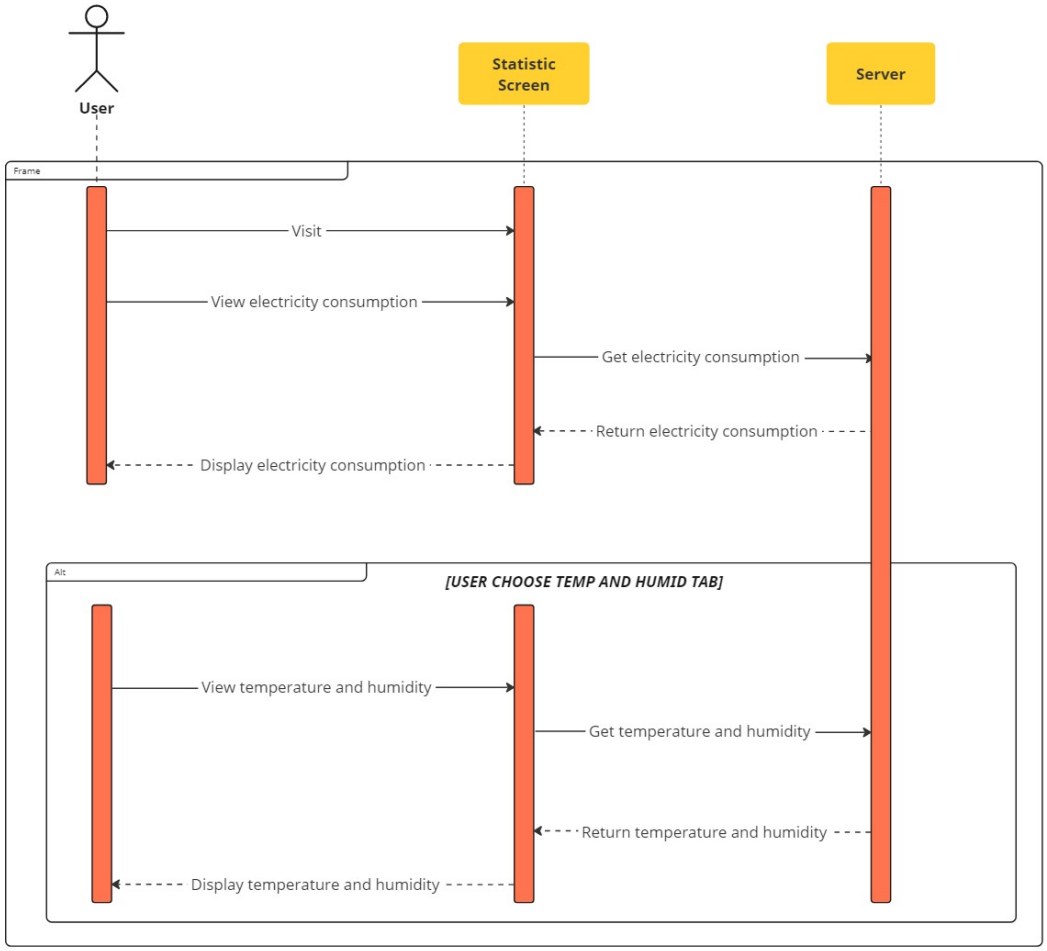

Figure 3.23: Sequence diagram of tracking electricity consumption, temperature, and humidity

- **Manage permission for a member**

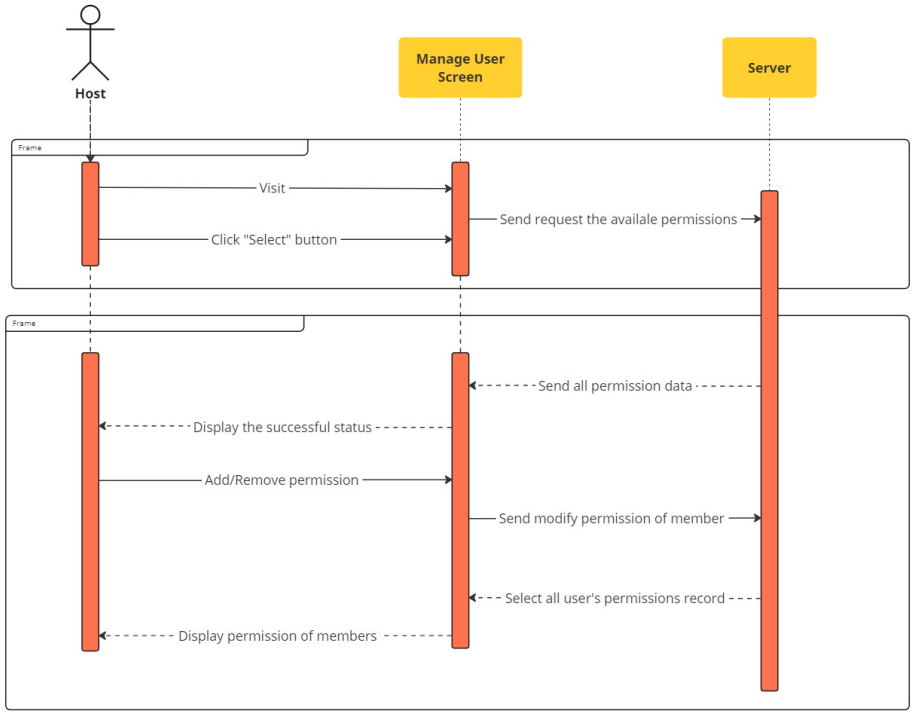

Figure 3.24: Sequence diagram of manage permission for a member

- **Invite a new member**

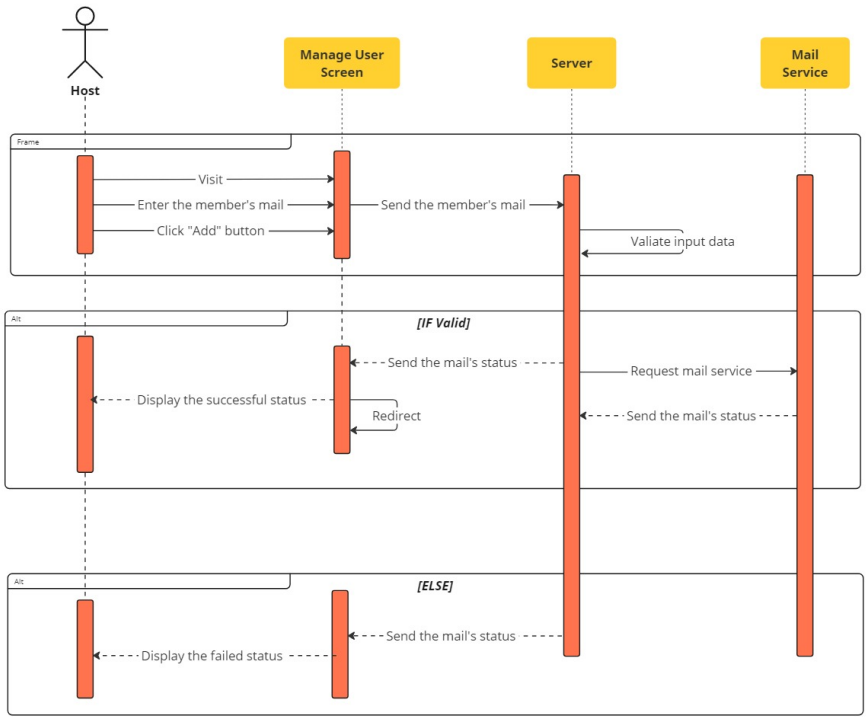

Figure 3.25: Sequence diagram of invite a new member

CHAPTER 4

IMPLEMENTATION

This chapter describes the detailed implementation of the AI system, the Mobile application, and how to set up the gateway with Adafruit.

4.1 Face Recognition

The deep learning-based facial embeddings that we used are both highly accurate and capable of being executed in real-time. In this section, we show you how to perform face recognition using OpenCV, Python, and deep learning. To create a complete system of face recognition, we must work on 3 very distinct phases in the following.

4.1.1 Data Gathering

Starting from the first step - Face Detecting, we will simply create a dataset, where we will store for each name, a group of photos with the portion that was used for face detection.

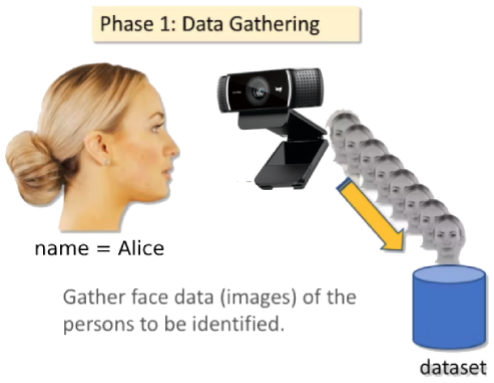

Figure 4.1: Create a dataset for face recognition system

The most basic task on face recognition is *"Face Detecting"*. Before anything, you must capture a face in order to recognize it, when compared with a new face captured in the future.

Next, create a sub-directory where we will store our facial samples and name it *"training_data"*:

The most common way to detect a face, is using the *"Haar Cascade classifier"*

```
faceCascade = cv2.CascadeClassifier('Cascades/haarcascade_frontalface.xml')
```

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, *"Rapid Object Detection using a Boosted Cascade of Simple Features"* in 2001. It is a machine learning-based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

After detecting the face, we need to capture the image to save it in the dataset using OpenCV. What we added, was an *"input command"* to capture a user name, which should be a string:

```
    name = input("Enter your name: ")
```

For each one of the captured frames, we save it as a file on a *"data"* folder:

```
    img_name = "data/faces/{}/{}.png".format(name, img_counter)
    cv2.imwrite(img_name, frame)
```

4.1.2 Train Model for Face Recognition

In this second phase, we must take all user data from our dataset and *"trainer"* the recognizer. This is done directly by a specific *face_recognition* function. The result will be a .pickle file that is saved on *"trainer/"* directory.

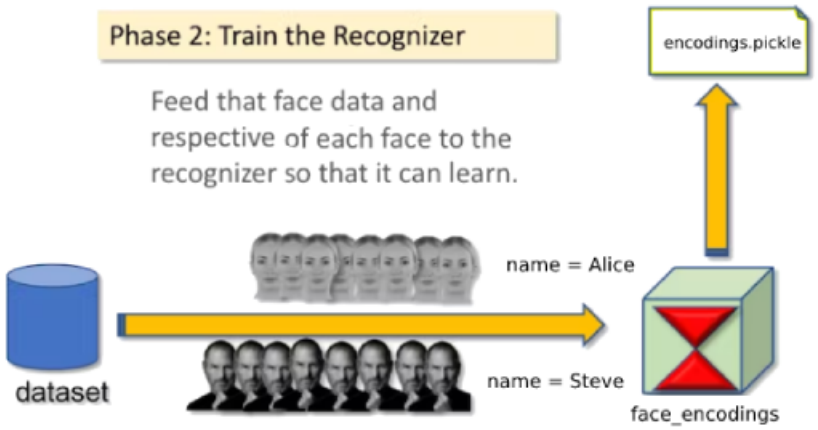

Figure 4.2: Encoding dataset

We will loop through all images from the dataset to train the model. Besides, OpenCV orders color channels in BGR, but the dlib actually expects RGB. The *face_recognition* module uses dlib, so before we proceed, let's swap color spaces by naming the new image rgb.

```
# load the input image and convert it from BGR (OpenCV ordering)
# 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="cnn")
# compute the facial embedding for the face
encodings = face_recognition.face_encodings(rgb, boxes)
```

Next, for each iteration of the loop, we're going to detect a face and encode (possibly multiple faces and assume that it is the same person in multiple locations of the image). We use two methods of *face_recognition*:

- **face_location**: find/localize the faces in a list of face. We pass two parameters to the *face_locations* method:
  - rgb: Our RGB image.
  - model: Either CNN or HOG. The CNN method is more accurate but slower. HOG is faster but less accurate.

- **face_encoding**: encoding the face into a vector which knows as turning the bounding boxes of the face into a list of 128 numbers.

Finally, we will export an encoding list of faces to a encoding.pickle file:

```
# 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(name)
    # dump the facial encodings + names to disk
    print("[INFO] serializing encodings...")
    data = {"encodings": knownEncodings, "names": knownNames}
    f = open(args["encodings"], "wb")
    f.write(pickle.dumps(data))
    f.close()
```

As you can see from our output, we now have a encodings.pickle file that contains the 128-d face embeddings for each face in our dataset.

```
[INFO] processing image 1/10
[INFO] serializing encodings...
[INFO] processing image 2/10
[INFO] serializing encodings...
[INFO] processing image 3/10
[INFO] serializing encodings...
[INFO] processing image 4/10
[INFO] serializing encodings...
[INFO] processing image 5/10
[INFO] serializing encodings...
[INFO] processing image 6/10
[INFO] serializing encodings...
[INFO] processing image 7/10
[INFO] serializing encodings...
[INFO] processing image 8/10
[INFO] serializing encodings...
[INFO] processing image 9/10
[INFO] serializing encodings...
[INFO] processing image 10/10
[INFO] serializing encodings...
 pop_os_hi  Smart-Home  (face-recognition 1)  16:03  ls -lh encodings*
-rw-rw-r-- 1 pop_os_hi pop_os_hi 11K Dec 15 16:03 encodings.pickle
 pop_os_hi  Smart-Home  (face-recognition 1)  16:03
```

Figure 4.3: Encoding face from dataset

4.1.3 Face Recognition System

Now, we reached the final phase of our project. Here, we will capture a fresh face on our camera and if this person had his face captured and trained before. Because we have created our 128-d face embeddings for each image in our dataset, we are now ready to recognize faces in images using OpenCV, Python, and deep learning.

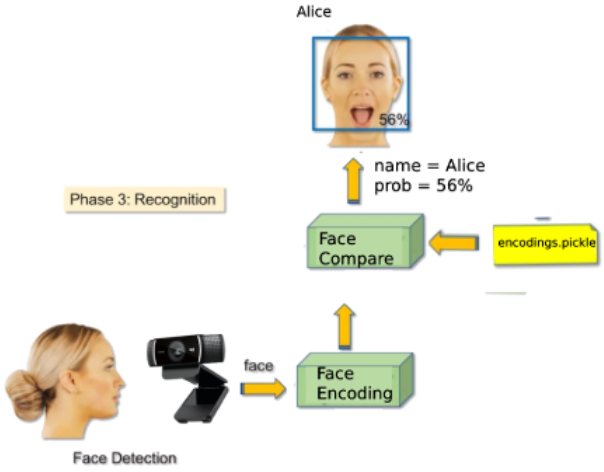

Figure 4.4: Recognizing faces in real-time

We attempt to match each face in the input image (encoding) to our known encoding dataset using the method *face_recognition.compare_faces*:

```
matches = face_recognition.compare_faces(data["encodings"], encoding, 0.54)
```

This function returns a list of True/False values, one for each image in our dataset. Internally, the *compare_faces* function is computing the Euclidean distance between the candidate embedding and all face in our dataset:

- If the distance is below some tolerance (the smaller the tolerance, the more strict our facial recognition system will be) then we return True, indicating the faces match.
- Otherwise, if the distance is above the tolerance threshold we return False as the faces do not match.

Essentially, we are utilizing a *"more fancy"* k-NN model for classification. We use a model CNN face detector to accelerate face recognition and be more accurate.

Finally, below is the result of the system in real-time:

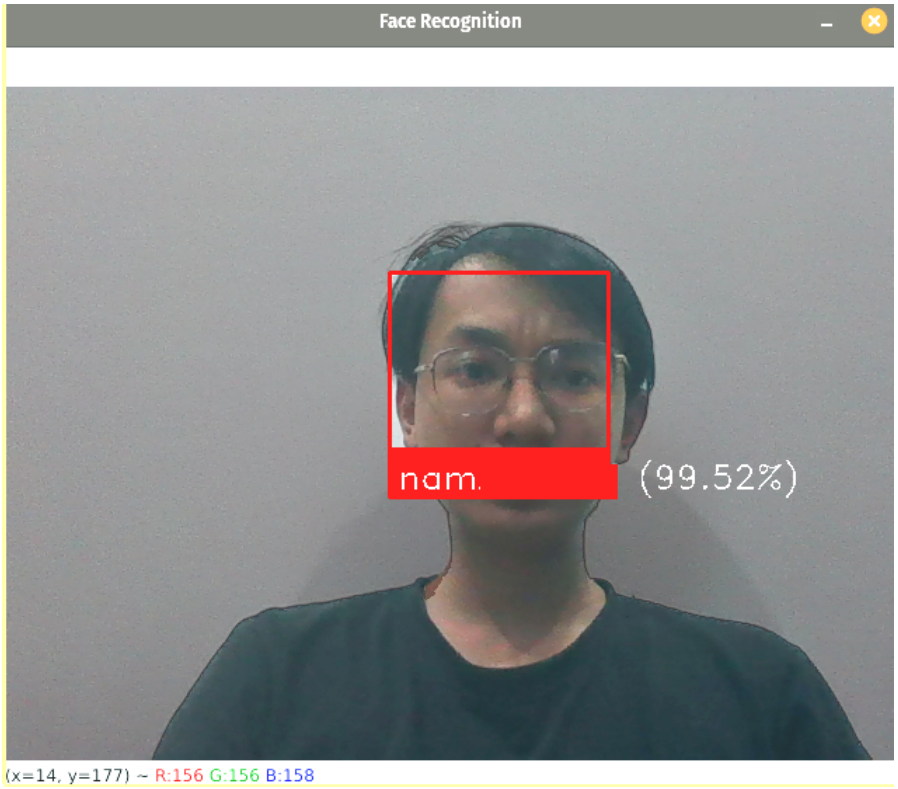

Figure 4.5: Face Recognition recognizes user's face

4.2 Voice Assistant

4.2.1 Speaker Verification

The speaker verification process includes 3 steps: collecting data, extracting features, training model, and testing model.

Collecting data

In this step, the user will enter the name and proceed to record the voice. Each user will have 5 recording files, which are saved as .wav files. Each recording file will be 5 seconds long. All these files will be saved in the *training_data* folder to train the model.

This process will be performed by function *record_audio* below:

```
def record_audio():
Name = (input("Please Enter Your Name: "))
for count in range(5):
    FORMAT = pyaudio.paInt16
    CHANNELS = 1
```

```
    RATE = 44100
    CHUNK = 512
    RECORD_SECONDS = 5
    ...
    print("----------Recording----------")
    Recordframes = []
    for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
        data = stream.read(CHUNK)
        Recordframes.append(data)
    print("----------Recording Stopped----------")
...
OUTPUT_FILENAME = Name + "-sample" + str(count) + ".wav"
WAVE_OUTPUT_FILENAME = os.path.join("trainingData", OUTPUT_FILENAME)
...
waveFile.close()
```

Extracting Features from Audio

The idea is to make the correct identification of the speaker by using the Gaussian mixture model. The first step while dealing with an audio sample is to extract the features from it i.e. to identify components from the audio signal. We are using the Mel frequency cepstral coefficient (MFCC) to extract the features from the audio sample. MFCC maps the signal onto a non-linear Mel-Scale that mimics human hearing and provides the MFCC feature vectors which individually describe the power spectral envelope of a single frame.

We considered the MFCC with tuned parameter as a primary feature and delta MFCC that is also known as differential and acceleration coefficients which are used to deal with speech information that is related to dynamics i.e. trajectories of MFCC coefficient over time it turns out to be the calculation of these trajectories.

This process will be performed by function *calculate_delta* and function *extract_features*:

```
def calculate_delta(array):
    rows, cols = array.shape
    deltas = np.zeros((rows, 20))
    N = 2

    for i in range(rows):
        ...
        deltas[i] = ...

    return deltas

def extract_features(audio, rate):
    ...
    delta = calculate_delta(mfcc_feature)
    combined = np.hstack((mfcc_feature,delta))
    return combined
```

Training Model

Gaussian Mixture Model (GMM) is one of the most popular models used for training while dealing with audio data so we used MFCC and GMM together to achieve the target of identifying the speaker correctly. GMM is used to train the model on extracted features.

```
def train_model():
    src = "trainingData/"
    dest = "models/"
    train_file = "training_set.txt"
    file_paths = open(train_file, 'r')

    count = 1
    features = np.asarray(())

    for path in file_paths:
        ...
        if count == 5:
            gmm = GaussianMixture(n_components = 6, max_iter = 200,
  covariance_type=...)
            gmm.fit(features)
```

```
            # dumping the trained gaussian model
            picklefile = path.split("-")[0] + ".gmm"
            ...
            count = 0
        count = count + 1
```

Testing Model

GMM models will be used to calculate the scores of the features for all the models. The speaker model with the maximum score is predicted as the identified speaker of the test speech.

The argument passed to function *test_model* will be the file stored as .wav and the result will be the name that the model identifies.

```
def test_model(fileName):
    dest = "models/"
    ...
    # load the Gaussian gender Models
    models = [pickle.load(open(fname, 'rb')) for fname in gmm_files]
    speakers = [fname.split("\\")[-1].split(".gmm")[0] for fname in
  gmm_files]

    # read the test directory and get the list of test audio files
    sr, audio = read(fileName)
    vector = extract_features(audio, sr)

    log_likelihood = np.zeros(len(models))
    for i in range(len(models)):
        gmm = models[i]
        scores = np.array(gmm.score(vector))
        log_likelihood[i] = scores.sum()

    winner = np.argmax(log_likelihood)
    winner_name = speakers[winner][7:]
    return winner_name
```

4.2.2 Extract entities

The goal of this service is that when the user talks with the virtual assistant, the system must determine what the intent is and then make a response to the user. Once the intent is known, the system must filter out the entities (including actions, devices, and city names). With the simple requirements of the smart home, our system has been trained to be able to identify a total of six intents, respectively:

Greet, Bye, Affirm, Deny, Inquire Time, Control Device

The first four intent is primarily used to support constructing the conversation with the user. Such as, when the user says "Bye", the system can communicate with a user based on intent extracted from that conversation. While the remaining two Intent, *inquire time* and *control device*, are used to predict the action of the user.

Suppose that we have the sentences of the user as follows:

- *"Bây giờ là mấy giờ?"*
- *"Mở giùm mình cây quạt"*
- *"Tắt giùm tôi cây đèn"*

The system will return *inquire time*, and *control device* intent with the above three sentences, respectively. When the intent *inquire time* and *control device* are identified, the system has to extract their entities as described below.

Display entities:

- **List of commands**: {"mở", "tắt"}
- **List of device**: {"quạt", "đèn"}

Action: {"Mở quạt", "Tắt đèn"}

4.2.3 Train Natural Language Understanding

Initiate The Project

First, we run the *rasa init* command to create the default project. Here, we consider two essential files that are needed to train the assistant, **config.yml** and **data/nlu.yml**.

- **config.yml**: Store the configuration for training the assistant, including *pipeline components*, and *tokenizers*.
- **nlu.yml**: Contains the training data for *intents classification* and *entities extraction*.

Prepare Training Data

Now, we will prepare the training data for the system, which is stored in the *data/nlu.yml* file. The following **Table 4.1** represents the intents and entities we want the model to classify and extract.

Intent	**Entities**
- greet	- device
- bye	- command
- affirm	- place
- deny	
- inquire time	
- control device	

Table 4.1: Intents and Entities need to be extracted by model

For the *intents ("greet", "goodbye", "affirm", "deny")*, we specify the common examples for *"greet"* and *"goodbye"* in Table 4.2.

- intent: greet	**- intent**: goodbye
examples: |	textbfexamples: |
- chào	- gặp lại sau
- xin chào	- tạm biệt
- chào anh	- chào tạm biệt
- chào chị	- hẹn gặp lại
- chào bạn	- lúc khác gặp lại
- chào buổi sáng	- về đây
- chào buổi tối	- good by
- chào buổi chiều	- cee you later
- chào em	- good night
- chào đằng ấy	- bye
	- goodbye

Table 4.2: Data Training For Intent Greet and Goodbye

We list some examples for training *inquire_time* and *control_device* intents. Furthermore, we provide lists of *lookup table* which contains vocabularies of three *entities ("device", "command", "place")*. All of the training data for each part above are described in **Table 4.3**, and **Table 4.4**, respectively.

- intent: inquire_time
examples: |
- Bây giờ là mấy giờ?
- [Việt Nam](**place**) bây giờ là mấy giờ?
- [London](**place**) bây giờ là mấy giờ?
- [New York](**place**) bây giờ là mấy giờ?

Table 4.3: Data Training For Intent Inquire Time

- intent: control_device
examples: |
- [tắt](**command**) [đèn](**device**)
- [mở](**command**) [đèn](**device**)
- [bật](**command**) [đèn](**device**)
- [tắt](**command**) [cửa](**device**)
- [mở](**command**) [cửa](**device**)
- [bật](**command**) [cửa](**device**)

Table 4.4: Data Training For Intent Control Device

Create Configure File for Training

In the *config.yml* file, we modify the content as **Figure 4.27** below. We use the *WhitespaceTokenizer* pipeline as our Tokenizer. For extracting the entities, we use both pipeline *RegexEntityExtractor* and *CRFEntityExtractor*.

Finally, we use the *DIETClassifier* pipeline to classify the user's intent.

pipeline:
- **name**: WhitespaceTokenizer
- **name**: RegexEntityExtractor
- **name**: RegexFeaturizer
- **name**: CRFEntityExtractor
- **name**: LexicalSyntacticFeaturizer
- **name**: CountVectorsFeaturizer
- **name**: CountVectorsFeaturizer
  analyzer: char_wb
  min_ngram: 1
  max_ngram: 4
- **name**: DIETClassifier
  epochs: 100
  constrain_similarities: true
- **name**: EntitySynonymMapper
- **name**: ResponseSelector
  epochs: 100
  constrain_similarities: true
- **name**: FallbackClassifier
  threshold: 0.3
  ambiguity_threshold: 0.1

Table 4.5: Configure pipeline

Train and Run The Assistant

After preparing the appropriate data and configuration file for the system, we start training the assistant. Then, with the output of the training step, we run the service by deploying the model locally with the URL equal *"http://localhost:5002/api"*. Whenever there is a user request, the Device Server runs an HTTP request to achieve the response as a JSON at the above URL.

The below example displays the output of a user's request.

```
{
  "text": "Tắt giùm mình cây đèn",
  "intent": {
     "name": "control_device",
     "confidence": 0.9988497495651245
  },
  "entities": [
   {
     "entity": "command",
     "confidence_entity": 0.9963920712471008,
     "value": "Tắt",
     "extractor": "DIETClassifier"
   },
   {
     "entity": "device",
     "confidence_entity": 0.9950836896896362,
     "value": "đèn",
     "extractor": "DIETClassifier"
   }
  ],
  "intent_ranking": [
   {
     "name": "control_device",
     "confidence": 0.9988497495651245
   },
   {
     "name": "inquire_time",
     "confidence": 0.00031712057534605265
    }
  ]
}
```


Table 4.6: Result

- **text**: The input as well as the user's message.
- **intent**: The final intent that is classified by the system when inferring to the model.
- **entities**: A list of all detected entities in the sentence. Each element is the JSON file containing.

  - The type of entity
  - Start and end position of the entity in the sentence
  - The value of this entity
  - And the pipeline used to detect it

- **intent_ranking**: Similar to the **entities**, this is a list that ranks all the intent that might be classified as the intent of this sentence ordering in the decrease of each confidence value.

4.3 Collecting data from sensor

Sending and receiving data the sensors is a serial connection. The actual process will be as follows:

1. **Specify the COM port**: We write the *getPort()* to do this. To get the port that is communicating by the serial, we will use the *serial* library. The command for opening the serial port:

```
ser = serial.Serial(port=portName, baudrate=9600)
```

2. **Read data from the sensor**: The *serial_read_data()* function will take the serial signal and return the number value of the sensor.

```
def serial_read_data(ser):
    bytesToRead = ser.inWaiting()
    if bytesToRead > 0:
        out = ser.read(bytesToRead)
        data_array = [b for b in out]
        if len(data_array) >= 7:
            array_size = len(data_array)
            value = data_array[array_size - 4] * 256 + data_array[
   array_size - 3]
            return value
        else:
            return -1
    return 0
```

3. Then we call *readTemperature()* and *readHumidity()* in the while loop. These two functions are called every 10 seconds. Since the sensor is transmitted by RS485 cable, we will declare the value as a ModBus 485

format array and send this value to the Adafruit. Below is Modbus 485 format and *readTemperature()*:

```
air_temperature = [3, 3, 0, 0, 0, 1, 133, 232]
def readTemperature(ser):
    serial_read_data(ser)
    ser.write(air_temperature)
    time.sleep(1)
    return serial_read_data(ser)
```

4.4 Control the device

Similarly, relay 1 and relay 2 (corresponding to the door, and the light respectively) are also communicated by RS485 protocol, we declare the value for it. The *setDevice1()* and *setDevice2()* will receive the status of the relay to control the device turns on or off.

```
relay1_ON  = [0, 6, 0, 0, 0, 255, 200, 91]
relay1_OFF = [0, 6, 0, 0, 0, 0, 136, 27]

def setDevice1(state, ser):
    if state == True:
        ser.write(relay1_ON)
    else:
        ser.write(relay1_OFF)
```

4.5 Implementing IoT Gateway

To implement the IoT gateway, we implemented it in *server.py* file. Specifically, the steps are as follows:

1. **Import libraries and initialize**: In this step, the most important thing is to check if the library *Adafruit_IO* installation is really complete or not. Besides, you also need to fill in the *Username* and *Key* of the account.

2. **Implement some necessary functions**: The connection between IoT Gateway and Server Adafruit is based on a special protocol, called MQTT (Message Queuing Telemetry Transport). This is a transmission protocol communication based on publish/subscribe mechanism,

dedicated to the Internet of Things (IoT). The following three implementation functions serve to operate the protocol MQTT at Gateway IoT. The implementation of these functions is presented as follows:

```
def connected(client):
    for feed in AIO_FEED_DEVICE:
        client.subscribe(feed)

def disconnected(client):
    sys.exit(1)

def message(client, feed_id, payload):
     if feed_id == "smart-home.door":
        global isDoorSignal, isDoor
        isDoorSignal = True
        if payload == "1":
            isDoor = True
        else:
            isDoor = False
```

3. **Configuration for Gateway**: The final step in the gateway implementation is to create an MQTT client object so that it can be linked to the functions created above.

4.6 Implementing Database

4.6.1 Creating Table

We use Sequelize to easily access the MySQL database. A model is an abstraction that represents a table in our database. Below is the implementation of creating **User** table.

```
const User = sequelize.define('users', {
    id: {
        type: DataTypes.INTEGER,
        autoIncrement: true,
        primaryKey: true
    },
    name: {
        type: DataTypes.STRING
```

```
    },
    email: {
        type: DataTypes.STRING
    },
    password: {
        type: DataTypes.STRING
    },
    phone: {
        type: DataTypes.STRING
    },
    role: {
        type: DataTypes.STRING,
        defaultValue: 'Member'
    }
})
```

4.6.2 Initializing Data

We initialize some necessary data such as host account, permissions, and device parameters.

```
INSERT INTO users (id, name, email, phone, role)
    VALUES (1, 'Hanh', 'huuhanhce2001@gmail.com', '0123456789', 'Host');
INSERT INTO users (id, name, email, phone, role)
    VALUES (2, 'Nam', 'kingnamland@gmail.com', '0111111111', 'Member');

INSERT INTO permissions (id, permission) VALUES (1, 'Living room');
INSERT INTO permissions (id, permission) VALUES (2, 'Bedroom');
INSERT INTO permissions (id, permission) VALUES (3, 'Garage');

INSERT INTO user_permission (userId, permissionId) VALUES (1, 1);
INSERT INTO user_permission (userId, permissionId) VALUES (1, 2);
INSERT INTO user_permission (userId, permissionId) VALUES (1, 3);

INSERT INTO devices
    (id, feed, status, last_time_capture, time_active, power_consumption)
    VALUES (1, 'smart-home.light-livingroom', '0', '0', 0, 0);
INSERT INTO devices
    (id, feed, status, last_time_capture, time_active, power_consumption)
    VALUES (2, 'smart-home.light-bedroom', '0', '0', 0, 0);
```

4.7 Implementing Mobile App

Nowadays, the use of smart home technology has become increasingly popular. With the help of smart home apps, people can control various aspects of their homes, such as lighting, temperature, and security, from their smartphones.

4.7.1 Authentication Screen

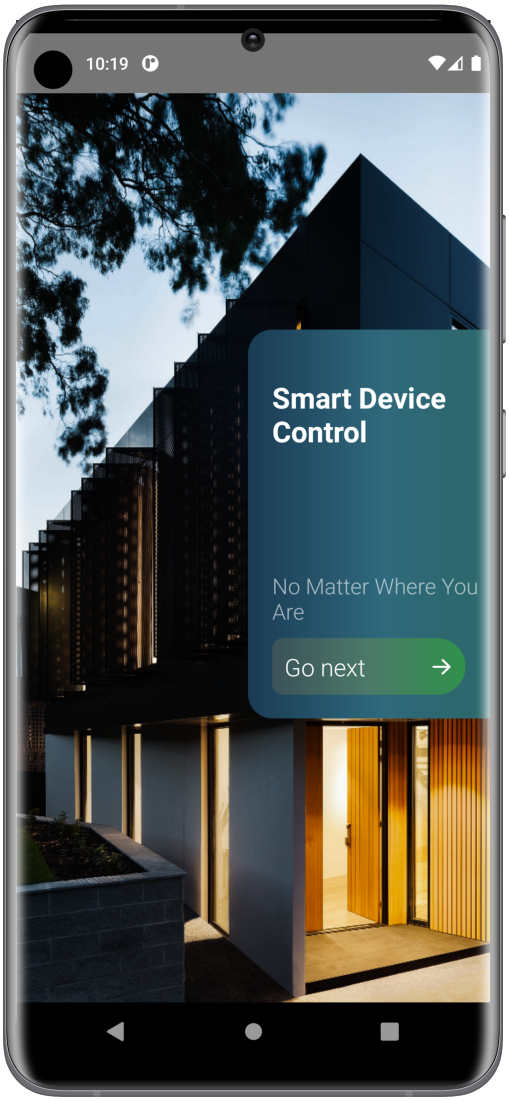

Figure 4.6: Landing screen of mobile app

Users must log in with their credentials, but new users can sign up for a new account with a valid token received from the host account. Forgot password option is available in case users forget their password.

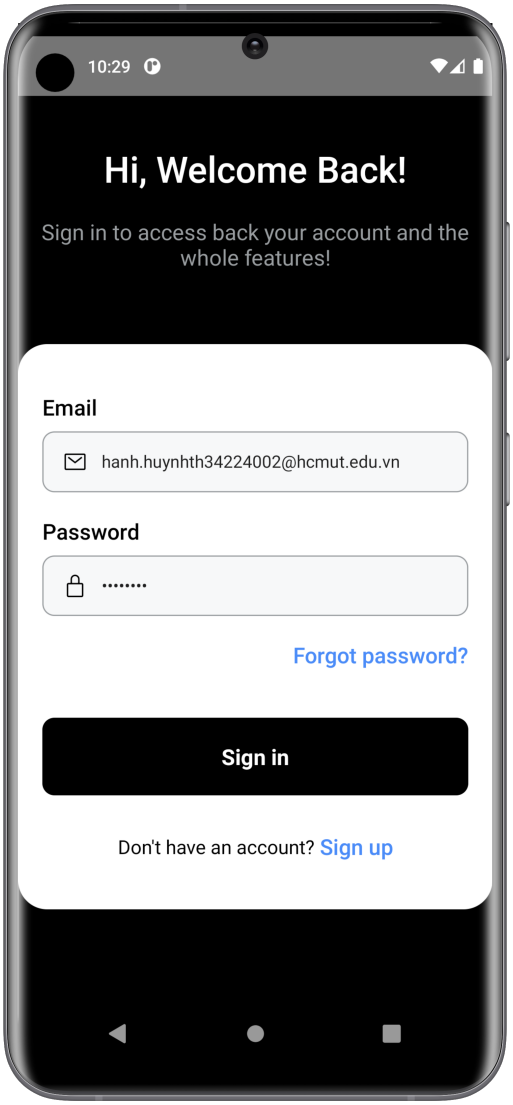

Figure 4.7: Login Screen

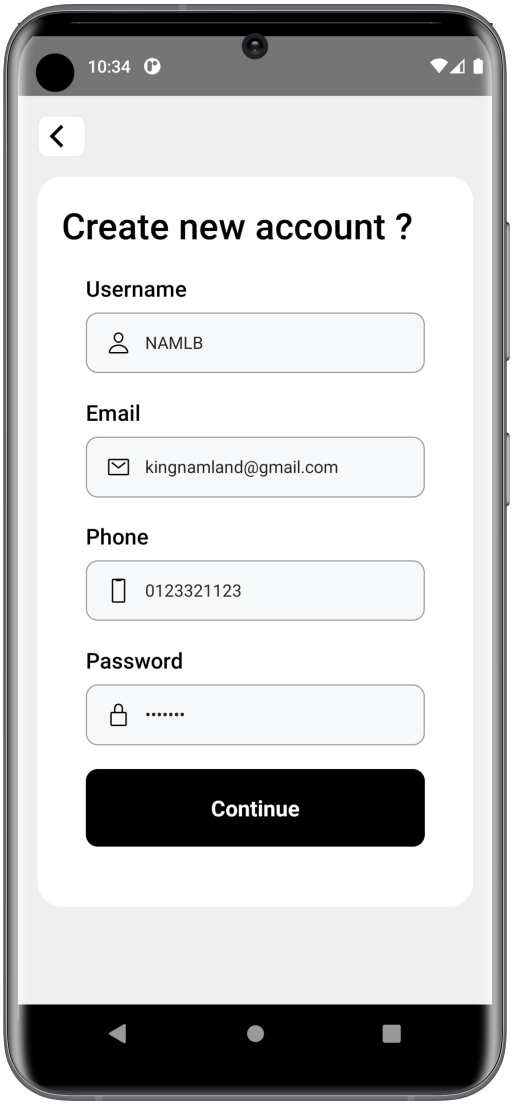

Figure 4.8: Register new member account screen

When creating a new account, the user receives a token from the host via email that needs to be verified on the "Verify Token From Email Screen". Similarly, if the user requests to reset their password, a token is generated by the system and sent to their email, which also needs to be verified on the same screen. This verification step ensures the user's identity and helps to prevent unauthorized access to their account. Only after successful verification, the user can proceed to create a new account or reset their password.

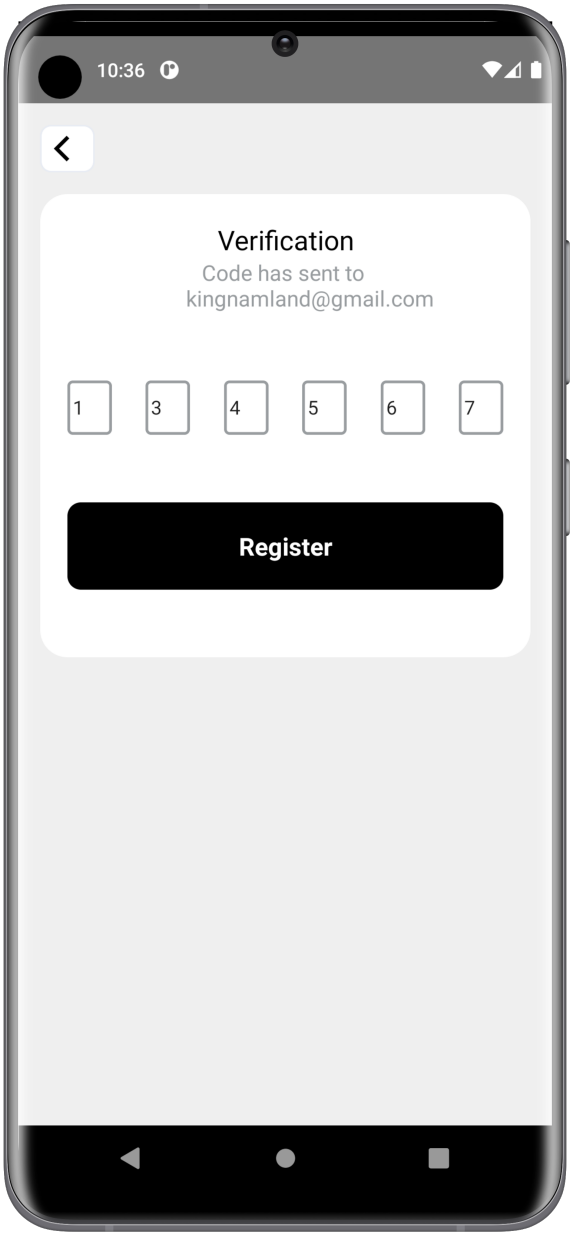

Figure 4.9: Verify token from email screen

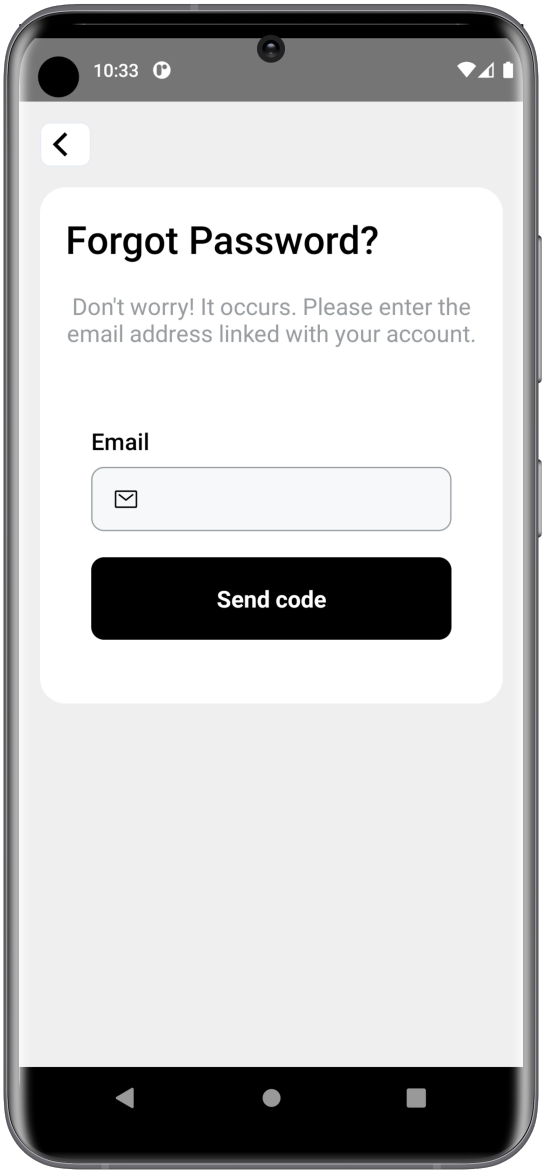

Figure 4.10: Reset password screen

4.7.2 Control Device Screen

The dashboard displays the current temperature and humidity levels in your home. This information is critical for ensuring that the temperature and humidity levels remain within the desired range. In addition, the dashboard features a toggle switch button that allows users to control various devices between each room remotely. This feature enhances convenience and accessibility. The switch button is easy to use, and users can toggle devices on or off with a simple tap. Overall, the dashboard provides users with critical environmental data and an easy-to-use control interface.

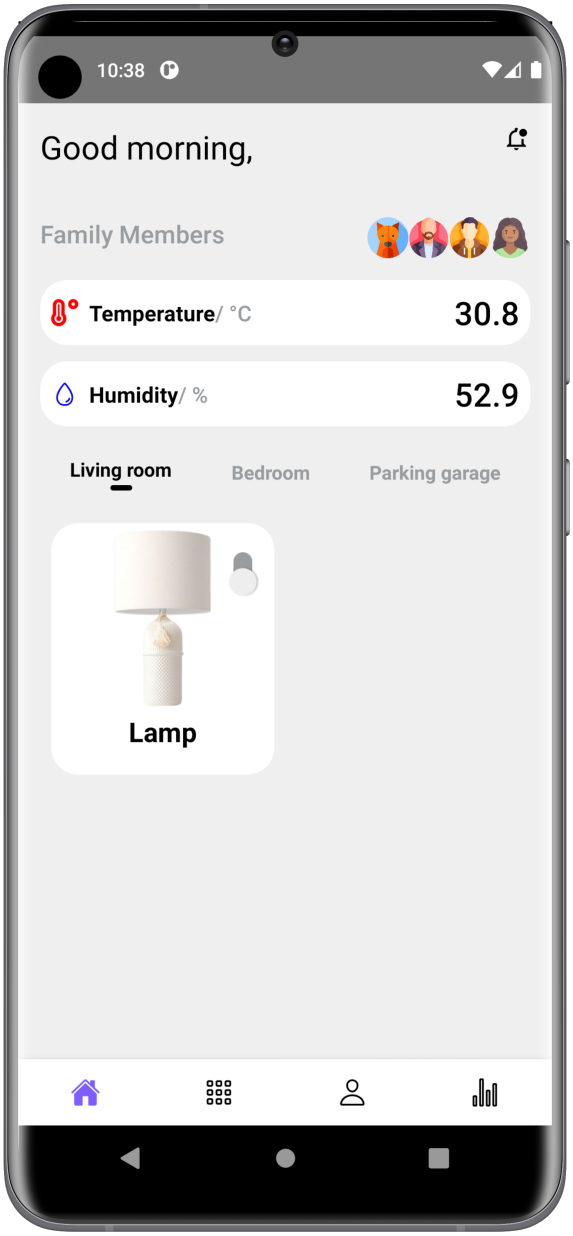

Figure 4.11: Home screen

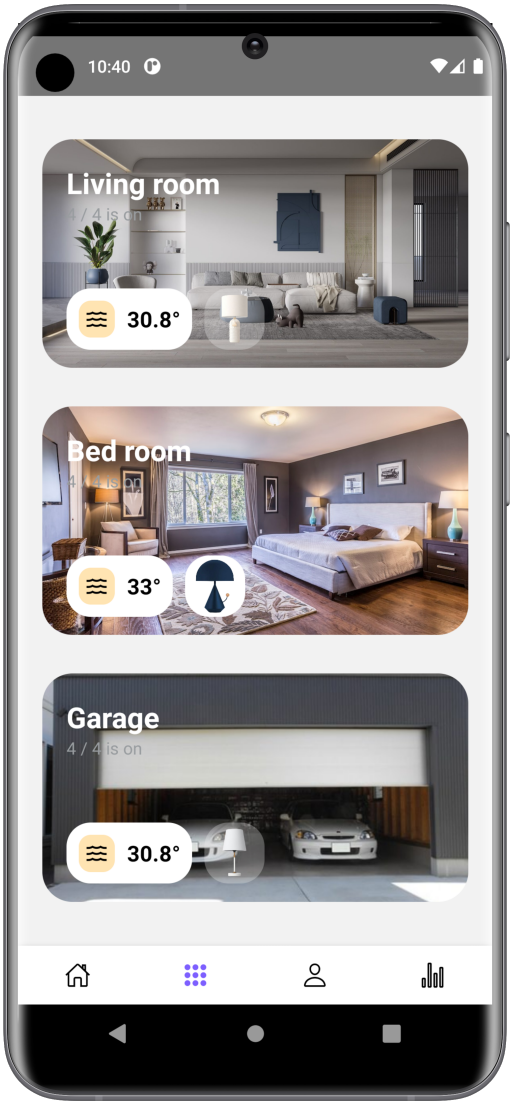

Figure 4.12: Dashboard screen

4.7.3 Set Schedule for Devices Screen

In a smart home, the screen of various devices such as smart TV, smartphones, tablets, and laptops can be scheduled. This can be done through the smart home app or using voice commands. For instance, the smart TV can be scheduled to turn on at a particular time and showcase a specific channel. Similarly, smartphones and tablets can be scheduled to go into sleep mode after a particular duration of inactivity. This not only saves energy but also ensures that the devices are well-maintained and last longer. The schedule can be adjusted according to the user's preference.

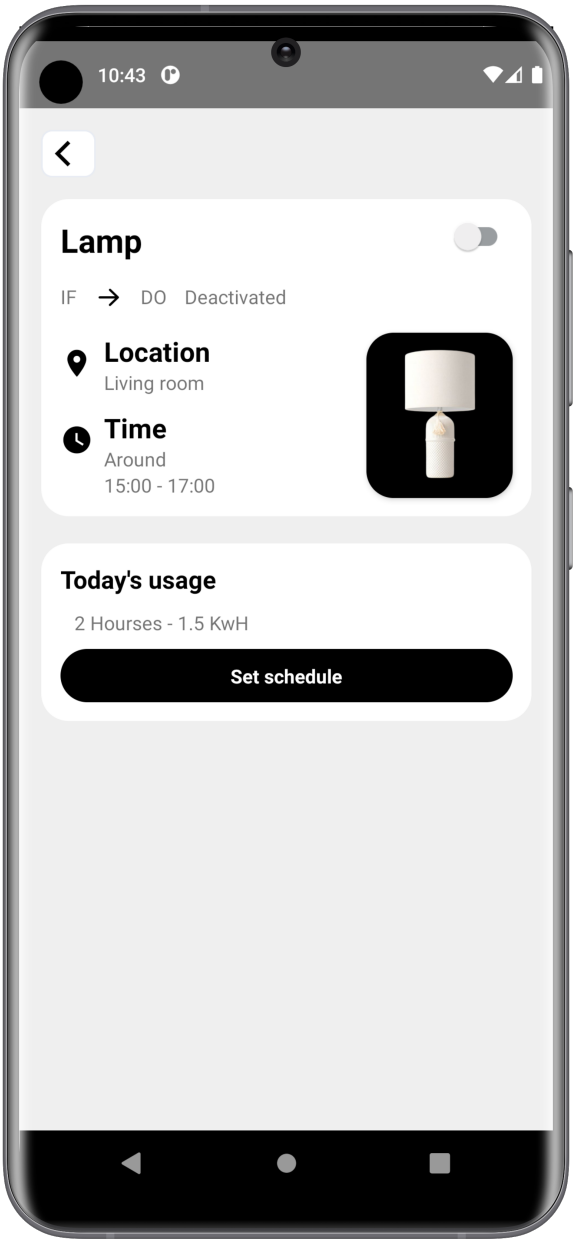

Figure 4.13: The detail of all condition device screen

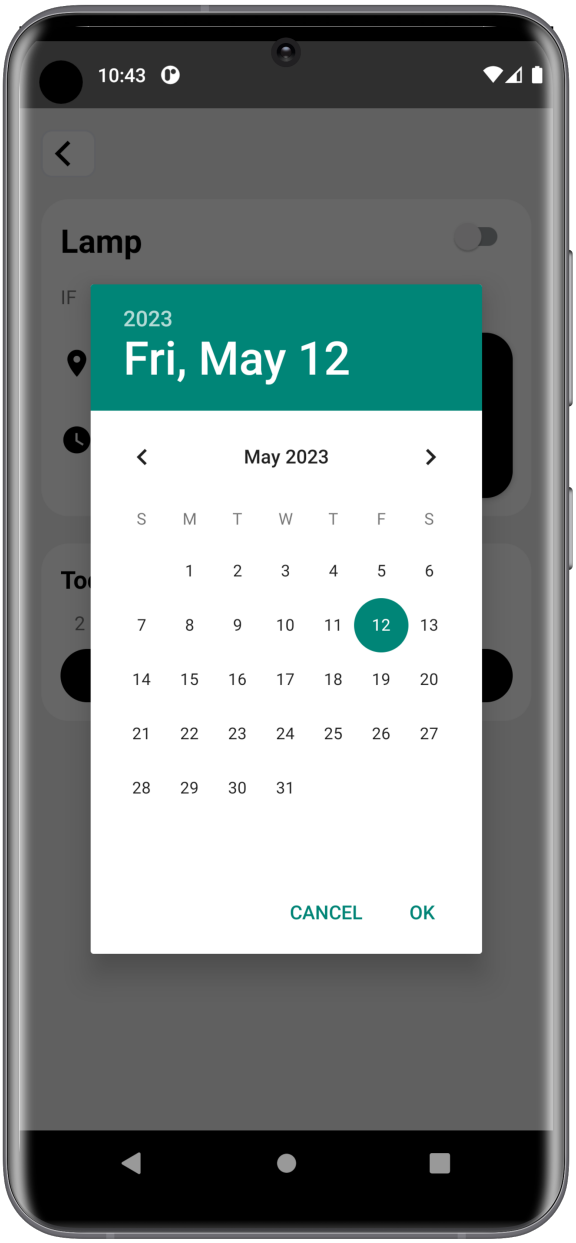

Figure 4.14: Set date in schedule screen

Figure 4.15: Set time in schedule screen

4.7.4 Invite new member and Set Permission Screen

The **Invite New Member** and **Set Permission** screen allows you to invite new members to your home and set their respective permissions. You can choose to invite members via email. In addition, you can specify what level of access and permissions each member will have, such as Living room, Bedroom, and Parking car. This screen provides a seamless and efficient way to manage the access of your team members and ensure that they have the appropriate level of access to your house.

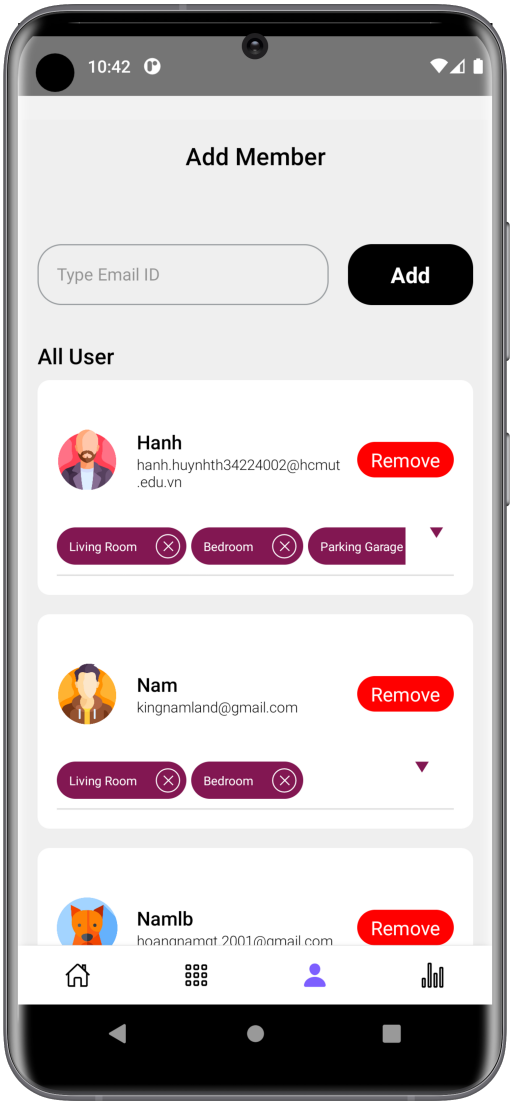

Figure 4.16: Invite and set permission for members screen

4.7.5 Statistics Screen

The statistics screen displays the power consumption for the selected device over a specific period. The user can view daily, weekly, or monthly usage and compare it to previous intervals. Additionally, a chart shows the temperature and humidity levels in a room over time, providing insight into the house's condition. The chart allows users to monitor whether changes in temperature or humidity could be affecting their condition. With these features, users can make informed decisions about controlling energy consumption and maintaining a comfortable indoor climate.

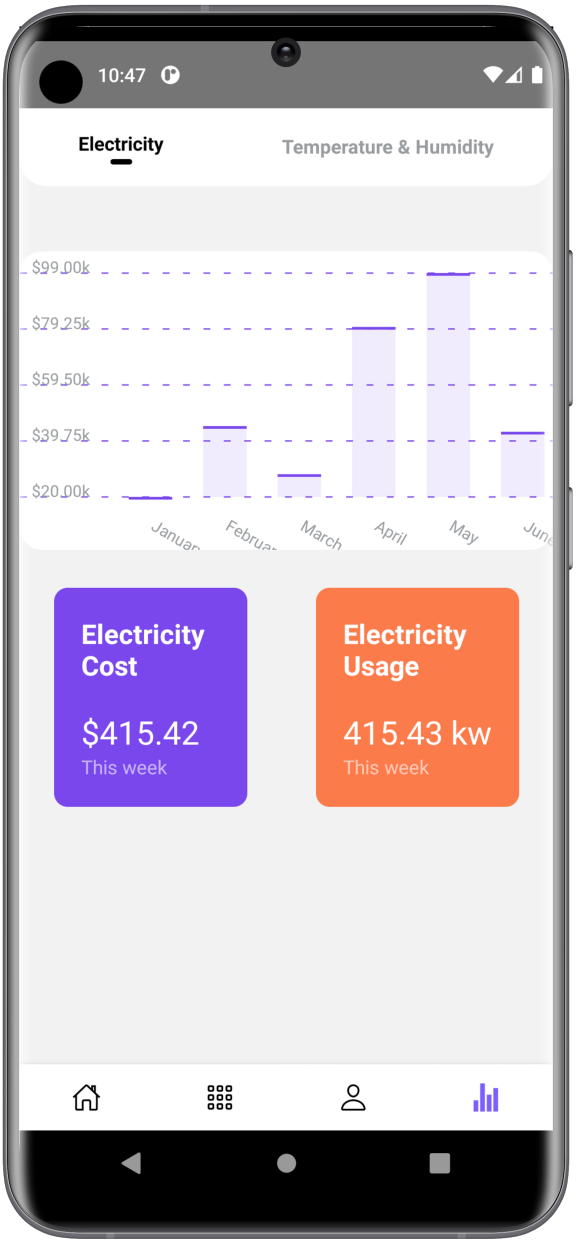

Figure 4.17: Power consumption screen

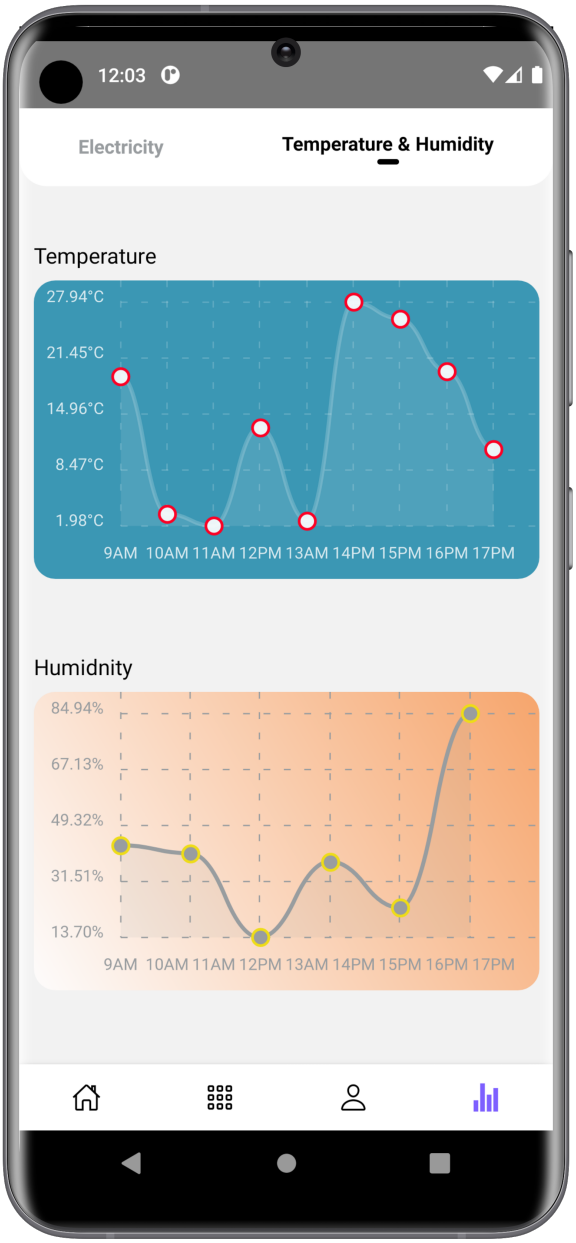

Figure 4.18: Temperature and humidity metrics screen

CHAPTER 5

CONCLUSION

In this chapter, we evaluate the system and summarize the project's progress by specifying complete tasks and the plan to develop our system in the future.

5.1 Setup Measurement

5.1.1 Face Recognition

To measure the accuracy of our face recognition model, we use the LWF dataset with reliable data labeled. The data set contains more than 11,678 images of faces collected from the web. Each face has been labeled with the name of the person pictured. 1680 of the people pictured have two or more distinct photos in the dataset.

To represent the evaluation function, we used statistical terms, including mean, standard deviation (Std.), minimum (Min), and maximum (Max) values on Euclidean-Distance (ED) results of the 128-Dimensional vector after encoding by our model. However, before going to the experiment step, we have to select some suitable parameters for the Classify Model to achieve the highest result.

There are two kinds of parameters that we must estimate the appropriate **DISTANCE_THRESHOLD** and the parameters of the Classify model.

- If the **DISTANCE_THRESHOLD** is too high, we will have a higher false positive error on our model. It means that if the invalid person has a distance of 0.5 and **DISTANCE_THRESHOLD** is 0.55. The system will wrongly predict that person

- If the **DISTANCE_THRESHOLD** is too low, we have another problem the false negative error is high, and the system will miss many valid persons. If the distance of the valid person ranges from [0.4 - 0.5] and the **DISTANCE_THRESHOLD** is 0.45. We have to identify approximately two times to recognize this person.
- Therefore, we have done the experiment that finds the range distance of true prediction and false prediction. From that, we can choose the most suitable **DISTANCE_THRESHOLD** for the Classify model. Below is the result of the experiment.

	Distance True Predict	**Distance False Predict**
Mean	0.347	0.522
Min	0.082	0.326
Std	0.063	0.048

Table 5.1: The result of the distance threshold

The results of the KNN model when we test in the known users and unknown users dataset. It includes accuracy, precision, recall score, and some information about the dataset. However, we only consider the accuracy score for the unknown user dataset to reduce the probability of predicting the user when they do not exist in the database.

Image Shape	(150, 150, 3)
Number of known user images	3870
Precision score known user	0.990
Recall score known user	0.856
Total processing time known user images (s)	1.732
Max processing time known user images (s)	0.0060
Min processing time known user images (s)	0.0015
Mean processing time known user images (s)	0.0018
Std processing time known user images (s)	0.0003

Table 5.2: Statistic result of KNN model with known user

Image Shape	(150, 150, 3)
Number of unknown user images	7808
Accuracy score unknown user	0.050
Total processing time unknown user images (s)	17.817
Max processing time unknown user images (s)	0.0060
Min processing time unknown user images (s)	0.0016
Mean processing time unknown user images (s)	0.0022
Std processing time unknown user images (s)	0.0004

Table 5.3: Statistic result of KNN model with unknown user

For the tests with the known users, the system has 99 percent corrects over all their predictions. Instead, we have only 85.6 percent Recall score known user. It means that the system has 14.4 percent to reject the correct users.

With the Accuracy score unknown user of 0.05 in testing with the unknown user, we can understand that the system has only 5 percent to predict a user when they do not exist in the system.

5.1.2 Speech Recognition

We use the Word Error Rate (WER) metric to evaluate the Speech Recognition engine. This is the most common metric used to evaluate ASR. Word Error Rate (WER) tells you how many words were logged incorrectly by the system during the conversation [19].

The formula for calculating WER is: **WER = (S+D+I)/N**

- **S (substitutions)**: When the system captures a word, but it's the wrong word.
- **D (deletions)**: These are words the system does not include.
- **I (insertions)**: When the system includes words that were not spoken.
- **N (total number of words spoken)**: How many words are contained in the entire conversation.

No	User message	Result	WER
1	Tắt đèn phòng khách	"tắt đèn phòng khách"	0%
2	Mở đèn phòng khách	"mở đèn phòng khách"	0%
3	Tắt đèn phòng ngủ	"tắt đèn phòng ngủ"	0%
4	Mở đèn phòng ngủ	"mở đèn phòng ngủ"	0%
5	Tắt tất cả các đèn	"tắt tất cả các đèn"	0%
6	Mở tất cả các đèn	"mở tất cả các đèn"	0%
7	Đóng cửa nhà xe	"đóng cửa nhà xe"	0%
8	Mở cửa nhà xe	"mở cửa nhà xe"	0%
9	Bây giờ là mấy giờ?	"bây giờ là mấy giờ"	0%
10	Thời tiết hôm nay thế nào?	"thời tiết hôm nay thế nào"	0%

Table 5.4: The WER metric of speech recognition engine

In this evaluation, we have tested the speech recognition engine on 10 different test cases. These test cases are sentences that users often interact with the system. They are divided into 3 categories: control the devices, ask about the time, and ask about the weather.

The column **"Result"** indicates that the speech recognition engine is capable of precisely recognizing and transcribing user requests. Based on the WER metric, we discover that the overall accuracy as a percent for the speech recognition engine is very high. That indicates that the system is working properly [19].

5.1.3 Speaker Verification

It is important to describe the parameters which are known as *de facto* standards for evaluating the performance of a speaker verification system. Two parameters widely used for the evaluation of biometric recognition systems are recognition accuracy and recognition speed [20]. Accuracy can be quantitatively described by the following measures:

- False acceptance rate (FAR)
- False rejection rate (FRR)
- Equal error rate (EER)

The false acceptance rate (FAR) measures the likelihood that the biometric system, based on the calculated score and chosen threshold, inaccurately

decides that the input feature set and the feature set selected from the database belong to the same person.

The false rejection rate (FRR) measures the likelihood that the biometric system, based on the calculated score and chosen threshold, inaccurately decides that the input feature set and the feature set selected from the database do not belong to the same person.

The equal error rate (EER) refers to a specific operating point of the biometric system where the FAR equals the FRR. It should be noted that the biometric system with the lowest EER value has the best decision-making performance.

Table 5.5 is the result of the experimental test on our speaker verification model. It indicates that our system is working well with very low EER.

Speech utterances	60
FAR	0.02%
FRR	0.03%
EER	0.025%

Table 5.5: The experimental test results on the GMM model

5.1.4 Intent classification

We validate and test dialogues end-to-end by running through test stories of Rasa. In addition, we also test the dialogue management and the message processing (NLU) separately.

Validating Data and Stories

Data validation verifies that no mistakes or major inconsistencies appear in our domain, NLU data, or story data. To validate, we run this command:

```
rasa data validate
```

As you can see, there is no error in data validation results, training a model can have a good performance, so it's always good to run the virtual assistant.

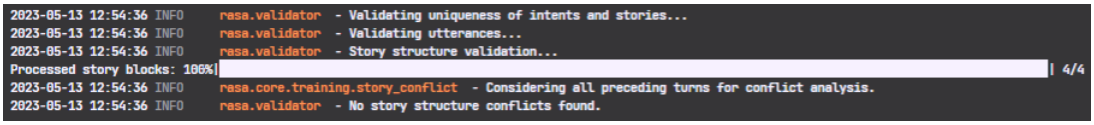

Figure 5.1: Validate data and stories

Evaluating an NLU Model

We also test the natural language understanding (NLU) model separately. Once our assistant is deployed in the real world, it will be processing messages that it hasn't seen in the training data. To simulate this, we should always set aside some part of your data for testing. We use **cross-validation**, which automatically creates multiple train/test splits.

Cross-validation automatically creates multiple train/test splits and averages the results of evaluations on each train/test split. This means all our data is evaluated during cross-validation, making cross-validation the most thorough way to automatically test our NLU model.

To run NLU testing in cross-validation mode run:

```
rasa test nlu --nlu data/nlu --cross-validation
```

After that, we get a report, confusion matrix, and confidence histogram for our intent classification model. The report logs precision, recall, and f1-score for each intent, as well as providing an overall average.

No	Intent	Precision	Recall	F1-score
1	control_device	1.0	1.0	1.0
2	goodbye	1.0	1.0	1.0
3	mood_great	1.0	1.0	1.0
4	inquire_time	1.0	1.0	1.0
5	mood_unhappy	1.0	1.0	1.0
6	greet	1.0	1.0	1.0
7	deny	1.0	1.0	1.0
8	affirm	1.0	1.0	1.0
9	inquire_weather	1.0	1.0	1.0

Table 5.6: The report logs of the intent classification model

The confusion matrix shows which intents are mistaken for others. Below is the confusion matrix:

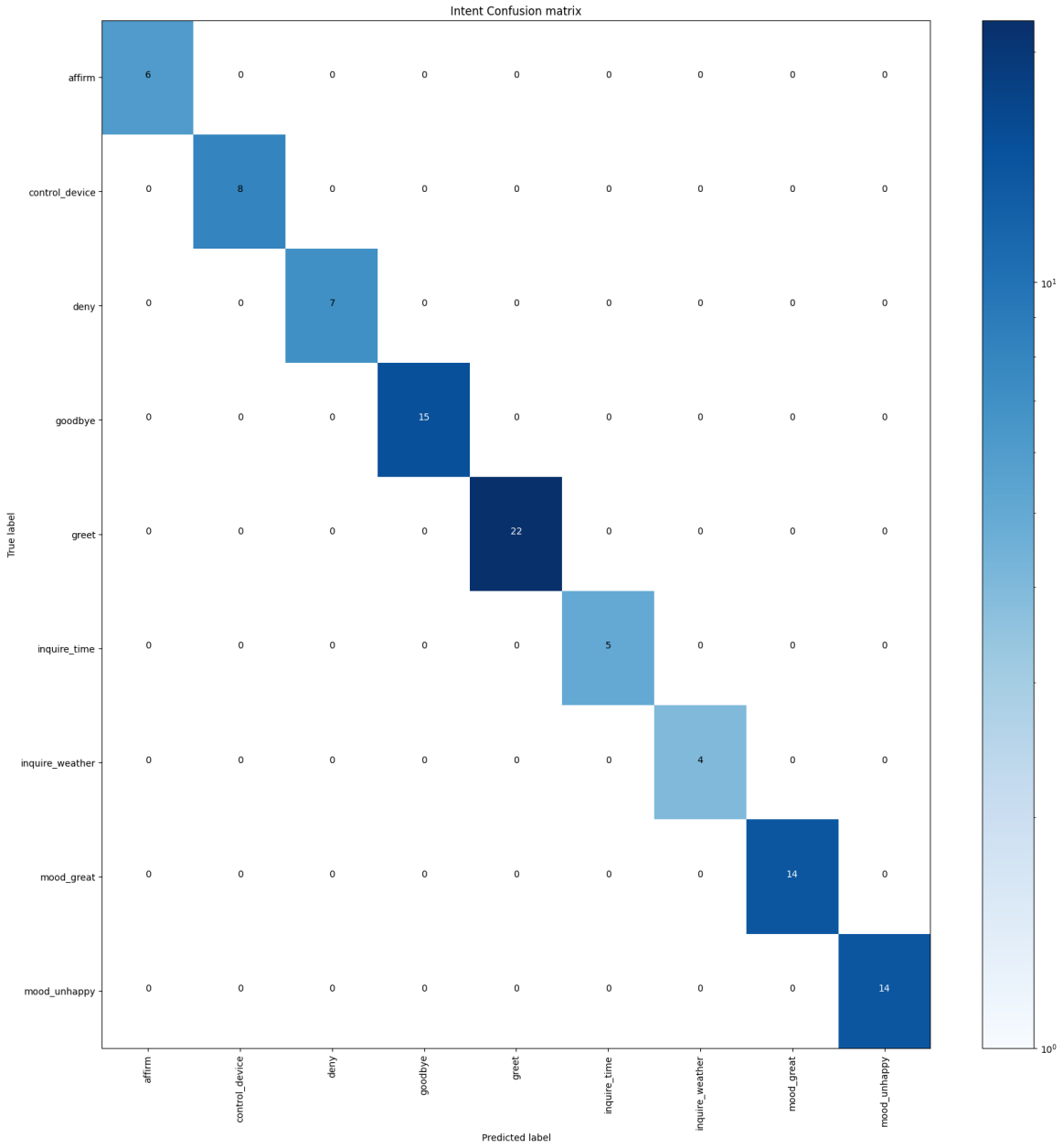

Figure 5.2: The confusion matrix of the intent classification model

The histogram visualizes the confidence for all predictions, with the correct and incorrect predictions being displayed by blue and red bars respectively. Improving the quality of your training data will move the blue histogram bars up the plot and the red histogram bars down the plot. It should also help in reducing the number of red histogram bars.

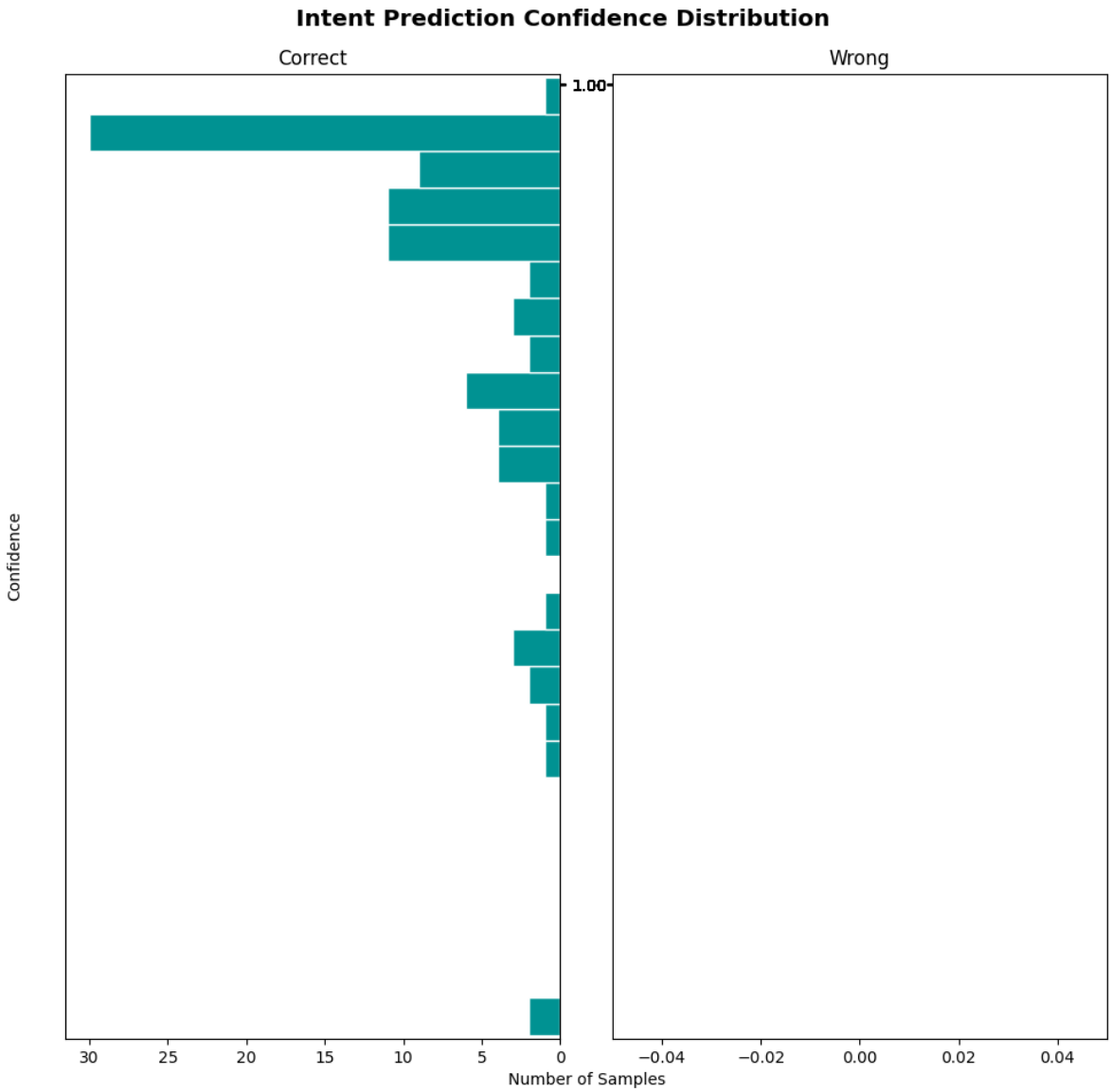

Figure 5.3: The intent histogram of the intent classification model

All the above metrics show that its performance is extremely good, in many cases the possibility of correct classification is extremely high. This helps the system to make accurate decisions and improve the user experience.

5.1.5 Mobile Application

We conduct the evaluation of the implemented functions mobile app through the method of black box testing.

No	Requirement	Result
1	The statistics and data on the application are fully displayed, exactly as the data collected from devices	Pass
2	The animations on the app interface display smoothly	Pass
3	Flexible, beautiful, resolution display panel interface high on many screen sizes, different phones	Pass
4	The application interface displays a full range of buttons, search areas, information entry area up to the present time	Pass
5	Create an account on the application successfully with a valid password, username has not been used before	Pass
6	Perform user permissions when logging into the system, hiding information for users who don't have permission to view	Pass
7	Schedule timing for control the devices to work on schedule	Pass
8	Host is possible to add members to the system through the email and add permission for each account	Pass
9	Members can track electricity consumption as well as the price to pay	Pass
10	The metrics of temperature as well as humidity are displayed real-time through chart	Pass

Table 5.7: Test sheet of the entire application system

5.2 Future Plan

In the future, we look forward to improving some issues and developing the system following:

Hardware device upgrades: In order to optimize real-time facial recognition and model training processes, our team will be enhancing the device with GPU support. This upgrade will enable quicker processing of data and more efficient performance compared to traditional CPU systems.

Enhanced security: Because most facial recognition system is easy to be attacked by spoofing methods. Therefore, in order to design a secure face recognition system in a real scenario, anti-spoofing techniques should be a priority from the initial planning of the system in the future.

Virtual Reality (VR) technology: VR technology can revolutionize the concept of smart homes. By creating a VR environment, homeowners can experience and control their homes in a more immersive way.

Add device monitor electricity: It measures the electricity usage of any device plugged into it. Besides, it can even connect to a smartphone app or web portal to track usage data over time

REFERENCES

[1] Meredith Hirt, Samantha Allen. *"10 Smart Home Trends This Year"*. Available at: https://www.forbes.com/smart-home-tech-trends/

[2] Smart Home Starter - Mar 30, 2022. *"Advantages and Disadvantages of a Smart Home"*. Available at: https://smarthomestarter.com/advantages-and-disadvantages-of-a-smart-home/

[3] Architecture & Design - Dec 17, 2021. *"The attractions of smart homes"*. Available at: https://www.architectureanddesign.com.au/the-attractions-of-smart-homes/

[4] Adam Geitgey - Sep 29, 2022. *"Modern Face Recognition with Deep Learning"*. Available at: https://medium.com/modern-face-recognition-with-deep-learning/

[5] Face Recognition. *"Face Recognition Docs"*. Available at: https://face-recognition.readthedocs.io/readme.html/

[6] Kelsey Foster - Sep 29, 2022. *"The Top Free Speech-to-Text APIs, AI Models, and Open Source Engines"*. Available at: https://www.assemblyai.com/blog/the-top-free-speech-to-text-apis-and-open-source-engines/

[7] Rasa Docs - Dec 12, 2022. *"Introduction to Rasa Open Source & Rasa Pro"*. Available at: https://rasa.com/docs/rasa/

[8] Abraham Zewoudie - Jul 12, 2022. *"Speaker Recognition and Verification"*. Available at: https://wiki.aalto.fi/Speaker-Recognition-and-Verification/

[9] Intel Corporation. *"What is a NUC?"*. Available at: https://www.intel.com/content/www/us/en/products/docs/boards-kits/nuc/what-is-nuc-article.html

[10] OhStem. *"Máy tính lập trình mini Yolo:Bit"*. Available at: https://ohstem.vn/product/may-tinh-lap-trinh-yolobit/

[11] Google for Education. *"Python Introduction"*. Available at: https://developers.google.com/edu/python/introduction

[12] CFI Team. *"Python in Machine Learning"*. Available at: https://corporatefinanceinstitute.com/resources/data-science/python-in-machine-learning/

[13] React Native. *"Core Components and Native Components"*. Available at: https://reactnative.dev/docs/intro-react-native-components

[14] NodeJS Team. *"Introduction to Node.js"*. Available at: https://nodejs.dev/en/learn/

[15] MDN Web Docs. *"Express/Node introduction"*. Available at: https://developer.mozilla.org/node-and-express

[16] MySQL 8.0 Reference Manual. *"What is MySQL?"*. Available at: https://dev.mysql.com/what-is-mysql.html

[17] Brent Rubell. *"Welcome to Adafruit IO"*. Available at: https://learn.adafruit.com/welcome-to-adafruit-io/what-is-adafruit-io

[18] Adrian Rosebrock. *"Face recognition with OpenCV, Python, and Deep Learning"*. Available at: https://pyimagesearch.com/face-recognition/

[19] Sekhar Vallath. *"Key Metrics for Evaluating Speech Recognition Software"*. Available at: https://symbl.ai/blog/key-metrics-and-data-for-evaluating-speech-recognition-software/

[20] Krčadinac, O., Šošević, U., & Starčević, D. (2021). Evaluating the Performance of Speaker Recognition Solutions in E-Commerce Applications. Sensors (Basel, Switzerland), 21(18), 6231. Available at: https://doi.org/10.3390/s21186231

STUDENT INFORMATION

List of project authors:

1. **Huynh Huu Hanh** - ID: 1910161
    - Phone number: (+84)944.071.272
    - Email: hanh.huynhth34224002@hcmut.edu.vn
2. **Truong Hoang Nam** - ID: 1910358
    - Phone number: (+84)327.453.594
    - Email: nam.truongcomputer@hcmut.edu.vn