Faculty of Computer Science and Information Technology

AIOT SMART DOOR LOCK FOR HOME SECURITY

Siti Maisarah binti Mohamed Mazlan

Bachelor of Computer Science with Honours (Network Computing)

2024

AIOT SMART DOOR LOCK FOR HOME SECURITY

SITI MAISARAH BINTI MOHAMED MAZLAN

This project is submitted in partial fulfilment of the

Requirements for the degree of

Bachelor of Computer Science with Honours (Network Computing)

Faculty of Computer Science and Information Technology

UNIVERSITI MALAYSIA SARAWAK

2025

KUNCI PINTU PINTAR BERASASKAN AIOT UNTUK KESELAMATAN RUMAH

SITI MAISARAH BINTI MOHAMED MAZLAN

Projek ini merupakan salah satu keperluan untuk
Ijazah Sarjana Muda Sains Komputer dan Teknologi Maklumat
(Pengkomputeran Rangkaian)

Fakulti Sains Komputer dan Teknologi Maklumat
UNIVERSITI MALAYSIA SARAWAK

2025

UNIVERSITI MALAYSIA SARAWAK

THESIS STATUS ENDORSEMENT FORM

TITLE **AIOT SMART DOOR LOCK FOR HOME SECURITY**

ACADEMIC SESSION: 2024/2025

SITI MAISARAH BINTI MOHAMED MAZLAN
(CAPITAL LETTERS)

hereby agree that this Thesis* shall be kept at the Centre for Academic Information Services, Universiti Malaysia Sarawak, subject to the following terms and conditions:

1. The Thesis is solely owned by Universiti Malaysia Sarawak
2. The Centre for Academic Information Services is given full rights to produce copies for educational purposes only
3. The Centre for Academic Information Services is given full rights to do digitization in order to develop local content database
4. The Centre for Academic Information Services is given full rights to produce copies of this Thesis as part of its exchange item program between Higher Learning Institutions [or for the purpose of interlibrary loan between HLI]
5. ** Please tick (√)

☐	CONFIDENTIAL	(Contains classified information bounded by the OFFICIAL SECRETS ACT 1972)
☐	RESTRICTED	(Contains restricted information as dictated by the body or organization where the research was conducted)
√	UNRESTRICTED	

Validated by

(AUTHOR'S SIGNATURE)
No 30, Jalan 3/33, Bandar Tasik
Puteri, Rawang 48020
Selangor.
Date: 23/06/2025

(SUPERVISOR'S SIGNATURE)
Date: 23/07/2025

Note * Thesis refers to PhD, Master, and Bachelor Degree ** For Confidential or Restricted materials, please attach relevant documents from relevant organizations / authorities

DECLARATION OF ORIGINALITY

My name is Siti Maisarah Binti Mohamed Mazlan with the matric number of 78532, hereby solemnly declare that the report titled '***AIoT Smart Door Lock for Home Security***' is prepared and completed authentically by me in partial fulfilment of the requirements for the degree of Bachelor of Computer Science with Honours (Network Computing) is a record of an original work carried out by me under the guidance of Associate Professor Dr. Mohamad Nazim Bin Jambli. I further declare that the work reported in this project has not been previously or concurrently submitted for any other degree at Universiti Malaysia Sarawak (UNIMAS) or any other institutions. I declare that I have not copied or plagiarize from any other sources except for quotations and citations which have been duly acknowledged.

Siti Maisarah Binti Mohamed Mazlan (78532)

Faculty of Computer Science and Information Technology

Universiti Malaysia Sarawak

ACKNOWLEDGEMENT

This project could not be successful if not for these amazing people that have been helping me throughout this project. First and foremost, I would like to express my heartfelt gratitude to my family, especially my mom who always support me with her unwavering belief no matter what happen. For every one of her prayers, thank you so much. And a special thank goes to my very best friend, Fahmi, for being there for me through thick and thin and through every sweat and tears. Without your constant assurance, support, and belief, I would never make it this far. I truly appreciate each and every word of comfort you give. Aside from that, I would also like to thank my project supervisor, AP Dr Mohamad Nazim Bin Jambli, for his invaluable guidance and continuous support throughout the development of this project. His insight has played a significant role in guiding the quality and direction of my work. To my friends, Nouf, Izzan, and Olivia, thank you for being a constant source of inspiration and support for me and for always be there for each other especially during the last semester of our student year. Thank you for always helping me and wait for me, you guys are the best. This project would not be as enjoyable if it is not for you all. Lastly, to my examiner, Dr Noralifah Binti Annuar and everyone who contributed to this project whether directly or indirectly, thank you. This achievement would not have been possible if it is not for your supports.

ABSTRACT

In this project, an AIoT Smart Door Lock—a secure, keyless home entry system that combines artificial intelligence (AI) and the Internet of Things (IoT)—is designed, developed, and evaluated. By leveraging real-time facial recognition as its primary authentication approach, which is processed on-device using a Raspberry Pi 4, the system overcomes the security shortcomings of traditional smart locks. It includes an alternative entry method for temporary visitors using dynamically generated QR codes for more flexibility. Key features include a PIR sensor for motion detection, a Blynk mobile app for complete remote management, and real-time intrusion notifications with picture snapshots sent via a Telegram bot. Through thorough User Acceptance Testing (UAT) with a variety of user groups, the prototype's efficacy was assessed. Functional testing verified the successful operation of all essential functionalities, such as remote controls, QR code access, and facial recognition in a various condition. The system's user-friendly interface and notable improvement in home security were praised by users, who also pointed out that system response time needed to be improved in the future. This study presents a successful proof-of-concept for an intelligent home security system with many layers that provides a reliable and adaptable substitute for traditional smart locks.

ABSTRAK

Projek ini membentangkan reka bentuk, pembangunan, dan penialaian sebuah sistem keselamatan rumah iaitu AIoT Smart Door Lock. Sistem ini menangani kelemahan kunci rumah tradisional dan konvensional dengan menggunakan pengecaman wajah secara langsung sebagai kaedah pengesahan utama, yang diproses secara setempat menggunakan Raspberry Pi 4. Bagi meningkatkan fleksibiliti sistem, ia turut menggabungkan mekanisme akses sekunder untuk tetamu sementara melalui kod QR yang dijana secara dinamik dan unik. Ciri-ciri utama sistem adalah pengesanan gerakan menggunakan sensor PIR, pencegahan pencerobohan secara langsung berserta gambar yang dihantar melalui bot Telegram, serta pengurusan jarak jauh yang komprehensif melalui aplikasi pintar Blynk. Selain itu, sistem ini menggunakan maklum balas berdasarkan Ujian Penerimaan Pengguna (UAT) bagi memastikan kecekapan, ketepatan dan fungsi sistem beroperasi dengan baik serta berjaya di penuhi ke dalam penambahbaikan sistem. metodologi seni bina sistem dan proses penilaian telah diperincikan dalam kajian ini, sekali gus menunjukkan keseluruhan potensinya untuk mengubah landskap keselamatan rumah secara menyeluruh. Maklum balas pengguna menunjukkan peningkatan ketara terhadap keselamatan rumah dan mesra pengguna, disamping mengenal pasti masa tindak balas sistem sebagai aspek utama untuk penambahbaikan pada masa hadapan. Kajian ini berjaya membuktikan konsep sistem keselamatan rumah pintar berbilang lapisan yang bagus dan fleksibel sebagai alternatif kepada kunci pintar sedia ada.

Table Of Contents

Lists of Figures

Lists of Tables

CHAPTER 1: INTRODUCTION

1.1 Introduction

The Internet of Things (IoT) has been evolving and revolutionizing due to the growth and change in Industrial Revolution 4.0 (IR4.). IoT comprises multiple devices to get connected and to create new innovations that enables it to share real-time data and communicating (Rahim et al., 2023). The technology is equipped with sensors, software and other technologies to gather and exchange data in real time, facilitating automation, control, and monitoring for a range of applications. IoT has undergone tremendous transformation with the onset of Industrial Revolution 4.0 (IR4.0) that prioritizes intelligent automation, data sharing, and advanced technologies like cloud computing, machine learning (ML), and artificial intelligence (AI) (Xu et al., 2021). The application of IoT varies from smart home where it enables IoT integration for example like controlling the lights remotely, healthcare where hospitals and clinics use wearable devices to monitor health metrics, as well as smart cities, Industrial IoT and implementation in agriculture.

Smart home is one of the integrations in IoT where IoT is implemented in the house systems and its technology integrates with home automation that can control and monitor many things such as controlling lights from a far. The automation system offers many functions to the home technology and one of it are securing the home through smart door locks. However, current smart door locks are susceptible to data breaches and unauthorized access, that puts homeowners' security at risk. These vulnerabilities highlight the need for more robust and intelligent security solutions.

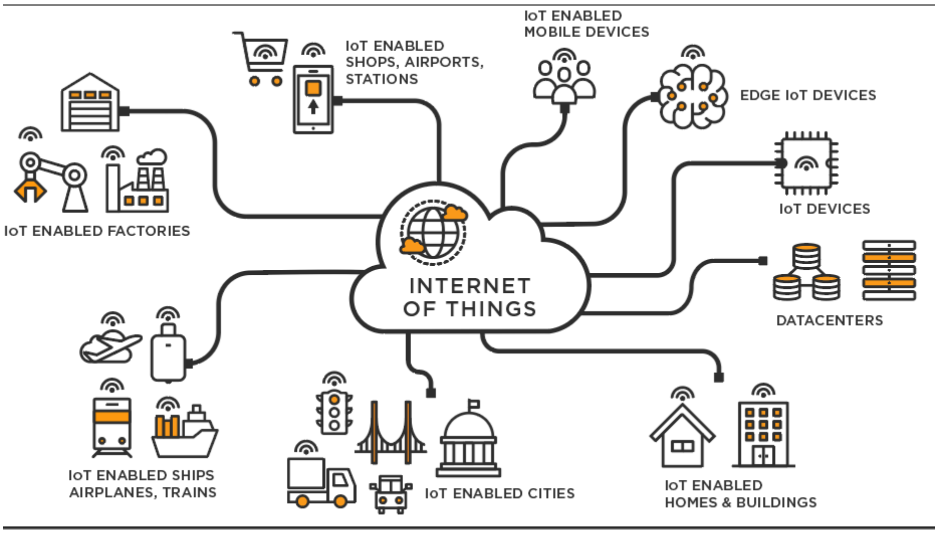


Figure 1.1: Integration of Internet of Things

With its cutting-edge features including real-time monitoring, pattern recognition, and decision-making, AI has become a powerful tool to address these issues. It is the emulation of human intellect in machines, allowing them to carry out tasks like learning, reasoning, and problem-solving that normally need human cognition (Russell & Norvig, 2021). The Artificial Intelligence of Things (AIoT) is a synergy that happens when AI is merged with IoT to improve the security and functioning of connected devices. AIoT uses AI algorithms to evaluate data gathered by IoT devices, allowing for automation, better user experiences, and predictive analytics.

Houses are often targeted by intruders, making security as a top priority for homeowners. According to Stolojescu-Crisan et al. (2021), people dedicated half of their life working to provide and gain something for themselves hence the house needs to be safeguarded and the first thing to do is to make sure the door has a strong security because it is the main way for us to go in and out of the house. Smart door lock represents the first step to enhance the security of the house by offering an automated and access control from it and over the last ten years, smart door lock system has use biometric system such as using fingerprint and facial recognition to access the house (Balraj Balla & Jadhao, 2023). The old conventional ways of lock and unlocking the door are by using a physical key. This method has a great number of vulnerabilities due to the potential to lost the key, forgot the key or even to get the key stolen. However, over the years the method has been advanced significantly and most of the house now can use a digital door lock where they are now using General Packet Radio Service (GPRS), sensors, and even camera (Zainudin et al., 2022) to enhance the security and unlock the door by using biometric features or digital pin.

Therefore, by AI and IoT to provide an advanced real-time monitoring and accurate identification of authorized users, this solution aims to revolutionize home security, addressing the traditional smart locks and enhancing user convenience and its security features. In order to improve homeowner security and convenience, this project will develop the "AIoT Smart Door Lock," which will provide hands-free, automatic access by fusing IoT technology with AI-based facial recognition that aligns with the growing demand for intelligent home security solutions. In addition to facial recognition, the system also integrates a QR code-based access feature for temporary or guest users, offering a secure and flexible method for controlled access as well as real-time alert, which the system includes Telegram notifications that alert homeowners immediately when unauthorized access is detected or motion is sensed near the door.

1.2 Problem Statement

Conventional, traditional and even current smart door lock like fingerprints and PINs are vulnerable to be exploit by intruder (Y Lakshmi Venkata et al., 2021). For instance, according to a 2019 study by Viderberg, 50% of homes with home automati on systems voiced worries about remote access security, pointing to problems like data b reaches or leaks in smart home devices. These weaknesses highlight the importance for a more reliable and safe solution. Therefore, this proposed project aims to address these vulnerabilities and provide an improved security against unauthorized access all while enhancing user experience through hands-free, seamless operation by utilizing an AI-powered facial-recognition smart system.

1.3 Project Scope

The goal of this project is to design and develop a prototype of a smart door lock system using facial recognition that integrated with AI and IoT to enhance home security and user convenience.

Facial Recognition features where the system combines both AI and IoT for precise, real-time monitoring and user identification. The OpenCV and MediaPipe libraries will be used in the development of this key features, guaranteeing precise and effective user identification. The facial recognition system will serve as the primary method for unlocking the door to replace the traditional ways to unlock the door.

In terms of technology, the prototype will include key hardware elements such as a buzzer, a Raspberry Pi 4, a PIR motion sensor, a servo motor, and an AI camera. These parts will allow for smooth operation of features like facial recognition, motion detection, and door locking and unlocking systems. This project will use python programming language to create the software and incorporate facial recognition algorithm as well as the IoT. Other than that, Blynk Software will serve as the platform for remote control and monitoring. The system will also have Wi-Fi connectivity to allow seamless device communication and give homeowners remote access. Additionally, the system is equipped with a Telegram bot notification feature to send alerts in real-time when unauthorized access attempt is detected. A QR code authentication method is also introduced to provide temporary access for guests or visitors, without requiring them to be permanently registered in the system.

Homeowners looking for improved security features like facial recognition, remote access, and real-time monitoring to adequately safeguard their properties are among the system's target consumers and compared to conventional locking methods, the hands-free and user-friendly design of the system makes it easier for older people and children to operate by doing away with the need for actual keys.

This prototype model will be evaluated through UAT for its efficiency to attempt to determine its accuracy, performance, and user satisfactions. This evaluation stage will make sure the system achieves its goals of improved simplicity and security.

1.4 Objectives

The project aims to analyse and design a smart door lock that can provide extra security and convenience to home environment and the home owner. The objectives of this project are as below:

1. To design and develop a smart door lock that integrates IoT and AI-powered facial recognition for real-time monitoring and secure access control.
2. To evaluate the prototype effectiveness based on User Acceptance Testing (UAT).

1.5 Methodology: Prototype

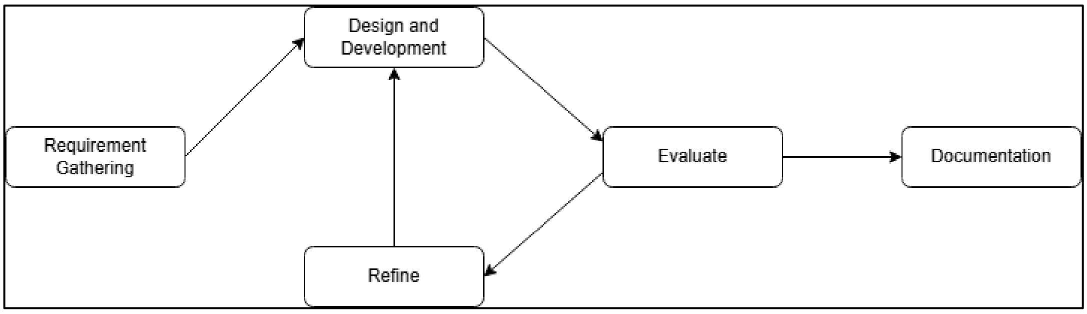


Figure 1.2: Prototype methodology diagram.

1.5.1 Requirement Gathering

The first step in this method is requirement gathering where in this phase, analysis is made to identify user needs, project goals, list key features and requirements including hardware and software used of the project are defined in detail. During this proses, research and survey are made as well as comparison on existing model to search for design and research gap. This phase helps in developing an ideal end-product.

1.5.2 Design and Development

The second phase is where designing and developing is done. A quick and simple design of the product is created. The prototype is created with basic functionality to give a brief idea of the product to the user. Then, after designing the product, an actual prototype is developed based on the information gathered in the first phase using hardware like camera module, lock mechanism and Raspberry Pi.

1.5.3 Evaluate

In this phase, User Acceptance Testing (UAT) will be made where user will test the prototype in real-world environment to verify its functionality is met the proposed outcomes. Feedback will be taken from this evaluation and taken to the next phase.

1.5.4 Refinement

Improve prototype depending on the feedbacks to make sure the prototype satisfies all requirements and objectives.

1.5.5 Documentation

Compiling all data, design improvements and findings into a comprehensive project report.

1.6 Significance of the project

The AIoT Smart Door Lock is a project that is the same as the existing smart door lock but this system uses facial recognition technology driven by AI and links it with the IoT for smooth, hands-free operation, in contrast to conventional smart locks that depend on PINs, passwords, or physical keys. This removes the need for actual keys and password memorization, making it simpler for kids, seniors, and homeowners to use. The AIoT Smart Door Lock is a better option for contemporary homeowners due to its many unique features over competing systems. First of all, the hands-free, facial recognition-based system is incredibly easy to use especially to users who might find conventional locking methods difficult. This makes it more accessible for a larger group of users simply by using facial recognition without requiring for physical keys and password memorization.

IoT-enabled devices such the Raspberry Pi and AI camera are used in the suggested solution to gather and send real-time data, which is then processed by AI algorithms to conduct facial recognition and user authentication. Because of this connectivity, users can operate and keep an eye on their door locks from any location with real-time monitoring and remote access using Wi-Fi. Additionally, by integrating secure communication protocols and cutting-edge encryption, the solution mitigates cybersecurity threats in existing smart locks. Moreover, the inclusion of Telegram alerts enhances real-time user awareness by sending instant notifications in cases of potential intrusion. The system also supports a guest access mechanism using scannable QR codes, allowing visitors or temporary users to unlock the door securely without being added to the authorized facial recognition database. These additions improve flexibility while maintaining security.

1.7 Project Schedule

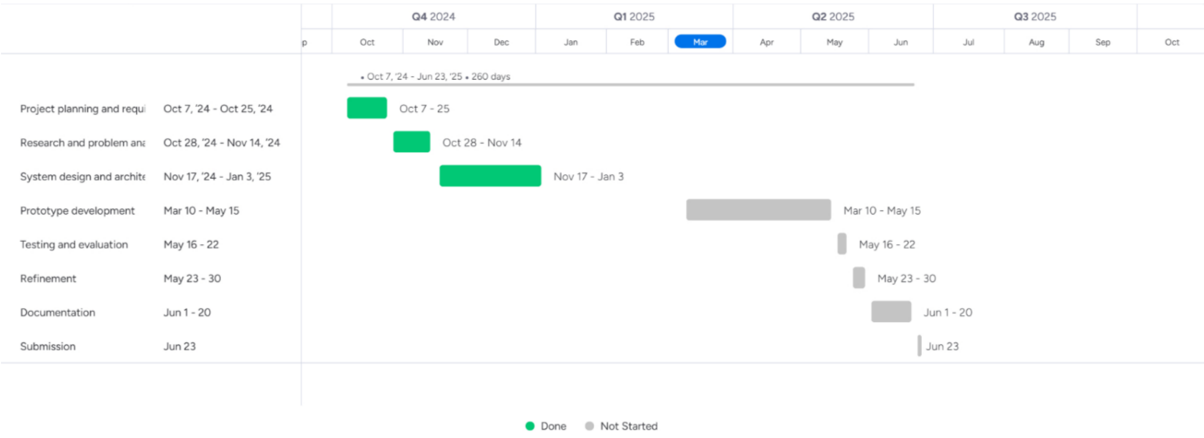


Figure 1.3: FYP project milestone

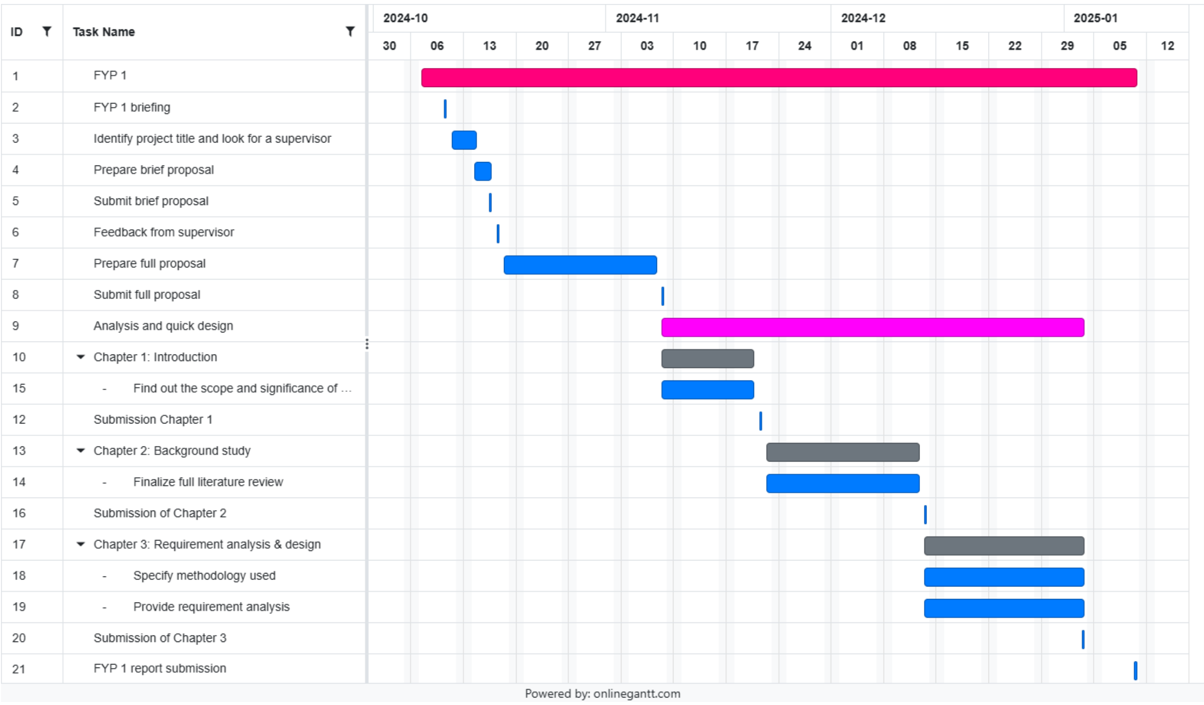


Figure 1.4: Gantt Chart of Schedule for FYP 1

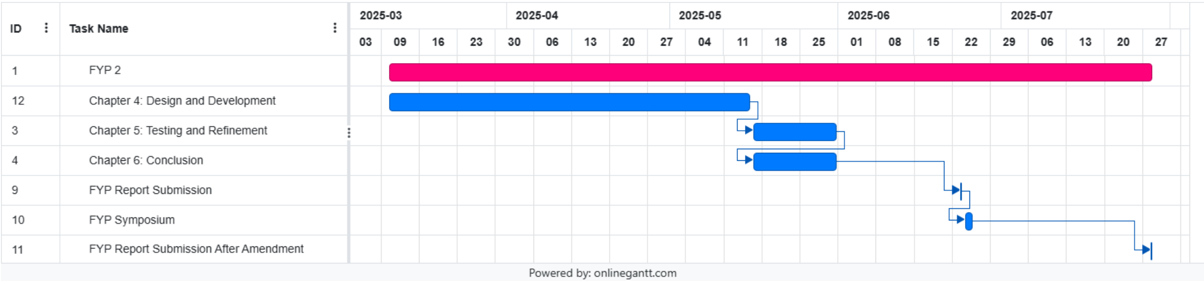


Figure 1.5: Gantt Chart of Schedule for FYP 2

1.8 Expected Project Outcomes

The outcome of this project should successfully create a user-friendly tool that provide a seamless and keyless entry for authorized users that improve user experience. By integrating IoT and AI technologies, the smart door lock system will offer advanced features such as real-time monitoring, remote access, facial recognition-based authentication, QR code guest access, and instant alerts via Telegram messaging. This project should provide a reliable smart door lock system that can prevent unauthorized access compared to traditional locking mechanisms that are prone to vulnerabilities such as key lost.

Through the use of a smartphone app, customers may remotely monitor and manage the door lock thanks to the IoT component. In the meantime, physical keys and password memorization are no longer necessary thanks to AI-powered facial recognition, which guarantees precise and effective identification of authorized individuals. The system is more dependable and resistive to unwanted entry due to the integration of IoT and AI, giving homeowners a solid and dependable home security solution.

1.9 Summary

In the context of Industrial Revolution 4.0, the "AIoT Smart Door Lock for Home Security" project seeks to revolutionize smart home technologies. This project will improve home security while providing a practical, hands-free solution by utilizing AI-based facial recognition and addressing flaws in existing systems. The goals, scope, and importance of the project have been outlined in this chapter, providing the groundwork for its advancement and assessment in the following chapters.

CHAPTER 2: LITERATURE REVIEW

2.1 Background

Home security has always been a big concern for individuals as it is to keep the safety of the residents, particularly to the vulnerable groups such as elderly and children. Apart from that the safety of the house is very important to specifically protects and shelter elderly and children. It also got criticized for its vulnerability and the entire way technology is developing brings about complete revolutions in the ways of security management. Home security has become one of the major concerns for the people nowadays and home security systems as well as smart homes has become a practice (Akasha et al, 2019). Conventional home security systems work but rely too much on manual operations or stand-alone gadgets such as CCTVs and alarm systems which may not be integrated or remotely accessible making them less efficient. In order to counter this problem one of the best ways to do it is by applying smart home system because it consists of security, privacy and power management as well as other functions (Abdulrahman et al, 2020).

A smart home can be monitored and controlled remotely which is a good approach for individuals to monitor the house. As stated by Qusay (2020) “A smart home system is defined as collection of sensors, actuators, communication devices, and computing devices that are connected to each other to provide homeowners with services and applications with minimum or no intervention” hence, the security and its safeness becomes a need and the demand is high due to various reasons. For example, IoT-enabled devices like smart cameras, door locks and motion sensors can communicate with each other and be controlled remotely via smartphones or web platforms like Blynk (Fitriyan et al., 2024). This system integrates various technologies like cloud computing, AI, and IoT. Changing the view of conventional home security system into a more intelligent, tightly manage and effective system. The IoT guarantees that houses enjoy a new degree of security, peace of mind and convenience by allowing devices to interact and work together. These technologies solve the drawbacks of traditional systems and paves the way to a more sophisticated and dependable home security system in the future.

Figure 2.1: A conventional mechanical lock requiring a physical key for operation. This type of lock is prone to vulnerabilities such as lost or stolen keys.

Figure 2.2: A smart home lock utilizing facial recognition technology for keyless entry. This system offers enhanced security and convenience compared to traditional locks.

There are many benefits to integrating these technologies. First, IoT makes it possible for devices to cooperate, creating a cohesive and effective security system. For example, motion sensors and cameras can be incorporated into smart door locks to identify intruders and notify homeowners in real time. Second, AI improves security with features like facial recognition, which can recognize authorized users and alert homes to potential threats when it detects unknown people. (Arif et al., 2020). Additionally, in order to solve the shortcomings of conventional centralized systems, blockchain technology is being investigated as a decentralized and secure means of protecting IoT networks. Additionally, smart home technologies increase energy efficiency and convenience. For instance, homeowners can remotely operate their security systems and save money and energy by using Wi-Fi and Bluetooth connectivity. By making it possible for homeowners to effectively monitor and manage their living areas, these technologies not only improve security but also provide piece of mind.

2.2 Security Mechanism

Artificial Internet of Thing (AIoT) linking Artificial Intelligence to the Internet with the object of developing intelligent, autonomous and secured systems. This technology makes decision on a trust-based which involves using multi attribute methods that ensure accurate and fast decisions. In home security, it enhances a purely IoT-based operation with smart decision making, advanced analytics, and real-time automation, which are critical to optimum security and user convenience. Smart decision-making refers to the ability of AI algorithms to analyse data and make informed decisions without human intervention. For example, to determine whether a user is an authorized user to enter the house, AI can evaluate facial recognition data in a smart door lock system. If the system detects an unauthorized user, the system has the ability to automatically sound an alarm or notify the homeowner if it identifies any suspicious activity or unauthorized user. These speeds up reaction times to any security risks and lessens the need for manual intervention.

Advanced analytics involves the use of AI to process and interpret large amounts of data collected by IoT devices. AI is used in advanced analytics to handle and analyse vast volumes of data gathered by Internet of Things devices. For example, a smart home security system can identify suspicious activity (like a burglar trying to break in) from routine activity (like a family member visiting the house) by analyzing patterns in motion sensor data. This improves overall security by enabling the system to deliver more precise and useful information. Aside that, Real-time automation enables IoT devices to respond instantly to changes in their environment. For instance, a smart camera can immediately trigger the facial recognition system and unlock the front door for authorized people if it detects motion at the door. By eliminating the necessity for manual processes like utilizing a physical key or inputting a PIN, this smooth automation increases user convenience.

The IoT are important as it is to any other appliances because it is found everywhere in daily life. It provides many solutions that will benefits people in achieving daily tasks. As stated by (Mohamad and Chad, 2019) “The Internet of Things (IoT) is a vision for and internetwork of intelligent, communicating objects such as home appliances, vehicles, factory machines, wearable devices and various types of sensors”. The IoT and Artificial Intelligence (AI) combines to enhance the performance of the technology and ensuring the availability, integrity and confidentiality for interconnected devices but it also have its unique security challenges. AIoT enables the connection of previously unconnected devices by relates it to the fast-growing technologies and web sensing devices that can accomplish reliable communication, intelligence technology, and data recognition. However, it leads to vulnerabilities to attacks and resource constraints that lack computational power. IoT devices can be targeted by hackers attempting to take advantage of network flaws because they are networked and frequently rely on wireless communication (e.g., Wi-Fi or Bluetooth). For instance, home security could be jeopardized if an intruder were to obtain illegal access to the system or intercept data being transferred between devices and due to their limited processing power, many IoT devices—including sensors and cameras—rely on third-party systems, such as cloud servers, to carry out complicated tasks like data analysis and facial recognition. In the event that the external resources are overloaded or unavailable, this reliance may result in delays in decision-making and raise the possibility of system failures.

The fusion of these two technologies includes multiple data sensing devices like the Radio-Frequency Identification (RFID), cloud, and wireless sensor networks (WSN) gives a complete service but also depends on the integration (Sumathi et al, 2023). AIoT's capacity to connect cognitive technology with real-world applications is transforming many industries, most notably home security, where it provides a unique combination of dependability, effectiveness, and creativity.

2.3 Technology for Home Security

With the incorporation of the IoT, home security technology has advanced drastically offering homeowners to increase the convenience and protection of their devices. This technology monitors and defends homes against threats and invasions by utilizing a variety of smart devices and systems. As modern home security relies heavily on IoT technologies to monitor and regulate home environments remotely, these systems is bound and packed with sensors, cameras, and alarms systems that may identify safety concern cases like intruders and gas leaks. They can then send homeowners real-time notifications via mobile devices. The system is improving to make it easier with the use of web servers and software platforms like blynk (Fitriyan et al., 2024). To further improve home security, cutting-edge technologies like blockchain and artificial intelligence are being investigated. Facial recognition is used by AI-based systems to recognize known faces and notify homeowners of possible dangers when unknown faces are found. By solving the shortcomings of conventional centralized methods, blockchain technology provides a decentralized method of protecting IoT networks (Arif et al., 2020). Furthermore, the smart home security connectivity that uses both WiFi and Bluetooth technologies can enhance the security, and it is energy efficient as well as cost-effective to connect via Bluetooth. By facilitating these technologies, it will give homeowners the ability to monitor and manage their living spaces efficiently.

2.4 Review of Existing Systems for Home Security

In this chapter, we will review three (3) similar existing systems as well as the tools that will be used in this project. Furthermore, a brief overview and key features of the proposed system will be included in this chapter.

2.4.1 IoT Enhanced Smart Door Locking System

In this paper, (Shanthini et al., 2020) proposed a smart door locking system that uses Bluetooth for user authentication using smartphone app. The objectives of the proposed system are to enhance the security in smart homes using Arduino UNO through a Bluetooth-controlled technology that allows the owner to monitor the buildings with their smartphone. The system eliminates the needs for physical key and provide security features like popup warnings for invalid access attempts that will trigger alerts through a buzzer and user credentials.

The method used in is by the door sending signal from a smartphone app via Bluetooth to the Arduino UNO that controls the servo motor. The system work when user enter their password and unique id in the app, the application will enable the 'LOCK' and 'UNLOCK' buttons. If the user clicks on 'UNLOCK' button then the apps will send command to the Arduino to activates the servo motor to unlock the door. But if the credentials are incorrect then it will trigger the buzzer alarm and alert SMS to the homeowner.

Figure 2.3: Valid credential and control button (Shanthini et al., 2020).

Figure 2.4: Valid credential and control button (Shanthini et al., 2020)

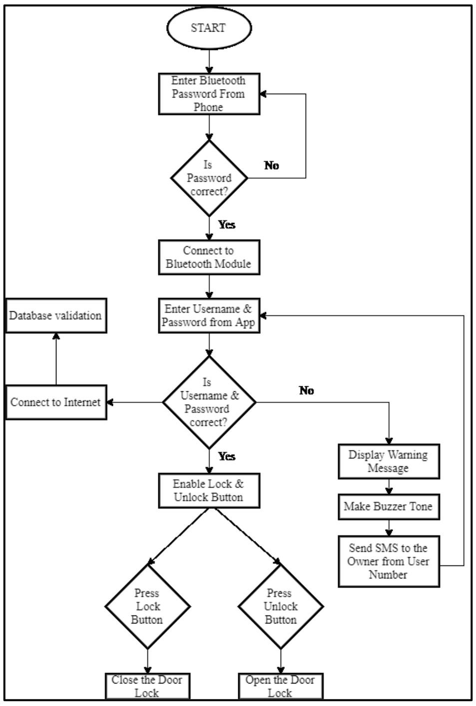


Figure 2.5: Flowchart of the system (Shanthini et al., 2020)

The advantages of this system are that the system enhanced the security mechanism through automated alerts for unauthorized access and the use of smartphone that is easy to navigate and use. Furthermore, it is a low-cost system and it is simple to install for user.

2.4.2 Design of a low-cost NFC Door Lock for a Smart Home System

In a research Design of a low-cost NFC Door Lock for a Smart Home System, the paper proposed a system that utilizes an NFC door lock that aims to be a user friendly based using Arduino boards as microcontroller (Pacheco & Miranda, 2020). In this paper, the system that was presented is based on the Near-Field Communication (NFC) technology and the Arduino Mega board that is a low-cost lock. The prototype is easy to implement and installed for control access.

The system's prototype uses three module which is Arduino module which is the Arduino Mega 2560, the NFC module which is the NFC Shield ITEAD PN532 that is intended to read the NFC devices and tags and lastly is the software module which they use Unity as the interface to interact with user. In this software, user can lock and unlock the door remotely using their smartphone and can see the graphic feedback when user utilizes the NFC tag to access.

The system work when the Arduino board connect to the physical lock and the microcontroller will takes action to lock and unlock the door when meet the authorized user. Other than that, the NFC tag is put inside a card and between both components there is NFC Shield that reads the tag and sends the information to Arduino to verify the user and act accordingly.

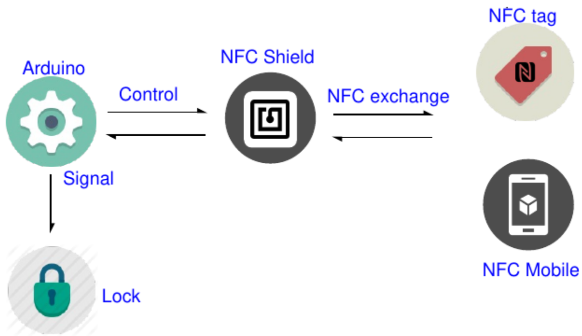


Figure 2.6: Component's interaction (Pacheco & Miranda, 2020)

Figure 2.7: Homepage of the system (Pacheco & Miranda, 2020)

Figure 2.8: Example of authorized user (Pacheco & Miranda, 2020)

The advantages of this system are it is easy to install and it is scalable on-demand because of the independent features the device hold. Other than that, the system uses Serial Peripheral Interface (SPI) mode to support Arduino's library for the NFC module and since the protocol is simple it is fast to transmit data continuously.

2.4.3 Smart Door Locking System

The research is done in Kumaraguru College of Technology Coimbatore, India which (Aswini et al., 2021) proposed an RFID-based door lock system along with OTP-driven technology to provide a high security solution to a household. The OTP feature is generated for door access with expiry time for visitors' access, reducing manual intervention and enhancing security. The proposed system is a cost-effective system and suitable for various application.

The system uses Raspberry Pi that act as the main control unit for the locking system which gets activated when the user presses a switch that will automatically notify the owner about the arrival of the person. The user can unlock the door using RFID tags, OTP PINs or using the keypad for password entry. Other than that, for OTP generation, the OTP will be generated when access is requested the system cloud will generate a random number that has expiry time to the user and upon successful authentication, the Raspberry Pi will activate the servo motor and electromagnet to unlock the door.

The system is convenience for the users because it eliminates the need to remember passwords and it provides multiple access options that is user-friendly. Other than that, the system eliminates any concerns of losing physical keys or theft intrusion that provides real-time monitoring features.

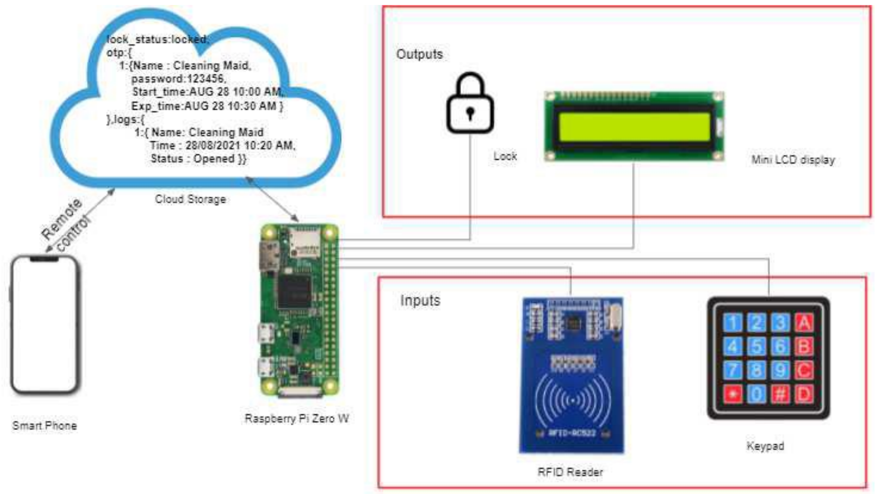


Figure 2.9: Overview of the proposed system (Aswini et al., 2021)

Other Smart Door Lock Existing System Papers

Table 2.1: Existing Smart Door Lock System Paper

Papers	**Descriptions**
Smart and Real-Time Door Lock System for An Elderly User Based on Facial Recognition	The smart door is design to ease the use of elderly to open the door using **facial recognition** features without the use of physical key (Saputra & Surantha, 2021).
Smart Door Locking System Using Wireless Communication Technology	The smart door is design to detect human using **Human Interface Remote Terminal Unit**

	(HIRTU) that will send signal to the door (Kumbhar & Deshmukh, 2017).
Smart Door Locking System using IoT	The smart door is design as a low-budget biometric lock **using fingerprint sensor in smartphone** (Patil et al., 2020).
IoT-Enabled Door Lock System	A **GPS-based smart door lock** is designed aiming to design a door lock system that does not need manual input from user for convenience purpose while also remaining secure (Adiono et al., 2019).
A Smart Door Lock Security System using Internet of Things	Smart door lock using **mobile application to automate admission** of illegal visitors by capturing their photograph using camera (Doshi et al., 2022).
Password Protected Electronic Lock System for Smart Home Security	Smart door lock using **electronic embedded lock** comprises of a keypad and LCD with microcontroller (Rahman et al., 2018).
Smart Security: An IoT-NFC Lock System for Efficient Access Management	Developing advanced **smart padlock system** (Shaik et al., 2024).

Existing Smart Digital Door Lock

Table 1.2: Existing Smart Door Lock System

Products	Features offered
Lockin V5 Max Palm Vein Digital Smart Lock with Viewer	Palm vein recognition 3D face recognition Real-time monitoring Smart doorbell
Kaadas K30 3D Face Recognition	Inside handle with in-built sensor
Philips EasyKey 702-1HW	Motion-sensitive PIR that takes photo or video Two-way intercom Multiple alert function
Kadonio Smart Handle Door Lock	Multiple locking & unlocking door system
Yale YDR343	Accessed via smart keypad or RFID card
Aqara Smart Lock A100 Zigbee	Multiple unlock method and integrated multiple sensors
Philips Digital Lock Easy Key 515K	Anti-dismantling alarm
Eva M5 Smart Door Lock	Syncs with mobile apps Multiple security options including one-time passwords
Philips Digital Lock EasyKey 303 VP	Emergency quick-open handle Multiple alert functions Convenient one-click remote unlock

2.5 Comparison Table Between Existing and Proposed System for Home Security

Table 2.3: Comparison table between existing and proposed system

Features \ Systems	IoT Enhanced Smart Door Locking System	Design of a low-cost NFC Door Lock for a Smart Home System	Smart Door Locking System	Proposed System
Authentication Method	Credential-Based authentication (Username and Password)	Near-Field Communication (NFC)	Multi-factor authentication using RFID tags and one-time passwords (OTPs)	Facial-recognition, QR code (guest access/temporary authentication)
Connectivity Options	Bluetooth	NFC Module and Serial Peripheral Interface (SPI)	Mobile applications, Bluetooth and Wi-Fi	Wi-Fi
Main Hardware Components	Arduino UNO Bluetooth module (HC-05) Piezo Buzer	Arduino Mega 2560 NFC Shield (ITEAD PN5532)	Raspberry Pi Zero W RFID Reader Keypad LCD Display	Raspberry Pi 4 Passive Infrared motion sensor (PIR)

User Interaction	Mobile application	Graphical feedbacks using Unity 3D	Mobile application	Mobile application
Notification Method	Buzzer only	None	Mobile alert	Telegram bot and buzzer alerts
Power Source	USB connection and external power supply	Multiple power source: USB connection AC/DC adapter Battery	AC/DC adapter and battery backup	AC/DC adapter

As seen in table 2.3 above, the proposed system uses higher-performance hardware, such as the Raspberry Pi 4, for on-device AI processing and presents a unique combination of facial recognition and temporary QR code access that is not present from the reviewed systems.

2.5.1 Analysis

Based on the table above, the comparison between three existing systems and a proposed system have been made. The three system which is IoT Enhanced Smart Door Locking System, Design of a low-cost NFC Door Lock for a Smart Home System and Smart Door Locking System shows a diversity approach. The existing system above all uses credential-based method to authenticate user which is by using username and password, one-time password, near-field communication and RFID. This method is effective but it can be compromised. The level of connectivity varies ranging from basic Bluetooth which is less secure to a combination of Wi-Fi, NFC, and SPI to have a greater functionality.

Shanthini et al., proposed a system that uses a unique username and password to access the door which if the credentials are invalid then systems will trigger a buzzer alarm that will notify the owner of unauthorized access has been made. The NFC authentication systems by Pacheco & Miranda, ensures that only authorized users with valid tag can unlock the door and the multi-factor authentication system by Aswini et al., enhanced the security of the systems.

Other than that, the existing systems focuses on using a basic component for example Shanthini et al., uses Arduino UNO and Bluetooth module in their proposed systems that indicates a limited automation and intelligence. The proposed system on the other hand, uses advanced components such as PIR motion sensor, Raspberry Pi 4 as the main processor and Raspberry Pi camera module V2 which is an AI camera that enable real-time facial recognition and intelligent motion detection. Furthermore, the existing systems uses mobile application and graphical feedback that indicates to limited versatility for users without advanced devices like smartphones.

These existing systems leads to a proposed system that uses facial recognition as its authentication method offering a biometric-based security that is resistant to duplication and theft as well as a hands-free locking mechanism. A unique QR code authentication method is also implemented for easy access process for guests visiting such as family members or close friends, adding to user-friendly system. Apart from that, it uses Wi-Fi for local connectivity and remote access that enhance conveniency and flexibility of a system. Furthermore, the system uses Blynk mobile application user interaction platform that allow user to know the activity, events and data from the systems. Telegram application is for alerting user with real-time notification when there is an unauthorized user try to walk through the premises. Hence, it outperforms the existing systems in term of security due to the biometric-based security, intelligence of the system due to the use of PIR motion sensor that detects movement, Blynk and Telegram application as the user interaction platform and security alert system respectively and the connectivity Wi-Fi which is much faster and reliable.

2.6 Critical Review

In terms of intelligence, security, and usability, the suggested AIoT Smart Door Lock system demonstrates notable improvements over existing systems. Using AI-powered facial recognition for biometric authentication is one of the major advancements. Facial recognition offers a very safe, contactless form of identification in contrast to credentials or physical tags that are susceptible to loss or theft. It complies with contemporary security standards by reducing the likelihood of duplicate or unwanted access. Furthermore, the system uses a PIR sensor for intelligent motion detection, which only activates facial recognition when motion is detected. This enhances system responsiveness while minimizing on unnecessary processing. For guest access, a QR code authentication feature is introduced, enabling temporary and secure access without being included to the facial recognition database. The addition of real-time Telegram alerts, which immediately notify the homeowner in the event of unauthorized access attempts, is a significant advancement. This greatly enhances situational awareness and response time, compared to systems that rely solely on buzzer sounds or visual indicators.

However, the cost and complexity to develop the proposed system are one of the drawbacks especially with the use of an AI camera and Raspberry Pi which is more expensive than the simpler Arduino-based systems. Other than that, the power consumption of this system might consume more compared to the existing systems and it requires an efficient power management strategy. Another potential challenge lies in calibration of the intruder deterrence to measures false triggers such as turning the system on for benign motions or animals which could cause needless disturbances. Despite these challenges, the proposed system offers a smarter, safer, and user-friendly solution that fixes the limitations of current smart door locks and is in line with the direction of smart home security.

2.7 Summary

In this chapter, three similar existing systems were analysed to identify the gap and weaknesses to proposed a new system that can improve the weaknesses. The systems were compared with each other and the advantages of each system been addressed briefly for better idea and overview.

CHAPTER 3: METHODOLOGY

3.1 Introduction

In this chapter, it outlines the methodology used to develop the AIoT Smart Door Lock system following the prototype model. The prototype is an iterative approach that involves five key phases which is requirement gathering, design and development, evaluation, refinement, and documentation. Each stage is designed to ensure the system meets its objectives while addressing the security limitations of traditional smart door locks. The chapter provides a thorough discussion of the progress made in each step as well as what will be done in moving forward. It begins with the requirement gathering stage, which surveys and research were done to determine the system requirements and user demands. The Design and Development phase comes next, which the system architecture, block diagram and flowchart were made using the requirements that were acquired. User Acceptance Testing (UAT) is used in the evaluation phase to gauge user happiness and system functionality. The Refinement phase focuses on enhancing the prototype to satisfy all needs based on the feedback provided and all of the findings, plans, and enhancements are finally compiled into a thorough report during the documentation phase. The project guarantees a methodical and iterative development process by adhering to this organized strategy, enabling ongoing enhancement and user needs alignment. This chapter serves as a development roadmap for the AIoT Smart Door Lock, outlining the actions done and the advancements made in achieving the project's goals.

3.2 Requirement Gathering Analysis

This phase involved identifying the needs of the target users, defining the project objectives, and listing the key features and requirements for the system. An analysis of current smart door lock systems was done in order to pinpoint the shortcomings and the gaps that this project seek to fill. Information on user preferences and technological trends was gathered through literature reviews. Based on this research, the following requirements were established:

- Functional Requirements: Facial recognition, real-time monitoring, remote access and motion detection.
- Non-Functional Requirements: User-friendly system, reliable and energy efficiency.
- Hardware Requirements: AI camera, PIR motion sensor, servo motor, piezo buzzer, and Raspberry Pi 4.
- Software Requirements: OpenCV, MediaPipe, Python, and Blynk for mobile interface.

3.2.1 Identify the user

The targeted user for this project is for homeowners and elderly. They are the main aims for this system since some homeowners need to enhance their home security in order to protect their house from theft and threats. Aside from that, elderly is not encourages to carry important keys around because they might lost the keys or if they are too young, they might not know how to unlock the door from outside. Additionally, the system can also benefit small businesses looking to improve access control and enhance security at their premises. Hence, this system aims to help ease the process to access the door remotely and hands free without having to do extra work in order to enter their building.

3.2.2 Hardware requirement

The proposed solution will require this hardware to build the prototype of the system which consists of AI camera, the camera module and other tools. Table below shows he details of the hardware required for this prototype.

Table 3.1: Hardware Requirements

Hardware	**Specification**
AI Camera	Raspberry Pi Camera
Raspberry Pi 4 Model B	Central Processing Unit
Passive Infrared (PIR) Motion Sensor	Detects movements using infrared radiation
Servo Motor	Control automatic door lock and unlock
Piezo Buzzer	Audio alert for intruders
AC/DC Power Supply	Provides consistent power supply

Based on the table above, the proposed system uses a Raspberry Pi 4 AI camera, Raspberry Pi camera module, servo motor, PIR motion sensor and AC/DC power supply for the hardware components to build the prototype. Raspberry Pi 4 act as the central processing unit that works together with its AI camera module to capture images and process those data using AI algorithms. The camera module is connected to the Raspberry Pi via Camera Serial Interface (CSI) port and captures a high-resolution image. From the data gathered by the module, the Raspberry Pi 4 will process the camera input using AI frameworks and then it will integrate with the control systems which is the servo motor.

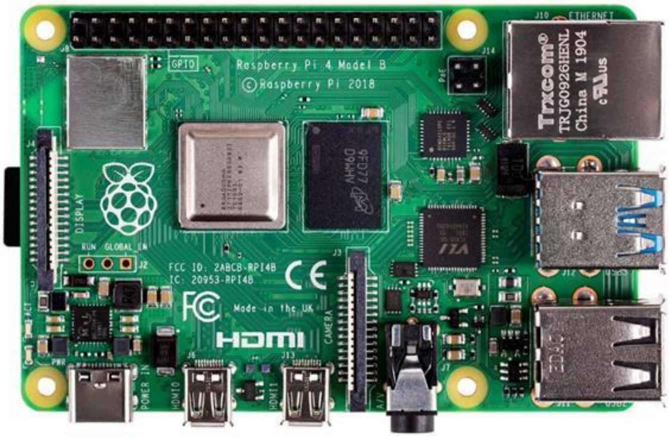


Figure 3.1: Raspberry Pi 4 Model B

Alongside with the Raspberry Pi 4, AI camera and its module, a PIR sensor is another important component for this project where it detects motion by sensing heat that is emitted by objects. Any objects that emit heat will be detected by the sensor because PIR sensor has pyroelectric material that is sensitive to infrared radiation can detect infrared radiation. Hence, it will trigger the output which it will send a high signal (ON state) to the output pin.

Figure 3.2: PIR Motion Sensor

3.2.3 Software requirement

Table 3.2: Software Requirements

Software	Explanation
Raspberry Pi Operating System	The software is needed to ensure the smoothness in running the Raspberry Pi hardware and to make sure the stability and control of the components.
OpenCV	OpenCV is for image processing and to do facial recognition.
MediaPipe	MediaPipe is an open-source framework developed by Google that can run machine learning applications in real time on edge devices like smartphones and Raspberry Pi. It is effective, lightweight solutions for applications such as iris identification, hand and motion tracking and face detection.

Blynk	The platform provides interface for users to access the system remotely for remote control and monitoring of the door lock using smartphone.
Playsound	Audio Playback is required to generate warning sound. It is lightweight and straightforward without any advanced setting up needed.

3.3 System Design

3.3.1 Block Diagram of Proposed System

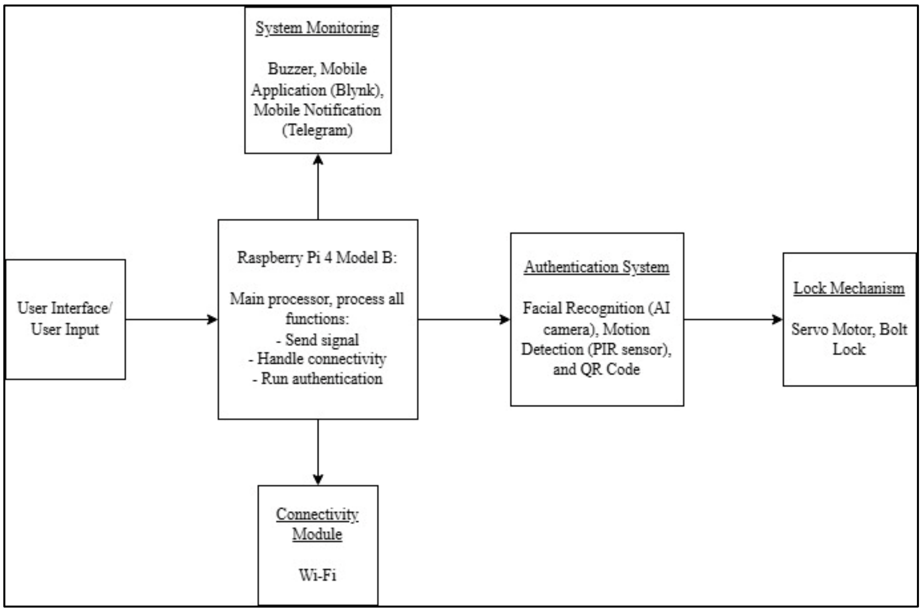


Figure 3.3: Block diagram for the proposed system

The diagram above illustrates the architecture and workflow interactions of the core components of the proposed system. The diagram from Figure 3.3 visually outlines the integration of input devices, processing unit, and output actions required for real-time authentication and control as well as the authentication and monitoring system.

The Raspberry Pi 4 serves as the system's central processing unit, managing Wi-Fi connections, facilitating component communication, and executing the authentication operation. It receives signals from the PIR sensor, AI camera, and QR code scanner module and processes this data determine access rights. When the PIR sensor detects motion, the AI camera takes a picture and uploads it to the Raspberry Pi so that OpenCV and the face_recognition libraries may process it for facial recognition. The servo motor is activated by the Raspberry Pi to unlock the door if the captured image is identified as authorized user. The door stays locked and a buzzer are triggered to alert individuals in nearby areas if the image is not recognized. The system notifies the homeowner simultaneously by sending an alert message using the Telegram Bot API and including a link to the photo that was taken.

The homeowner can create a temporary QR code using the Blynk mobile application in the event that visitors need access. When the visitor scans this QR code at the door, the Raspberry Pi unlocks the door to provide temporary access if the QR is verified. The user can also review system logs, register a new user, set up alarms, and manually lock and unlock the door using the Blynk app. Real-time control and monitoring are ensured by the Wi-Fi connection between the Raspberry Pi.

3.3.2 Flowchart of The Proposed System

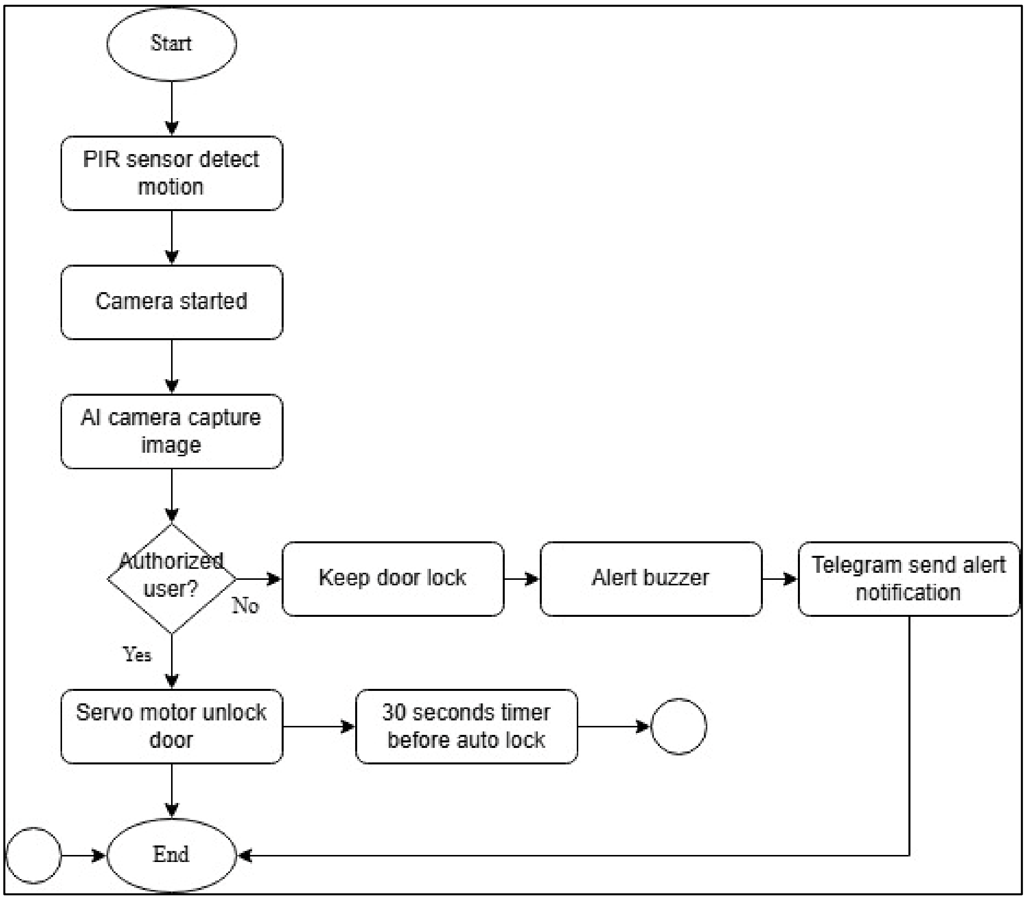


Figure 3.4: Flowchart of the proposed system

Figure 3.4 shows the sequential logic behind how the proposed system operates from startup to decision-making during access attempt. The system starts by activating PIR sensor to monitor the surroundings for any motion in which the sensor will scan the area within a few meters continuously. When motion is detected, the Raspberry Pi is instantly activated by the system to take a snapshot of the person at the door. The taken image is then processed by the system using facial recognition algorithm, which compares it with a dataset of authorized users that has been previously stored to determine if the user is authorized. If a match is found, the servo motor is used to automatically unlock the door allowing access to the user to enter the property without the need for PINs or physical keys.

The door stays lock and the system will activate an alarm system that act as a warning to deter possible intruders if the image captured does not match any authorized image data. The image of the unauthorized user is saved for reference and logging in the system and Blynk application. In parallel, the system listens for input from a QR codes generated by the owner, scans it to the camera. The system skips the facial recognition stage and unlocks the door if the code is valid for a guests or visitors temporary access to the system.

Following each decision whether to grant access or deny it – the system resets and returns back to its initial state, prepared to identify the next motion event. This loop maintains a streamlined decision flow centered around motion, recognition, and response while enabling ongoing monitoring and interaction.

3.3.3 System Architecture

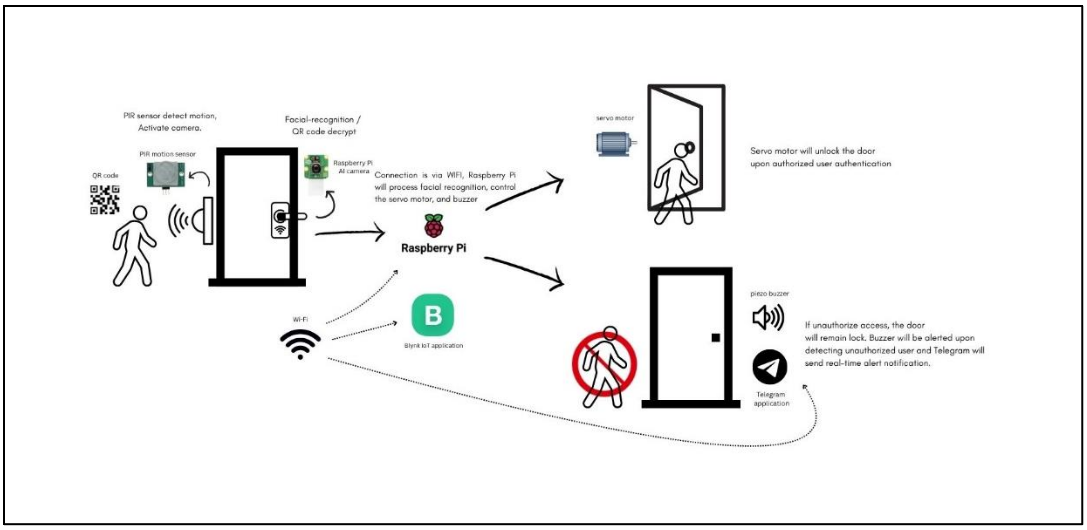


Figure 3.5: General system design and architecture

The system architecture provides in Figure 3.5 above shows the overall system overview of the hardware and software components used in the AIoT Smart Door Lock system and explains how they interact to perform key functions such as motion detection, facial recognition, guest access, and real-time alerts.

The Raspberry Pi 4 at the center of the design managing the door lock mechanism, processes image data, coordinates sensor input, and connects to external services like Telegram and the Blynk app. When the PIR motion sensor senses movement close to the door, the system starts up. This causes an image of the subject to be captured by the Raspberry Pi Camera Module v2. In order to compare the image with the recorded encodings of authorized users, it is locally processed on the Raspberry Pi using the OpenCV and face_recognition libraries. The lock system automatically unlocks the door by turning on the servo motor if a match is detected.

The system saves the snapshot to the unauthorized_access/ folder, keep the door lock, and activates a buzzer if the image is not recognized. Simultaneously, a Telegram notification with a link to the taken image included is sent to the owner to notify them of a possible intruder activity.

The homeowner can use the Blynk app to create a QR code guest access in cases where short-term access is required. Without being added to the facial recognition database, the visitor can unlock the door by scanning the code with an additional module that is connected to the Raspberry Pi. This feature offers secure, time-limited access to the guests or visitors making it easier for them to access the property.

3.4 System Development

The development of the system which is the prototype and the application platform will be built and explanation will be more in-depth and more detail in the next chapter. The development phase will be installing the required frameworks and libraries on Raspberry Pi, configure facial recognition, motion detection features with Blynk using Python programming language. Plus, all hardware and software components will be tested to produce a working prototype.

3.5 System Evaluation

3.5.1 User Acceptance Testing

The evaluation phase in the methodology will be conducted by User Acceptance Testing (UAT) to ensure the prototype meets the requirements and achieve its objectives. The UAT is a process that structured to gather feedbacks that is constructive and identify any area that needed improvements. This can help in evaluating the system's functionality whether it meets the requirement or not while gathering the feedbacks from respondents to refine and improve the system.

For the process, around 10 respondents will be recruit that vary from homeowners, students and elderly individuals. The prepared prototype system will be used in this process and all key features such as facial recognition, PIR motion sensor functionality, the emergency alarm, Blynk mobile application, and QR code functionality to the system will be included as well. The evaluation will be based on a prepared scenarios to test on:

- Facial recognition smart door lock functionality.
- Simulate unauthorized access attempt and its response.
- Use the Blynk application to control the door lock manually.
- Generate and scan QR code for temporary access.

The success rate and user opinions will be taken as the feedback for this process and a Likert scale will be provided to the respondents to rate the system. Other than that, the interactions between respondents and the system will be recorded. All of the above method and process will be analysed to identify the issues and actionable insights. Hence, the UAT phase will help in providing inclusive feedback that can help to improve the proposed system.

3.6 Refinement

The prototype will be improved to address problems or flaws in light of the feedback received during the evaluation phase. For instance, the algorithm will be adjusted to increase processing speed if users complain about delays in facial recognition. Likewise, the PIR sensor's sensitivity will be changed if the motion detecting system sets off false warnings. Until the prototype satisfies all specified requirements and accomplishes the project goals, this iterative process will continue.

3.7 Documentation

The final step in the methodology is to document every data and process. In this phase, every project-related finding, data, activities, and outcomes will be compiled into one single document that will serves as the project report. The documentation will used word processing tools such as Microsoft Word for writing and any chart and diagram is using Draw.io. The report will be an important resource for assessing the project's performance and offering suggestions for further research because it meticulously documents each stage of the project.

3.8 Proposed System

The proposed system for this project is AIoT Smart Door Lock for Home Security that aims to develop smart door lock using facial recognition to identify authorized users. This system is to enhance the security and user experience when accessing the door. This system integrates with AI camera and then will notify Raspberry Pi to send information to the servo motor to unlock the door.

3.9 Summary

In conclusion, this chapter analyses the requirements that is needed to develop the systems. The design is made based on the requirement gathering phase from Chapter 1 and Chapter 2. The system architecture and general design is included to give a better understanding about the proposed system. Additionally, in this chapter the methodology used to develop the system is explained in more details. In the next chapter, more explanation will be included on the system development of the proposed system.

CHAPTER 4: DESIGN AND DEVELOPMENT

4.1 Introduction

In this chapter, the development of the prototype will be discussed further in details. It focuses on the requirement analysis and design implementation of the proposed system in Chapter 3. The implementation falls within the prototype cycle, which is design and development phase. In this phase the step is repeated until a fully functioning prototype is built and prepare it for delivery. Every implementation feature directly relates to the goals outlined in Chapter 1 to ensure that the system design is in line with the project objectives. Objective 1, which focuses on integrating AI and IoT for real-time secure access, is fulfilled through facial recognition using the Raspberry Pi camera, OpenCV, and the Blynk IoT platform and objective 2 which is to evaluate the prototype through UAT in the real-world performance to collect systematic feedback from user. The system's functionality is further reinforced by other features like Telegram notifications and QR guest access, which show a thorough implementation of the AIoT idea for smart home security.

4.2 Feature Implementation

A few software needed to be installed and setup ahead of time before starting the development on the proposed system to ensure smooth integration and development of all system components. The system, which runs on a Raspberry Pi 4, uses Python as its primary programming language for the back-end. For facial recognition processing, Python libraries like OpenCV and MediaPipe were installed, and audio alerts were triggered using the Playsound module. These tools were chosen in order to provide an accurate and real-time authentication system through the AI camera module. The Blynk mobile app was used to provide an intuitive front-end interface that allows users to see door status, receive notifications, and, if needed, make manual modifications. This platform was selected due to its simplicity of use and compatibility with Raspberry Pi.

The system was configured with Wi-Fi modules to enable communication between the mobile application and the door lock. By using Blynk, an IoT device monitoring and control platform, as the remote interface, the homeowner was able to lock and unlock the door and get real-time alerts on their smartphone.

A PIR Motion Sensor was set up with the Raspberry Pi to process motion activity by detecting movement in surrounding areas and starting the face recognition process. The door lock system was physically operated by servo motors. The Raspberry Pi 4 was used to integrate and test every component of the system, including real-time motion detection, facial recognition, and audio alerts (buzzer), and mobile app notifications, to guarantee full performance.

4.2.1 Hardware Design and Setup

The hardware setup for the system consists of ten essential components that work together to perform motion, detection, facial recognition, access control, and alert notifications. Table 4.1 summarizes the components, the quantity used and its functions.

Table 4.1: Lists of hardware and quantity used.

Components	**Quantity (pcs)**	**Function**
Raspberry Pi 4 Model B	1	Main controller
Raspberry Pi Camera Module V2	1	Capture an image
PIR Motion Sensor	1	Detect motion or movements

Servo Motor	1	Control physical lock mechanism
Bolt Lock	1	Lock mechanism
Piezzo Buzzer	1	Audio alert
Breadboard	1	Prototyping hardware connection
Jumper wires	<20	Electrical connections
Power supply	1	Provide power to the components

4.2.2 Software Installation and Configuration

4.2.2.1 Setting Up Raspberry Pi 4

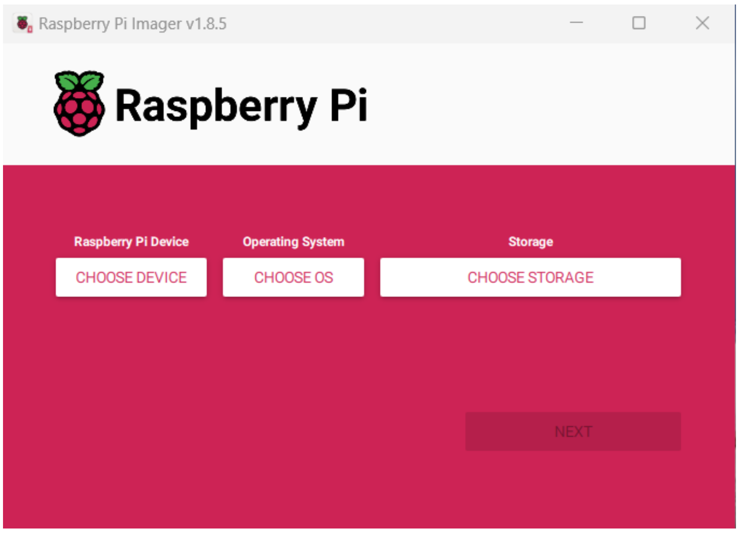


Figure 4.1: Raspberry Pi imager setup – Raspberry Pi imager interface; choosing the device, operating system and storage used.

Before starting the development of this system, the Raspberry Pi 4 needed to be prepared with appropriate operating system. The Raspberry Pi imager was downloaded on the official website, https://www.raspberrypi.com/software. After downloading the imager, the setup for user project needs to be chose first before integrate it into the

Raspberry Pi. The device used, operating system that the project will be working with and the storage of the system was asked to set.

Figure 4.1 illustrates the Raspberry Pi Imager interface, where the Raspberry Pi 4 Model B was chosen as the device used. This project chose Raspberry Pi OS Lite (64-bit) as the operating system because it is lightweight and suitable for headless operation without a graphical interface and a microSD card was chosen as the storage device to hold the OS and project files.

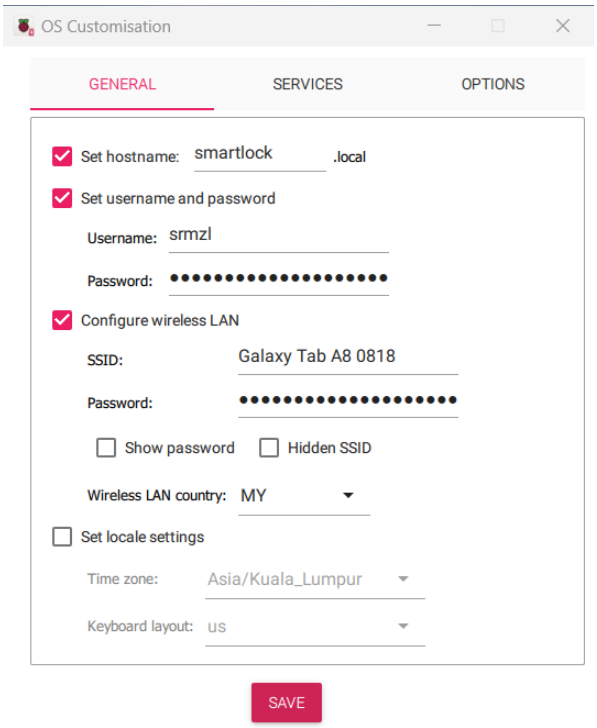


Figure 4.2: Raspberry Pi imager setup – setting up hostname and password for ssh

A secure, effective, and headless development environment is essential for an embedded system such as an AIoT door lock, and it was the goal of the Raspberry Pi 4's initial configuration. Several significant modifications were made in the Raspberry Pi imager system, as seen in Figure 4.2. By pre-configuring the Wi-Fi credentials and assigning a unique hostname (smartlock), the device can be easily recognized on the network and connect instantly when it boots up. This creates a basic layer of protection against unwanted network access, along with creating a strong, non-default password.

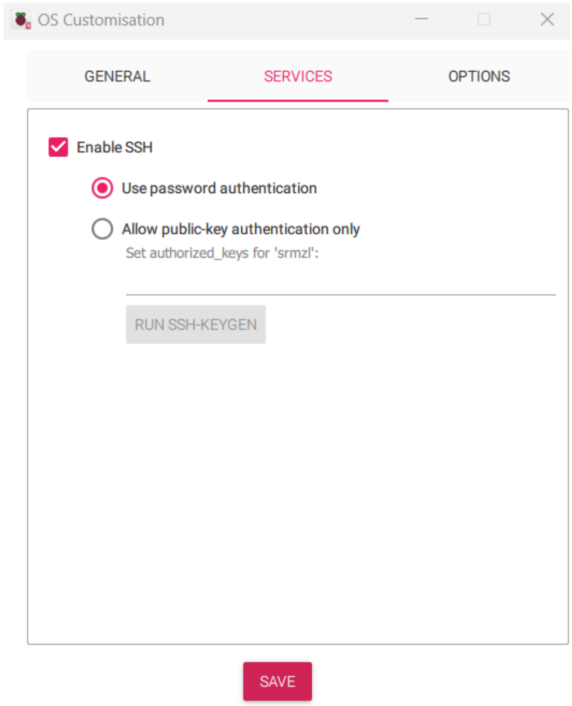


Figure 4.3: Raspberry Pi imager setup – enable SSH setting.

Additionally, enabling the Secure Shell (SSH) protocol, as seen in Figure 4.3, was essential for facilitating a "headless" workflow. With this approach, the Raspberry Pi does not require a separate display, keyboard, or mouse to be physically connected to it. Rather, from a primary development machine, it enables safe, encrypted remote access. This simulates a real-world deployment scenario where the device would function independently without direct physical peripherals, in addition to streamlining the development process by permitting the use of powerful tools like Visual Studio Code with its Remote-SSH extension. Therefore, these first actions served as the foundation of developing a reliable and professional platform for testing and development.

4.2.2.2 Setting Up Python Environment

After setting up the Raspberry Pi OS, the next step was to configure the Python environment inside the operating system. Python version 3.12 was installed to serve as the primary language for the system development. In order to control dependencies and ensure interoperability with different libraries, a virtual environment named myenv was created using the command python3 -m venv myenv. The necessary libraries such as face_recognition, RPI.GPIO, and blynklib, were installed using command pip once the environment was enabled using source myenv/bin/activate. This configuration made it possible to create a clean, isolated environment that was customized to meet the demands of the project.

4.2.2.3 Connecting Raspberry Pi via Visual Studio Code (VS Code)

Connecting to the Raspberry Pi connection via SSH:

ssh <username>@<raspberry-pi-IP>

Connecting Raspberry Pi on VS Code using SSH:

Ctrl + shift + P → **Remote-SSH: Connect to Host...** → <username>@<raspberry-pi-IP>

Enter password, select platform: Linux.

4.2.2.4 Lists of Libraries used

Table 4.2 below shows the lists of Python libraries used in this system throughout the process.

Table 4.2: Lists of Third-Party Libraries used.

Third Party Python Library	**Justifications**
RPIO.GPIO	To interact with Raspberry Pi GPIO pins
face_recognition	For face detection and recognition
PIL (Pillow)	For image processing for QR scanning
pyzbar	To decode QR codes
BlynkLib	To connect to the Blynk application
requests	For uploading images to imgbb API
qrcode	For generating guest access QR code

4.2.2.5 Folder Structure and Project Path

For an effective development, testing and deployment on the Raspberry Pi environment, the whole system is developed and organized within */home/srmzl/fyp* file path. This hierarchical directory ensure that all necessary modules, scripts, and data are arranged clearly.

The system consists of seven main python files and three subfolders:

- Main.py -> The main execution script file that integrate motion detection, image capture, facial recognition, lock/unlock, alert features as well as blynk.
- Camera.py -> Enables the Raspberry Pi camera module to take pictures when motion is detected.

- Frecog.py -> Run facial recognition function where it compares taken pictures to a list of approved face encoded to the system.
- Alert.py -> Control buzzer to trigger alert sound when an unauthorized face or unusual loitering is detected.
- Lock.py -> Control servo motor to lock or unlock the door upon response to the Raspberry Pi upon recognizing authorized user.
- User.py -> Consists of script that records and registers new authorized user faces.
- Images/ -> Contain lists of photos taken during motion detection events for processing and documentation.
- Authorized_access/ -> Contain lists of authorized access faces.
- Unauthorized_access/ -> Contain lists of unauthorized access faces.
- Qr/ -> Contain lists of generated QR code images.
- Used_qr.txt -> Contain guests ID lists.

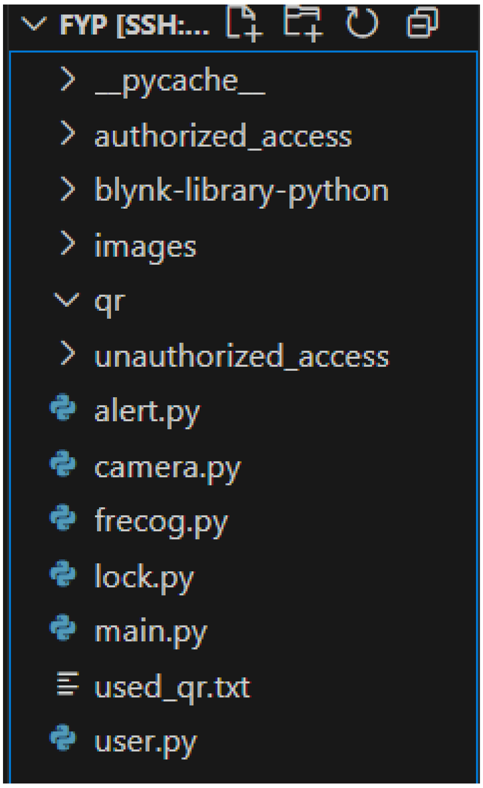


Figure 4.4: Lists of files and folder under FYP directory

Figure 4.4 shows the lists of files and folders in the system. The system debugging, testing, feature implementation and expansion are made easier by this methodical approach that facilitates modular development.

4.3 Proposed AIoT Smart Door Lock System Implementation

This section will provide the overview of the proposed system through pictures and result collected as its data.

4.3.1 System Flow Overview

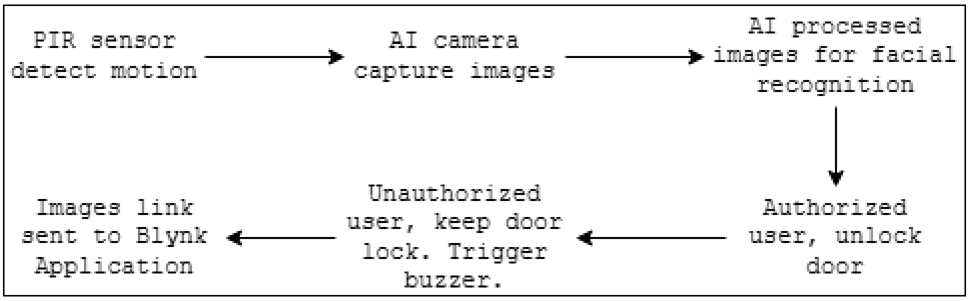


Figure 4.5: System flow overview

From figure 4.5 above, the system flow overview is shown. The system will sense motion using PIR motion sensor and then activate AI camera which is the Raspberry Pi camera module V2 to capture images and then process it for facial recognition to determine the next step which to unlock the door or leave it locked. However, if the image processed is identified as not authorised then it will be accompanied by buzzer sound to alert user. After that, the image link will be sent to Blynk application for data collection.

4.3.2 Motion Detection and Trigger Flow

The AIoT Smart Door Lock system begins operating through the main.py script. A Passive Infrared (PIR) motion sensor attached to GPIO pin 4 is used to continuously track movement. The system initiates the primary workflow—image capture, user verification, and the proper response—when motion is detected.

Motion detection code snippet:

```
if GPIO.input(PIR_PIN):

        update_activity_log("Motion detected!")



        img_path = image()

        print("Analyzing image...")

        is_authorized = face(img_path)



        if is_authorized:

            unlock(with_autolock=True)

            update_activity_log("Door unlocked via face/QR")

        else:

            locked()

            full_alert("Unauthorized access detected!", img_path)

            update_activity_log("Unauthorized access attempt")
```

The Raspberry Pi turns on the camera to take a still picture as soon as it detects motion. The facial recognition module (frecog.py) is then used to analyse this image. The servo motor is activated to unlock the door if a registered user is recognized. Then the alert module, which comprises a buzzer sound and a Telegram notice with the taken image, is activated if the system denies access. Every 0.2 seconds, the script checks for motion in a low-latency, real-time loop. Through GPIO, it provides appropriate hardware cleanliness. The script is dependable and hardware-safe given that cleanup() is called upon exit.

The code snippet above demonstrates the authentication-based conditional branch of the system response. The if face() function with the img_path to check if the user is authorized. If it is true, the system will recognize the face as authorized and proceed to unlock the door via unlock(). Otherwise, if it false, it will detect it as no face detected or did not recognize the face, keep the door locked and send alert through full_alert() function that will trigger t_alert() inside alert.py script to uploads the captured image and send alert notification to Telegram.

4.3.3 Camera Module

The camera module is triggered to take a high-resolution picture of the object that is in front of the door when motion is detected. To ensure that every captured image is individually identifiable and appropriately arranged for further processing or reference, this image is then automatically saved in the images/ folder with a timestamp in the filename. The images saved in the folder is for data collection and is accessible using the Blynk application.

Camera.py code snippet:

```
def  image():

    timestamp = time.strftime("%Y%m%d-%H%M%S")

    img_path = f"/home/srmzl/fyp/images/{timestamp}.jpg
```

4.3.4 Face Recognition

The facial recognition procedure is managed by the frecog.py script. First, it will load images from user.py and then image that is captured to convert the facial feature data to compare to the set authorized image in the data. The code makes use of the pyzbar library to read temporary QR codes created for guest access and face_recognition library to identify known users. The initial step the system does when it receives an image from the camera module attempts facial recognition. The system uses a tolerance threshold to compare the new image with the pre-encoded face data from authorized individuals that is loaded at runtime then access is given to the user if a match is found.

The code snippet below shows the system looks for QR data if no match is found or if there is nonface in the picture for example, a visitor holding up a QR code. Access is allowed if an authorized guest QR code is found and has never been used and the QR token is invalidated to stop reuse. Both secure, temporary guest access and permanent user access are made possible by this dual-authentication approach.

Frecog.py code snippet:

```
if not encodings:

    print("[FR] No face found.")

    if scan_qr(img_path):

        print("[ACCESS] Guest access granted via QR")

        return True
```

4.3.5 Lock Mechanism

The servo motor that controls the door lock bolt is controlled by lock.py script based on the face recognition results. The servo motor spins to 90°, unlocking the door if an authorized user is spotted. Otherwise, the servo motor returns to 0° state in the event of an unapproved image or no match found, ensuring that the door is firmly locked.

The servo motor is connected to GPIO pin 18 of the Raspberry Pi and is operated using Pulse Width Modulation (PWM) signals. The unlock () function is activated upon identifying an authorized user. This feature unlocks the bolt by rotating the servo to about 90°. To make sure the door isn't accidentally left open, the system automatically activates the re-lock mechanism after a 30-second delay. By removing the need of manual re-locking, this automation improves convenience and security. Likewise, when locking is required, either after a timeout or via a manual lock command from the application—the locked () method makes sure the bolt is firmly engaged by rotating the servo to 0°.

Lock.py code snippet:

```
def unlock():

    setup()

    print("Unlocking door...")

    servo.ChangeDutyCycle(7.5)  # Unlock position 90 degree

    time.sleep(1)

    print("30 seconds before auto-lock...")

    time.sleep(30)
```

```
    print("Auto-locking door...")

    servo.ChangeDutyCycle(2.5)  # Lock position 0 degree
```

4.3.6 Alert System

The alert.py script is intended to serve as an auditory warning system by turning on the system's piezo buzzer. When activated, it alerts anyone in the vicinity of possible unwanted access by blaring the buzzer five times quickly. This alert feature is triggered when motion is detected for prolonged amount of time without successful authentication, suggesting potential suspicious activities, or when an unidentified image is detected.

Alert.py code snippet:

```
def full_alert(message="Intruder detected!", image_path=None):

    t_alert()

send_telegram_alert(message, image_path)

.

.

def t_alert():

    for _ in range(5):

        beep(0.3, 1000)
```

The code handles both local and remote alert mechanisms and improves the system’s reaction to unauthorized access attempts. The function t_alert () uses pulse width modulation (PWM) to activate the passive buzzer attached to GPIO pin 16 when it is activated. The script notifies homeowners remotely by integrating with the Telegram messaging app in addition to the local alarm. The send_telegram_alert () function sends the image link to the Telegram message after uploading the unauthorized user's captured image to the imgbb image hosting service. A secure bot token is used to send the alert message to a specified conversation. This guarantees that security events are reported in real time, even when the user is not at home.

Both the remote Telegram alert process and the local buzzer are contained within the full_alert () function. Providing layered notice and logging of any possible intrusion, it serves as the primary alert trigger. In order to increase system performance and remove needless delays, the script incorporates a connectivity check using Google's DNS server to prevent sending alerts without an internet connection.

4.3.7 Mobile App Access via Blynk

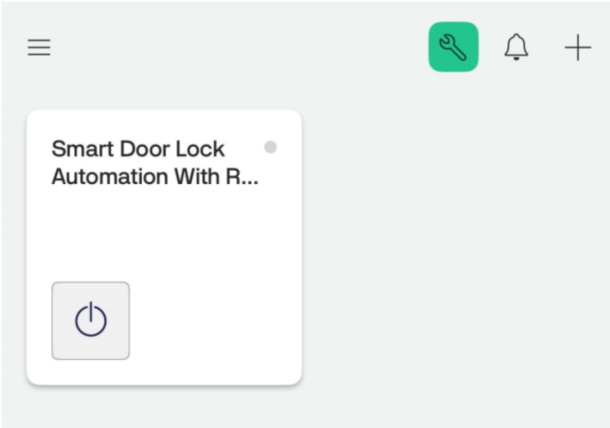


Figure 4.6: Blynk interface.

The smart door lock system can be controlled and monitored through Blynk application using blynk.py script that controls the communication between Raspberry Pi and the Blynk mobile application. Figure 4.6 above is the option shown in the mobile application to choose the system. Upon clicking on the Smart Door Lock Automation With Raspberry Pi option, the interface as in Figure 4.7 is shown indicating all features provided to the user.

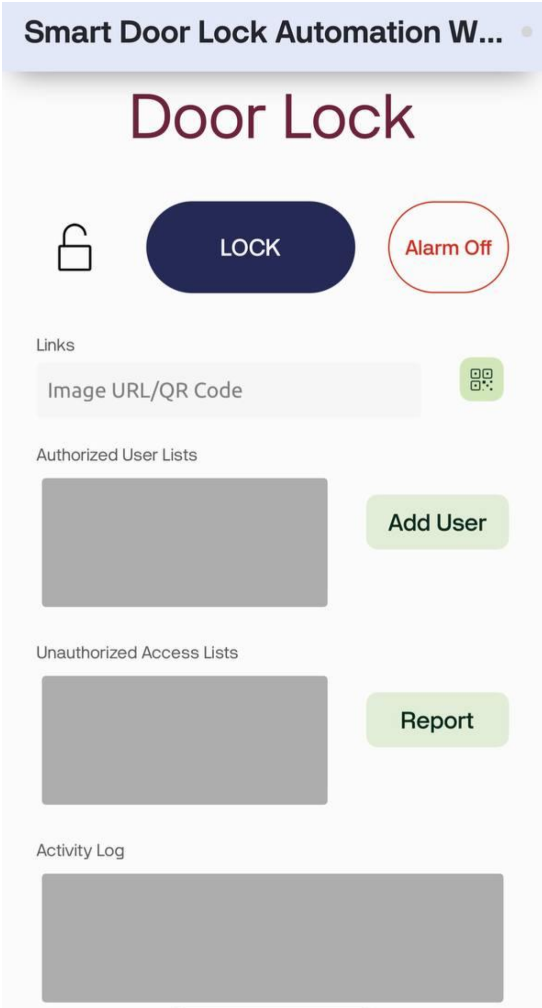


Figure 4.7: User interface on Blynk.

In the figure above, the system interface is design in a compact way. Allowing user to see every event and features from the same page. The lock/unlock button is placed highest among other features due to its main function of remote access to the system which is unlocking and locking the door. There is a small button to generate a QR code and a text space next to it that sends the generated QR code image link. The application also provide user with a set of lists where it will display authorized user username registered in the system, unauthorized access attempt link with a timestamp and an activity log that records every interaction made with the system. Aside from that, it also provides add user button and alarm button for user to add a new registered user to the system and manually triggering the alarm. Table 4.3 below shows the virtual pin assigned for each feature.

Table 4.3: Blynk Virtual Pin Mapping.

Virtual Pin	**Reason**
V0	Lock or unlock door button
V1	Manual alarm activation button
V2	Show QR code image link
V3	Add new user button
V4	Display list of authorized users
V5	Display lists of unauthorized access attempts
V6	Show activity log of system interactions
V7	Generate QR code button
V8	Generate report in PDF

The blynk.py script facilitates remote control and system interaction via the Blynk mobile application. A number of operations, including door control, alarm triggering, user registration, QR code generation, and report export, are associated with key virtual pins (V0 to V8) as seen in Table 4.3 above.

Blynk function code snippet:

```
@blynk.on("V0")

def blynk_lock_control(value):

    if int(value[0]) == 1:

        locked()

    else:

        unlock()
```

This part of the code handles door locking and unlocking mechanism through the Blynk mobile application. With this code, users can access the system remotely via their smartphone.

```
@blynk.on("V3")

def register_new_user(value):

    if int(value[0]) == 1:

        success = capture_image()
```

Users are able to register a new authorized user straight from the mobile app through this part. When it is activated, the new user's face is captured and stored in the system.

```
@blynk.on("V7")

def generate_qr(value):

    ...

    img = qrcode.make(qr_data)

    img.save(qr_path)
```

def generate_qr creates a unique QR code that visitors can scan and upload it to a website that hosts images. One-time guest admission is then made possible by sharing the link via the app.

```
@blynk.on("V8")

def generate_report_trigger(value):

    ...

    bot.send_document(chat_id=CHAT_ID, document=f)
```

The "Generate Report" button in the app is connected to “V8” datastream causes the system to create a PDF with recent activity logs, authorized users, and unauthorized attempts. The bot.send_document(chat_id=CHAT_ID, document=f) will send the report to Telegram application directly for user to access.

4.3.8 Authorized Face Registered

To register or add new user to the database as the authorized user, user.py script is used to take picture using Raspberry Pi camera. The captured image is saved to the database for upcoming facial recognition by saving it to the authorized_access/ folder. For the first authorized user, the system currently utilizes an image saved as sara.jpg as shown in Figure 4.8. For future additions of authorized users will be eventually saved as user1.jpg, user2.jpg, and so on.

Figure 4.8: Authorized user data, sara.jpg.

By capturing a picture with the Raspberry Pi camera and storing it in the authorized_access/ directory, the user.py script makes it possible for new authorized users to register their faces to the system. The capture_image () function takes and saves a high-resolution picture using the libcamera-still command, and next the get_next_filename () function automatically increases filenames to avoid duplication. After that, the newly saved image is encoded and included in the dataset for facial recognition.

User.py code snippet:

```
def get_next_filename(folder):

    os.makedirs(folder, exist_ok=True)

    count = 1

    while True:

        filename = f"user{count}.jpg"

        if not os.path.exists(os.path.join(folder, filename)):

            return filename

        count += 1

.

.

def capture_image():

    ...

    subprocess.run([

        "libcamera-still",

        "-o", filepath,

        "--nopreview",

        "--timeout", "1000",

        "--width", "3280",

        "--height", "2464",
```

```
    "--brightness", "0.2",
    "--contrast", "1.2"
], check=True)
```

The code snippet demonstrates that each newly registered authorized user is given a unique file name such as user1.jpg, user2.jpg, and etc dynamically by the function to make sure that every registered user face is stored with a unique name and checks the authorized_access folder to prevent overwriting the same filename as the existing users. The capture_image () segments in the code captures a high-resolution picture using the Raspberry Pi camera module. The picture is used when doing the face recognition process for upcoming authentication and is kept in the approved user directory.

4.4 System Demonstration

In this section, the system is demonstrated through screen capture and images taken throughout development process.

4.4.1 Motion Detection and Image Capture

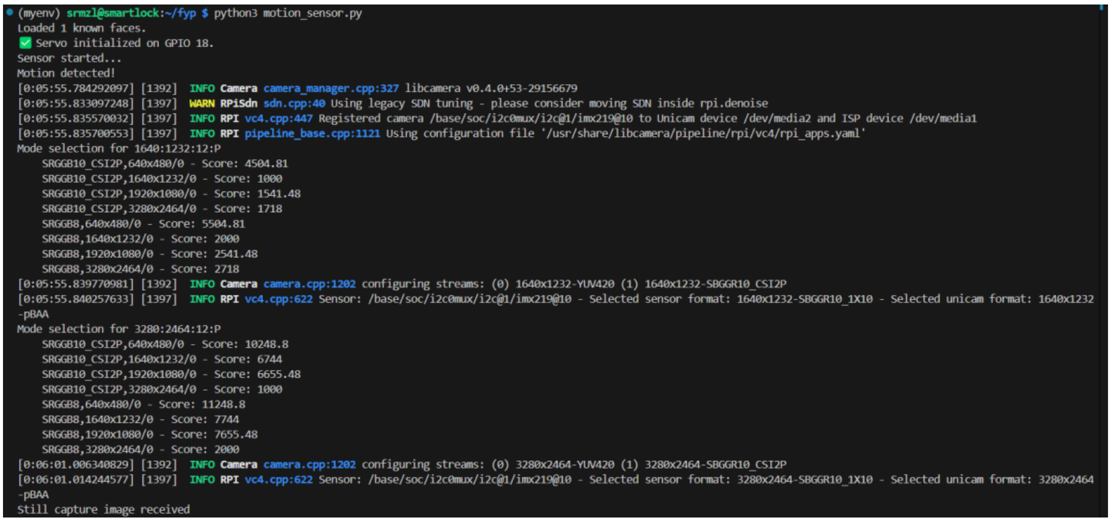


Figure 4.9: motion_sensor.py activates sensor, camera, and saving captured image to images/ folder.

Figure 4.9 presents the terminal output generated when the main.py script is executed, recording the initial startup and motion-trigger sequence that forms the basis of the control process for the system. The log starts with system initialization, during which the script loads the necessary modules and verifies that the database has successfully retrieved one approved user profile. The system simultaneously initializes important hardware parts, most notably the servo motor that operates the door lock.

The system then starts the Passive Infrared (PIR) motion sensor and records the status message "Sensor started...," indicating that it is now actively monitoring the activity in the surrounding area. The console records "Motion detected!" when motion is detected, which instantly activates the Raspberry Pi camera module through the libcamera interface. In order to ensure a high-quality image capture, the log verifies that the camera was successfully configured with optimal sensor mode and resolution settings (e.g., 1640x1232).

The message "Still capture image received," confirms that the picture was successfully shot and saved in the images/directory, appears at the end of the operation. Subsequent authentication procedures, such as facial recognition or QR code verification, use this image as input. This result, combined together, confirms that the system's initial motion detection and image acquisition process is functioning as planned, providing a smooth transition between sensor input and camera capture and serving as a crucial initial step in the security protocol of the smart door lock.

4.4.2 Authorized Face Recognition

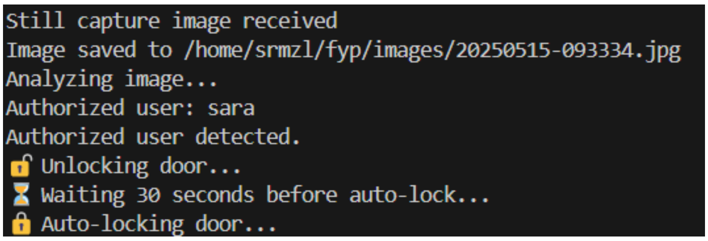


Figure 4.10: frecog.py analyzing and comparing captured image with data in authorized_user/folder; door unlock, authorized user detected.

Figure 4.10 illustrates the expected result of the system upon successful authentication based on facial recognition. The entire chain of events, from picture analysis to access permission and post-entry security enforcement, is captured in this terminal output. The procedure starts when a taken image is sent to the facial recognition script, which logs "Analyzing image…" to show that the system has started processing visual input for identification.

After successfully recognizing the face in the picture and comparing it to a previously registered profile in its database, the system verifies that the person is an authorized user. This is clearly logged with the message "Authorized user: sara," proving that its AI-powered recognition module accurately identified the user. As shown by the command "Unlocking door…," the system unlocks the door after this verification, turning on the servo motor to provide physical access.

The system has an automatic re-lock process in place to ensure ongoing security. The log "Waiting 30 seconds before auto-lock…" shows that a 30-second countdown timer is started when access is granted. The message "Auto-locking door…" confirms that the system re-engages the locking mechanism when the countdown has finished. By preventing the door from accidentally remaining unlocked, this integrated time function minimizes the possibility of unwanted entry once access has been approved.

This output demonstrates the facial recognition module's intelligence and dependability by demonstrating that it can accurately identify a registered person, initiate the appropriate hardware steps, and automatically restore security conditions—all without the need for human intervention. It displays the smooth transition between physical access control and authentication logic and shows the essential features of the smart door lock system.

4.4.2.1 Types of Authoried Access Condition

The system was tested with authorized users in multiple scenarios to validate its ability to function in real-world situations. These include not facing the camera straight ahead, turning the head, wearing cap or covering the head, or wearing spectacles. Other than that, the system also allow access to an authorized QR code generated from the system. Figure 4.11, 4.12, and 4.13 shows the example of authorized access images captured by the camera. This shows the face recognition module's ability to adapt to small changes in appearance or orientation, which is important for assuring a seamless and intuitive user experience without requiring constant, perfect alignment.

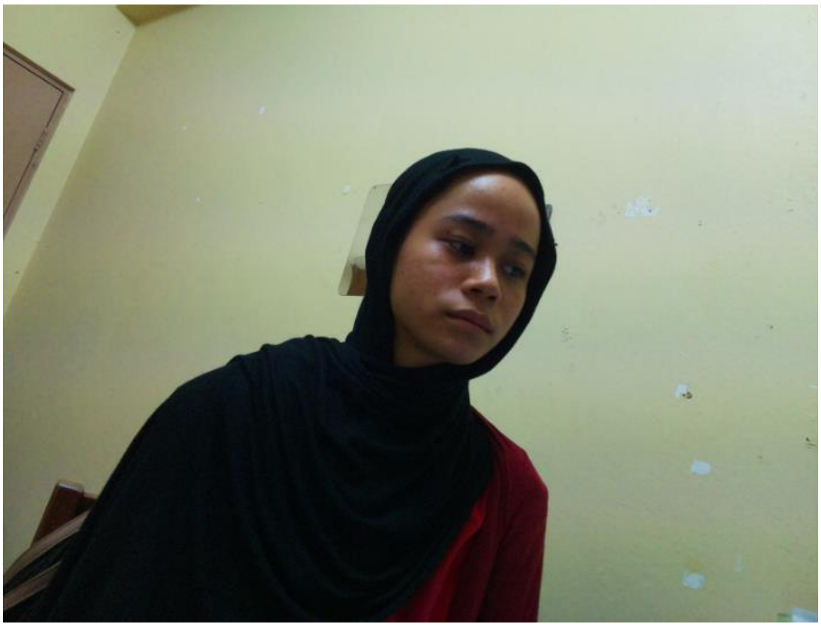


Figure 4.11: Face positioned at different angle, eyes not looking at camera.

Figure 4.12: User wearing cap, spectacles and eyes not looking at camera.

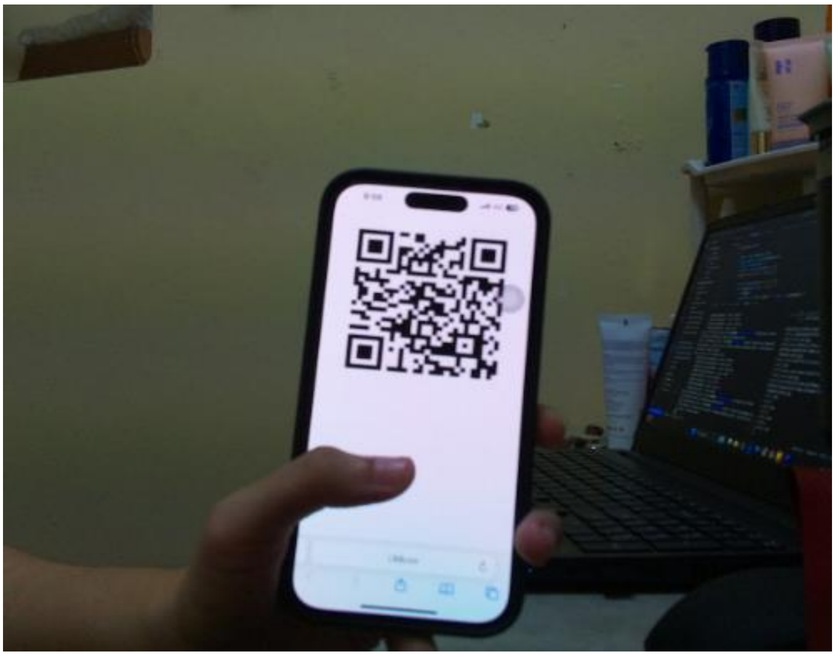


Figure 4.13: QR code access.

4.4.3 Unauthorized Face Alert

The system's automated reaction to an unauthorized access attempt is illustrated in Figure 4.14, demonstrating its adherence to the "default-deny" security approach. In this case, the motion sensor picks up movement, which causes the camera to turn on and take a picture. The facial recognition module then examines the image, comparing the identified face to the approved user database kept in the authorized_access/ directory. The phrase "No face found" appears at the end of the analysis, indicating that the person does not match any recorded profiles.

The message displaying "Unauthorized user!" indicates that the system has immediately classified the access attempt as unauthorized after this authentication failure. After then, a multi-step security protocol is carried out. In order to ensure that the incident can be evaluated at a later time, the system first logs visual evidence by saving the taken image to the unauthorized_access/ directory as shown in Figure 4.16. The log "Locking door…" confirms that the servo motor is activated to enforce a locked condition in order to physically secure the property. At the same time, an urgent warning “Potential intruder detected!”—and an on-site sound alert are triggered. When the message "Please check the camera feed." appears, the user or system administrator is prompted to look into the situation.

This result shows how responsive and dependable the defense mechanisms of the AIoT Smart Door Lock are. The system successfully blocks unwanted access while retaining complete insight into the situation by blocking entry by default, documenting evidence, and sending out real-time alarms. When proactive and autonomous security is a top concern, this feature is crucial for both home and small office settings.

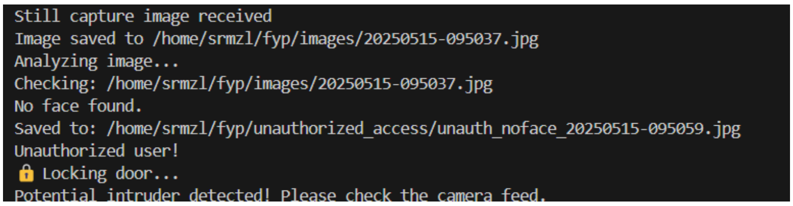


Figure 24.14: frecog.py analyzing and comparing captured image with data in authorized_user/folder; door remain locked, unauthorized user detected.

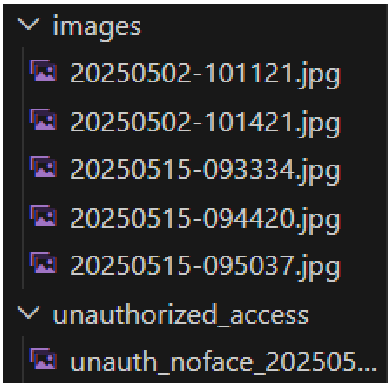


Figure 4.15: Unauthorized user image saved to unauthorized_access/ folder.

4.4.3.1 Types of Unauthorized Access Condition

To confirm the recognition system's consistency, unauthorized access conditions were evaluated as well. These include users who are not in the database, wear masks, cover a portion of their faces and face not looking at camera. These access attempts were promptly denied by the system, which also sent out the required alerts. These examples demonstrate the system's capacity to work in a variety of real-world situations where unauthorized access attempts may occur. Figure 4.16 and 4.17 are the examples of the images captured for an unauthorized access.

Figure 4.16: User wear mask, face positioned at different angle, and eyes not looking at camera.

Figure 4.17: User wear mask, spectacles, and face not looking at camera (no face detected).

4.4.4 Blynk Mobile App Control

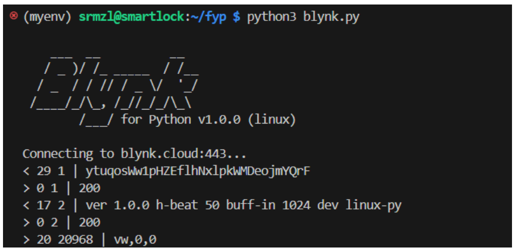
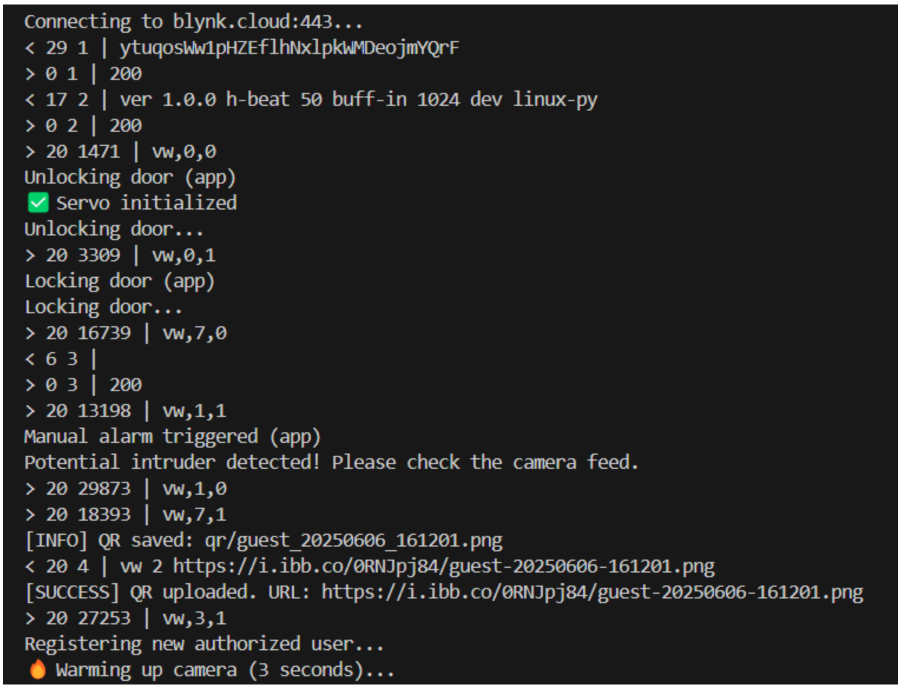


Figure 4.18: Blynk application activated, data is recorded when user access the application.

This section focuses on the system integration with Blynk mobile application for remote control and monitoring system. Figure 4.18 above shows the terminal console activating the blynk cloud using command python3 blynk.py. The system connects to the Blynk cloud by executing the python3 blynk.py command, which makes it achievable to control the smart lock from a mobile device. The terminal output sets up the communication channel and verifies a successful connection to the Blynk server. This connection enhances the smart door lock system's real-time responsiveness and provides simplicity by allowing user to submit actions from their smartphone, such as unlocking or locking the door. It also connects the user interface with hardware, which is crucial for smart home applications.



```
Connecting to blynk.cloud:443...
< 29 1 | ytuqosWw1pHZEflhNxlpkWMDeojmYQrF
> 0 1 | 200
< 17 2 | ver 1.0.0 h-beat 50 buff-in 1024 dev linux-py
> 0 2 | 200
> 20 1471 | vw,0,0
Unlocking door (app)
✅ Servo initialized
Unlocking door...
> 20 3309 | vw,0,1
Locking door (app)
Locking door...
> 20 16739 | vw,7,0
< 6 3 |
> 0 3 | 200
> 20 13198 | vw,1,1
Manual alarm triggered (app)
Potential intruder detected! Please check the camera feed.
> 20 29873 | vw,1,0
> 20 18393 | vw,7,1
[INFO] QR saved: qr/guest_20250606_161201.png
< 20 4 | vw 2 https://i.ibb.co/0RNJpj84/guest-20250606-161201.png
[SUCCESS] QR uploaded. URL: https://i.ibb.co/0RNJpj84/guest-20250606-161201.png
> 20 27253 | vw,3,1
Registering new authorized user...
🔥 Warming up camera (3 seconds)...
```

Figure 4.19: Multiple system status from Blynk application.

A detailed log of the AIoT Smart Door Lock system's ongoing session with the Blynk cloud platform is shown in Figure 4.19, demonstrating the range of real-time communication and administrative controls made possible by the Blynk mobile application. The log shows that the Raspberry Pi hardware is successfully synchronized with the remote-control interface after a successful handshake and secure connection to the cloud server (blynk. cloud:443). The base for smooth command execution between the user and the system is laid by this dependable link.

The image shows a series of remote commands that the user can submit through the Blynk app after the initial handshake. These include specific entries like "Locking door (app)" and "Unlocking door (app)," which make it clearly evident that mobile instructions, not local triggers like motion sensors, were used to change the door's physical state. The "Manual alarm triggered (app)" log, which immediately creates a high-priority alert—"Potential intruder detected!"—also demonstrates the system's capabilities for manual intervention for security problems. This allows the app to remotely activate the buzzer for emergency deterrent.

The log shows the advanced administrative tools that users can access in addition to basic access control. For instance, "[SUCCESS] QR uploaded" is displayed when a guest QR code is generated and remotely uploaded. This is followed by a response message that confirms the successful transmission of the public link to the QR image back to the app for sharing. The last entries demonstrate that even user onboarding may be controlled remotely via the application by capturing the initial step of a new authorized user registration sequence, which includes camera warm-up operations.

The image highlights a fully bi-directional communication infrastructure and demonstrates the system's solid interoperability with the Blynk IoT platform. Users can actively operate door mechanisms, set off alarms, create temporary credentials, and manage user profiles from any location in addition to being able to monitor and receive notifications. This turns the smart lock into a complete, mobile-accessible home security system rather than just an automation tool.

4.4.4.1 Adding New Authorized User Using Blynk Application

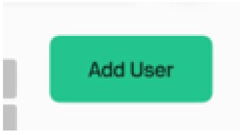


Figure 4.20: Add User button in Blynk application.

This functionality shows how the system enables the owner to remotely add a new authorized user via the Blynk application by clicking “Add User” button as shown in Figure 4.20. The system stores the still image (e.g., user1.jpg) when it is captured and verified, and it indicates that the new user has been added by showing "New user added." in the terminal console as seen in Figure 4.21. This step is necessary to ensure that the system can develop and adjust by learning to identify new users who are allowed on the property. It also improves the scalability and flexibility of the system.

```
Still capture image received
Image saved as user1.jpg
New user added.
```

Figure 4.21: Adding new user data to the system using Blynk application.

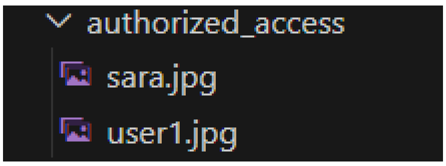


Figure 4.22: New user added under authorized_access/ folder.

Figure 4.22 shows that the still captured image is saved authorized_access/ folder which is a folder that saves all authorized user images in the system. As shown, the folder contains both sara.jpg and user1.jpg, confirming that the data structure and storage mechanism are operating as intended. Additionally, it verifies that the picture management subsystem keeps a well-organized record for manual reference or facial recognition.

4.4.4.2 Access System Using QR Code

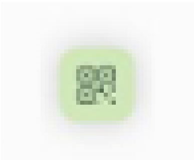


Figure 4.23: QR Code button in Blynk application.

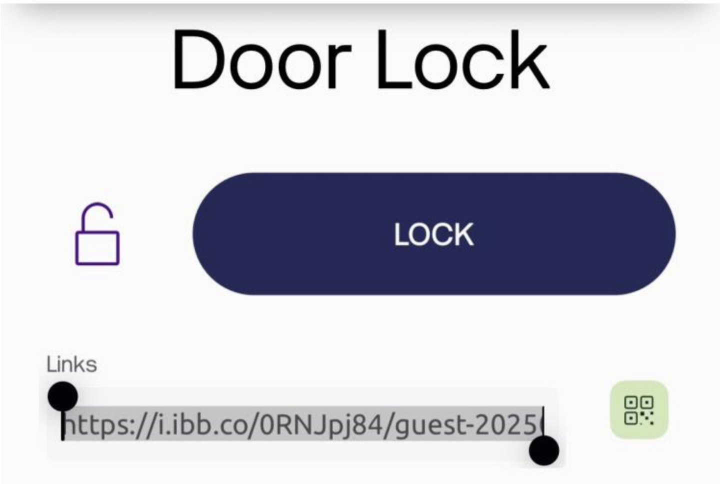


Figure 4.24: Generated QR link in mobile application.

Figure 4.25: Generated QR code send to i.ibb.co (imgbb).

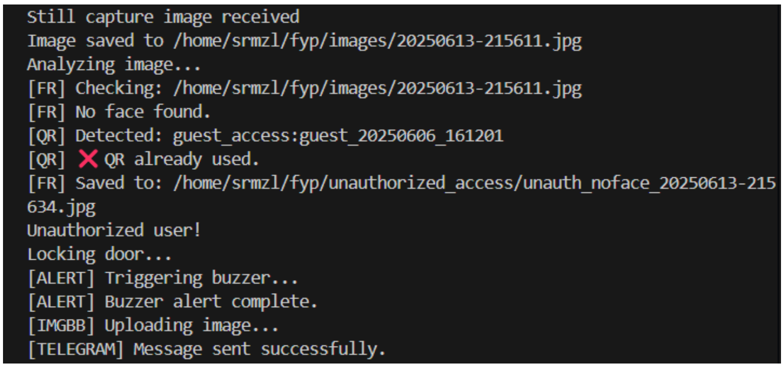
Figure 4.24 and 4.25 illustrates the system's support for guest access through QR codes. A unique QR code is generated upon request by clicking assigned button in Figure 4.23, and uploaded to a cloud platform (i.ibb.co), which is afterwards accessible via a mobile application. This method increases the system's adaptability for visitors or delivery by permitting temporary or one-time access without requiring permanent registration.

```
Still capture image received
Image saved to /home/srmzl/fyp/images/20250606-171837.jpg
Analyzing image...
[FR] Checking: /home/srmzl/fyp/images/20250606-171837.jpg
[FR] No face found.
[QR] Detected: guest_access:guest_20250606_161201
[QR] Valid guest QR detected
[ACCESS] Guest access granted via QR
Authorized user detected.
```

Figure 4.26: System output for using QR code to unlock the door.

This output in Figure 4.26, demonstrates how the system recognizes an authorized QR code, verifies the visitor's credentials, and allows access to the system. The system starts by executing its standard multi-factor authentication procedure. When it first tries facial recognition, the result "[FR] No face found" is returned, meaning that the person is not listed in the database of permanently registered users. After realizing this, the system moves on to its second check, where it finds and decodes a QR code that has been presented and logs it as "[QR] Detected: guest_access…."

The message "[QR] Valid guest QR detected," which indicates that the code is genuine, active, and unused, is then displayed after the system compares the QR code with its internal records. The system logs "[ACCESS] Guest access granted via QR" after the verification is completed, and then it updates the status to indicate that "Authorized user detected." This means that when the QR code was successfully validated, the system considers the visitor to be temporarily authorized. The efficiency and dependability of the QR-based guest access feature are validated by this outcome. It draws attention to the system's multi-tiered authentication capability, which handles temporary access credentials with the same amount of scrutiny and logging as permanent user entries.



```
Still capture image received
Image saved to /home/srmzl/fyp/images/20250613-215611.jpg
Analyzing image...
[FR] Checking: /home/srmzl/fyp/images/20250613-215611.jpg
[FR] No face found.
[QR] Detected: guest_access:guest_20250606_161201
[QR] ❌ QR already used.
[FR] Saved to: /home/srmzl/fyp/unauthorized_access/unauth_noface_20250613-215
634.jpg
Unauthorized user!
Locking door...
[ALERT] Triggering buzzer...
[ALERT] Buzzer alert complete.
[IMGBB] Uploading image...
[TELEGRAM] Message sent successfully.
```

Figure 4.27: System output if use used QR code to unlock the door.

On the other hand, the system terminal output in response to a used QR code is shown in Figure 4.27 above. The log shows how well the systems multi-layered security protocol handles token replay attacks by capturing the entire sequence of the protocol. Multiple defensive measures are included in the response and the system reports the incident by saving the taken image to the unauthorized_access directory, secures the door by performing a lock command, and sounds the buzzer as an audio warning. Furthermore, the system instantly sends a Telegram notice to the homeowner or system administrator in real time after uploading the image to a remote hosting provider (IMGBB). This all-encompassing security response assures remote awareness of potential breaches in addition to local defense. The system's capacity to identify, stop, and report unauthorized attempts using previously verified credentials is highlighted in the figure, which also successfully illustrates its defense against QR code use. By providing guests, employees, or temporary users with secure access while maintaining the integrity of the main facial recognition security infrastructure, the QR approach increases the flexibility of the system.

4.4.4.3 Authorized User Lists

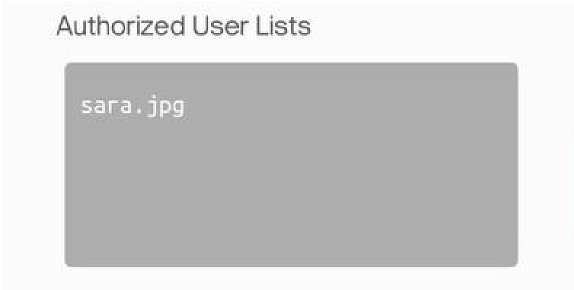


Figure 4.28: Terminal widget in Blynk application showing authorized user lists.

All individuals who have previously been registered and given access are listed in this feature as shown in Figure 4.28. User can use it as a quick reference to confirm who is currently allowed in the system. As seen in the example, sara.jpg is already in the system, ensuring that users are accurately listed and accessible for administrative and security reasons.

4.4.4.4 Unauthorized Access Lists

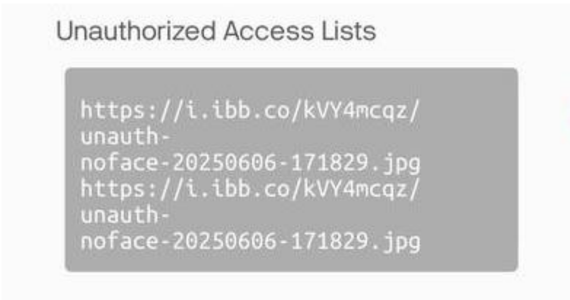


Figure 4.29: Terminal widget in Blynk application showing links of unauthorized access attempt lists.

Figure 4.29 shows all failed access attempts by unidentified users are documented in this section. Unauthorized image URLs (such as attempts without a facial match) are recorded by the system. This functionality, which enables the user to examine questionable activity or intrusions, is essential for security reasons. It can also help in identifying potential security threats and implementing preventive measures.

4.4.4.5 Activity Log Lists

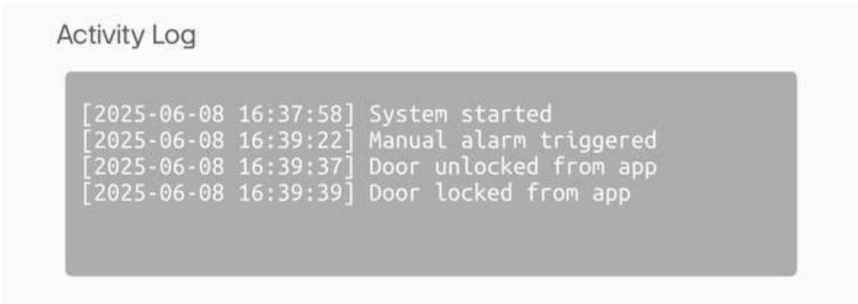
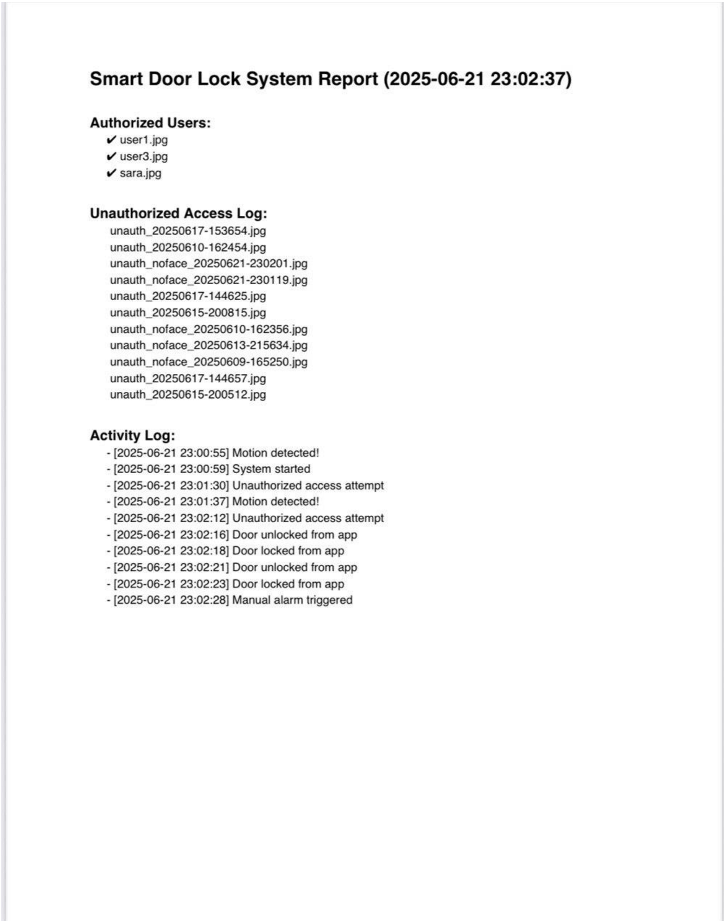


Figure 4.30: Terminal widget in Blynk application showing lists of activity logs when system started.

Figure 4.30 shows all key system events are kept in chronological order in the activity log. Transparency and traceability are provided by entries like "System started," "Manual alarm triggered," and changes in door lock status. This log is particularly helpful for troubleshooting, post-event analysis, and tracking system activity over time. It ensures accountability and operational insight for users.

4.4.4.6 Generate Report from Blynk

Figure 4.31: Report button in Blynk application.



Smart Door Lock System Report (2025-06-21 23:02:37)

Authorized Users:
✔ user1.jpg
✔ user3.jpg
✔ sara.jpg

Unauthorized Access Log:
unauth_20250617-153654.jpg
unauth_20250610-162454.jpg
unauth_noface_20250621-230201.jpg
unauth_noface_20250621-230119.jpg
unauth_20250617-144625.jpg
unauth_20250615-200815.jpg
unauth_noface_20250610-162356.jpg
unauth_noface_20250613-215634.jpg
unauth_noface_20250609-165250.jpg
unauth_20250617-144657.jpg
unauth_20250615-200512.jpg

Activity Log:
- [2025-06-21 23:00:55] Motion detected!
- [2025-06-21 23:00:59] System started
- [2025-06-21 23:01:30] Unauthorized access attempt
- [2025-06-21 23:01:37] Motion detected!
- [2025-06-21 23:02:12] Unauthorized access attempt
- [2025-06-21 23:02:16] Door unlocked from app
- [2025-06-21 23:02:18] Door locked from app
- [2025-06-21 23:02:21] Door unlocked from app
- [2025-06-21 23:02:23] Door locked from app
- [2025-06-21 23:02:28] Manual alarm triggered

Figure 4.32: Report generated from the Blynk application.

A practical feature that enables users to create a report straight from the Blynk mobile application is included into the AIoT Smart Door Lock system. A customized "Report" button as shown in Figure 4.31 interface allows users to access this feature. This reporting feature, which automates the process of evaluating user behaviour and security incidents, is intended to improve system transparency, traceability, and simplicity of monitoring. When the “Report” button is clicked, the system gathers important information in the application and exports it as a PDF document and send it to Telegram application that can be downloaded or shared. There are three main parts included in the report which is a list of authorized users, lists of unauthorized user access image links, and the activity log of the system. Once the PDF document is generated, the data in the application will be deleted to replace with a new one. Figure 4.32 shows the generated PDF documents.

This reporting feature significantly enhances the usability and professional feel of the AIoT Smart Door Lock system. It offers transparency from a security perspective, enabling homeowners to regularly examine system functions and access attempts. Since individual actions may be linked to certain users or times, it also creates accountability for families or shared residences. The report can serve as a formal record of entry logs and intrusion alerts in professional settings, such as home businesses or small offices.

4.4.5 Telegram System Notification

The integration of Telegram as a real-time alarm system for detecting unauthorized access is highlighted in this section. Whenever an unauthorized individual tries to enter the designated area (House), the system immediately alerts the user through a specially designed bot called "MySmartDoorLock.". As shown in Figure 4.33 and 4.34, the bot sends a warning message along with a link and visual snapshot of the unauthorized user. By allowing the owner to remotely monitor security concerns, take prompt action, and maintain situational attention at all times, this feature ensures immediate attention and improves the responsiveness of the system. It greatly enhances the smart security system's overall dependability and user experience.

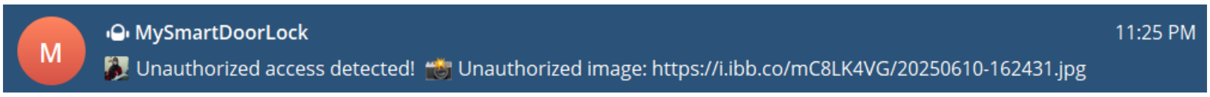


Figure 4.33: System's "MySmartDoorLock" telegram bot.

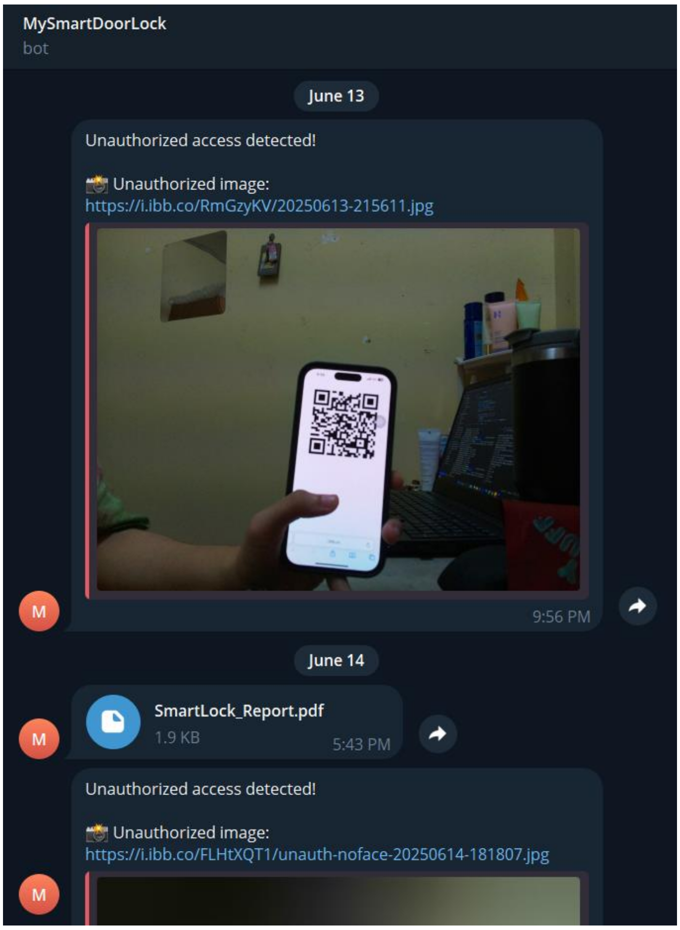


Figure 4.34: Notification alert in "MySmartDoorLock" telegram chat.

4.5 Prototype Design

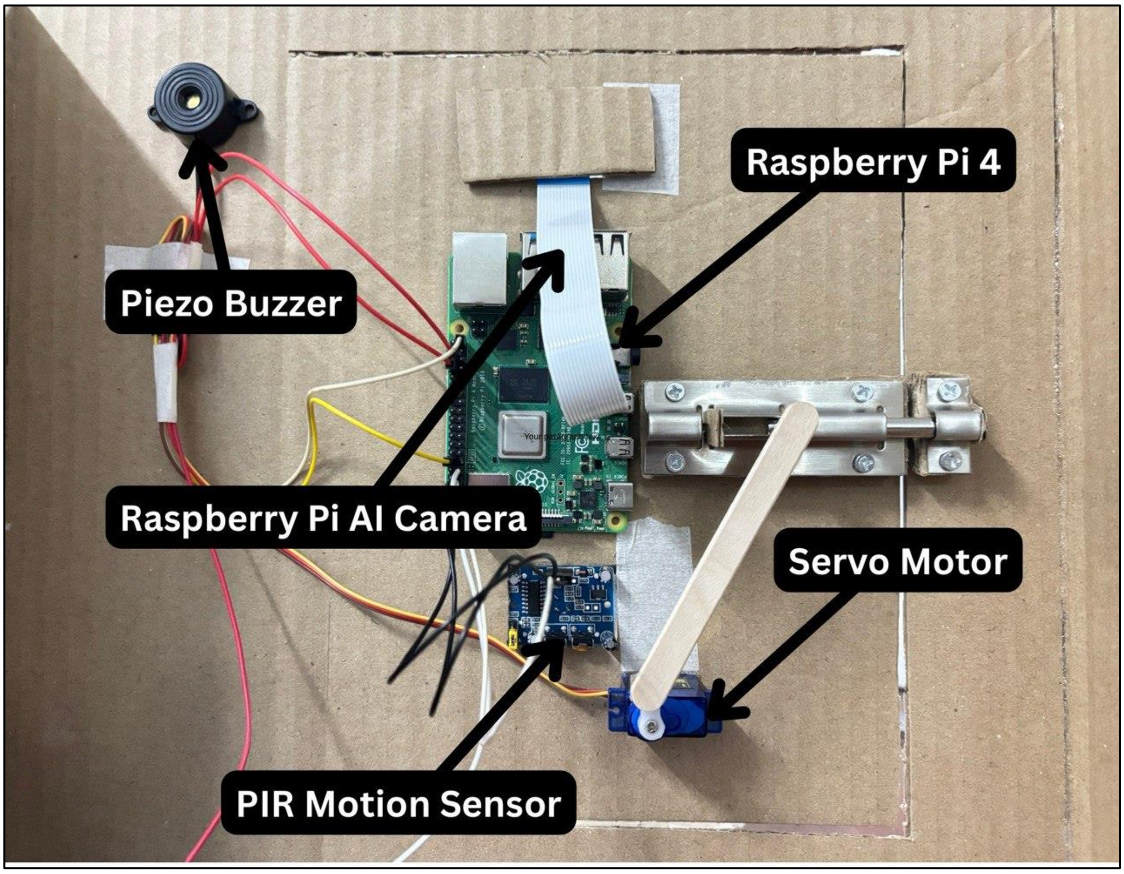


Figure 4.35: Hardware placement on prototype.

As shown in Figure 4.35 and 4.36, a proof-of-concept prototype of the AIoT Smart Door Lock was developed in order to test the system's end-to-end operation in an actual environment. In order to provide a solid foundation for installing and testing all essential hardware components, the prototype was put together on a structured cardboard platform that resembled a section of a door. Full integration testing of the software logic with mechanical and sensory hardware was made possible by this physical architecture, ensuring smooth communication between all modules.

The Raspberry Pi 4, which serves as the system's central processing unit, is at the center of the prototype. The Raspberry Pi, which is positioned in the middle of the prototype, controls all communication, logic control, and decision-making functions. Every peripheral part, such as sensors, actuators, and alert systems, is directly interfaced with the GPIO pins of the Pi. This central location improves structural organization, minimizes possible signal deterioration, and maximizes wiring efficiency.

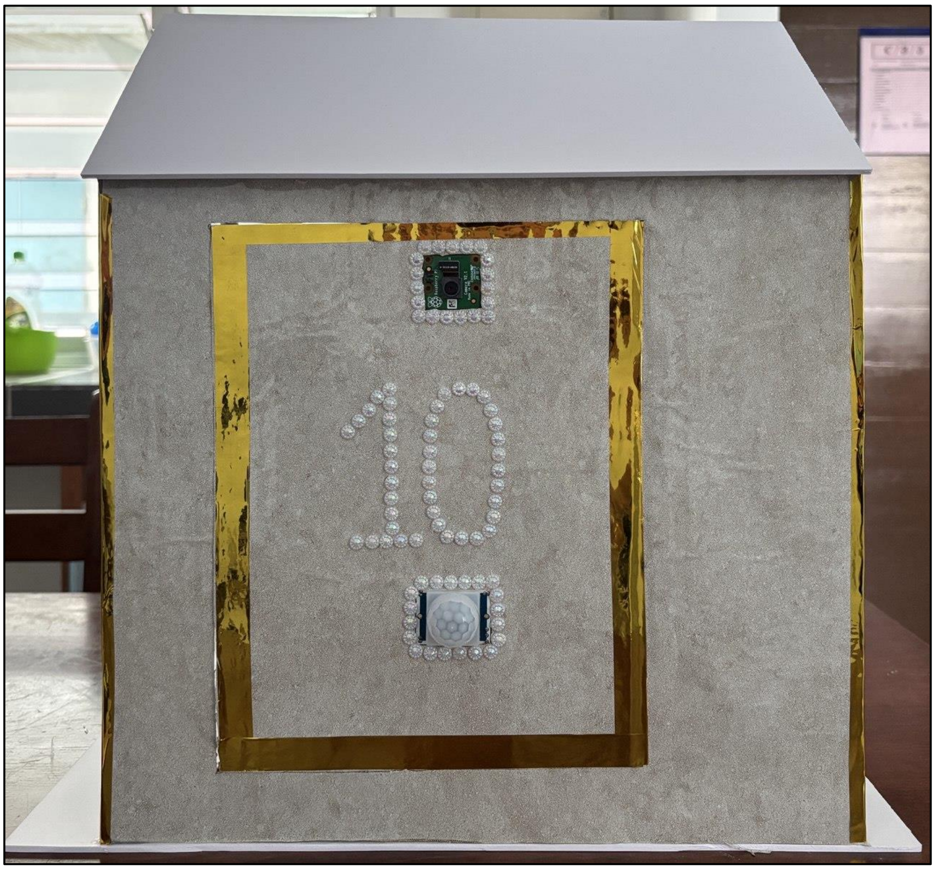


Figure 4.36: Prototype Design.

To efficiently detect approaching motion within its coverage zone, the PIR motion sensor is positioned lower on the board. The sensor alerts the Raspberry Pi to movement by sending an input signal, which causes the AI camera module to turn on. The camera is positioned to keep a direct line of sight toward the user's face and is connected via the Camera Serial Interface (CSI) ribbon connection. The camera is only powered on when motion is detected due to this sequential activation approach, which is purposefully made to be energy-efficient.

A standard sliding bolt lock is used as the locking mechanism, and it is mechanically connected to a servo motor using a specially made lever arm that uses a wooden popsicle stick. The servo rotates to remove the bolt and unlock the door when the Raspberry Pi detects an authorized user and sends a Pulse Width Modulation (PWM) signal to it. The device reinforces physical security by automatically re-engaging the lock after a short delay (30 seconds). With this setup, digital access control is successfully converted into a physical locking mechanism.

A piezoelectric buzzer is positioned in the upper-left corner of the setup to provide instant on-site alerts. Certain circumstances, such unauthorized access attempts or manual alert triggers from the Blynk app, cause this component to become active. The buzzer's loud, pulsating sound is intended to warn people in the vicinity of a security breach and deter intruders. Its high location guarantees the best possible sound dispersion.

In conclusion, the prototype effectively illustrates how every system module is fully integrated both logically and physically. Every element is arranged with care to replicate realistic deployment situations while preserving testability and visibility. The AIoT

Smart Door Lock design's viability is confirmed by this model, which offers a foundation for further development and application in real life.

4.6 Summary of Chapter 4

In Chapter 4, the implementation of the system is thoroughly explored and explained. The AIoT Smart Door Lock prototype whole development was described in this chapter. On the Raspberry Pi, every module including those for motion detection, facial recognition, servo control, alert systems, and mobile app integration was successfully set up, created and tested. In a smart home security scenario, the system demonstrated the ability to successfully execute real-time authentication and alerts. Overall, this chapter validates the prototype's operational success in a smart home environment. In the next chapter, the testing stage will be done and documented in details.

CHAPTER 5: TESTING AND EVALUATION

5.1 Introduction

To ensure the effectiveness, stability, and quality of the proposed AIoT Smart Door Lock system, testing and evaluation are crucial stages in the system development lifecycle. These stages guarantee that the system not only achieves its goals but also operates dependably in a variety of real-world scenarios. This chapter evaluates the ability of the system to provide safe, hands-free access through the use of facial recognition and real-time monitoring through Internet of Things integration.

The testing phase is divided into two primary categories: functional testing and non-functional testing. Verifying essential system functions including motion detection, facial recognition, door locking, and mobile app interactions is the focus of functional testing. These tests include unit testing, system testing and integration testing to ensure every part function both individually and cooperatively. In non-functional testing, the system's reaction time, performance, usability, and dependability are evaluated in a variety of scenarios. User Acceptance Testing (UAT), which collects input from actual users to gauge satisfaction and pinpoint areas for improvement, is also included in this test. In order to provide light on the overall efficacy of the suggested solution, this chapter describes the testing methodologies, tools, scenarios, and results acquired throughout the evaluation process.

5.2 Testing

5.2.1 Functional Testing

The functional testing was conducted manually by observing respondents interact with the system in a controlled setting. User was given a feedback form to provide feedbacks upon finish testing the scenarios. Each feedback and test cases were documented using a checklist and were summarizes in table 5.1 under section 5.1.1.1 below, where the case ID, simulated scenarios, expected outcome and user comment space is given.

5.2.1.1 UAT Scenarios and Test Cases

To ensure that the AIoT Smart Door Lock system performs as intended in real-world setup and environment a series of functional test cases were designed and executed through User Acceptance Testing. Every test case simulates typical situations that might occur in day-to-day operations and focuses on a particular user interaction or system reaction. All essential functionalities, such as motion detection, facial recognition, alarm triggering, mobile app control, and offline capability, are covered in these scenarios.

Table 5.1: UAT test cases and outcomes.

Test Case ID	**Scenario**	**Expected Outcome**	**(√/X)**	**User Comment**
TC01	User **approaches the door,** and the **PIR**	Camera is activated		

	motion sensor detects movement			
TC02	**Authorized user's face** is detected	Door unlocks automatically		
TC03	Unauthorized face detected	System saves captured image and sends alert via Telegram		
TC04	User unlocks the door via Blynk app	Door unlocks remotely		
TC05	User locks the door via Blynk app	Door locks remotely		
TC06	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent		
TC07	User approaches the door but is **wearing a mask or sunglasses**	System still recognizes the user and unlocks the door (if facial recognition is robust)		
TC08	**Multiple faces detected at once** (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks		

TC09	Guest scans valid QR code shown on their phone	System grants access and unlocks the door		
TC10	Guest scans an **expired or invalid** QR code	System denies access and triggers alert		

The purpose of these test cases is to validate whether the system reacts appropriately to both expected and unexpected inputs. For example, the fundamental functions that are directly related to system security includes motion detection (TC01), identifying an authorized user (TC02), and managing unauthorized access (TC03). Other test cases, such (TC05) and (TC06), assess the Blynk mobile app's remote door control capabilities, and (TC09) focuses on how the system responds to a frequent real-world limitation which is no internet connectivity.

More complicated situations such as (TC07) and (TC08) assess the resilience and adaptability of the facial recognition system in less-than-ideal conditions, such as when the user is partially disguised or when numerous users are recognized at the same time. Furthermore, (TC04) tests the ability of the system to identify and react to lingering motion that could be a sign of suspicious activity. All things considered, these test cases perform as a thorough functional testing framework, ensuring that the system not only satisfies technical requirements but also adheres to user expectations related reliability, responsiveness, and usability.

5.2.2 Non-Functional Testing

To assess the overall functionality, responsiveness, and usability of the AIoT Smart Door Lock system, non-functional testing was carried out in addition to functional testing. These tests were designed to evaluate the system's operational quality and user experience, and not only its functionality. Non-functional testing is necessary to assess the prototype's effectiveness, scalability, and readiness for the real world. Users who took part in the UAT were asked to fill out a Likert scale questionnaire as part of this assessment. This feedback provided insight into aspects including perceived security effectiveness, speed of responses, accuracy of the facial recognition system, and user-friendliness.

5.2.2.1 Likert Scale User Feedback

The Likert scale ranged from **1 (Very Disagree)** to **5 (Very Agree)** and Users were asked to rate four important aspects of the system. The questions are shown in the table below, along with a space to note their answers.

Table 5.2: Likert Scale from User Acceptance Testing Form

Questions	**1**	**2**	**3**	**4**	**5**
Is it user-friendly?					
Does the facial recognition function work perfectly?					
Is it fast?					
Do you think it help prevent unauthorized access?					

The purpose of these questions was to gauge the system's perceived efficacy and usefulness from the viewpoint of the user. After compilation, the findings will serve as an outline for analysis and interpretation of the system's overall performance on Section 5.3. Future improvements will be supported by the input received from this section, ensuring that the system's non-functional and functional aspects meet user expectations.

5.3 Analysis

The system passed all revised test cases across a number of user groups, including homeowners, elderly, and visitors, during the UAT phase, demonstrating exceptional performance. Every essential feature, including motion detection, remote control through the Blynk app, QR code-based guest access, facial recognition (including when wearing masks or sunglasses), motion detection, and Telegram alert notifications, worked perfectly. The system reliably generated notifications, recognized authorized users, denied unauthorized users, and appropriately replied to both valid and invalid QR codes. These results verify that the suggested solution satisfies all technical specifications and performs as planned in real-world use cases. Despite the flawless functional performance, the Likert scale survey and open-ended suggestions revealed a discrepancy between perceived accessibility and operational success. System speed was the main issue raised by users. Although all ten participants thought the system was helpful and improved home security, their answers to the question about system speed ranged from positive to unfavourable.

5.3.1 User Acceptance Testing Scenario Results

5.3.1.1 Homeowners

Table 5.3: Homeowners UAT Cumulative Results

Test Case ID	Scenario	Expected Outcome	Results
TC01	User **approaches the door,** and the **PIR motion sensor detects movement**	Camera is activated	**PASS**
TC02	**Authorized user's face** is detected	Door unlocks automatically	**PASS**
TC03	Unauthorized face detected	System saves captured image and sends alert via Telegram	**PASS**
TC04	User unlocks the door via Blynk app	Door unlocks remotely	**PASS**
TC05	User locks the door via Blynk app	Door locks remotely	**PASS**
TC06	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	**PASS**

TC07	User approaches the door but is **wearing a mask or sunglasses**	System still recognizes the user and unlocks the door (if facial recognition is robust)	**PASS**
TC08	**Multiple faces detected at once** (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	**PASS**
TC09	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	**PASS**
TC10	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	**PASS**

5.3.1.2 Elderly

Table 5.4: Elderly UAT Cumulative Results

Test Case ID	Scenario	Expected Outcome	Results
TC01	User **approaches the door,** and the **PIR**	Camera is activated	**PASS**

	motion sensor detects movement		
TC02	**Authorized user's face** is detected	Door unlocks automatically	**PASS**
TC03	Unauthorized face detected	System saves captured image and sends alert via Telegram	**PASS**
TC04	User unlocks the door via Blynk app	Door unlocks remotely	**PASS**
TC05	User locks the door via Blynk app	Door locks remotely	**PASS**
TC06	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	**PASS**
TC07	User approaches the door but is **wearing a mask or sunglasses**	System still recognizes the user and unlocks the door (if facial recognition is robust)	**PASS**
TC08	**Multiple faces detected at once** (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	**PASS**

TC09	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	**PASS**
TC10	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	**PASS**

5.3.1.3 Guests/Visitors

Table 5.5: Guests/Visitors UAT Cumulative Results

Test Case ID	**Scenario**	**Expected Outcome**	**Results**
TC01	User **approaches the door,** and the **PIR motion sensor detects movement**	Camera is activated	**PASS**
TC02	**Authorized user's face** is detected	Door unlocks automatically	**FAIL**
TC03	Unauthorized face detected	System saves captured image and sends alert via Telegram	**PASS**
TC04	User unlocks the door via Blynk app	Door unlocks remotely	**PASS**

TC05	User locks the door via Blynk app	Door locks remotely	**PASS**
TC06	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	**PASS**
TC07	User approaches the door but is **wearing a mask or sunglasses**	System still recognizes the user and unlocks the door (if facial recognition is robust)	**PASS**
TC08	**Multiple faces detected at once** (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	**PASS**
TC09	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	**PASS**
TC10	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	**PASS**

The UAT above is for the Guests/Visitors user group. This group produced highly positive results, with almost all test cases passing, for the 'Guests/Visitors' user group. The 'FAIL' recorded for test case TC02: Authorized user's face is detected is a notable outcome. This conclusion represents a critical confirmation of the security architecture of the system, even though it is indicated as a failure against the predetermined intended outcome.

The system is purposefully made to differentiate between homeowners who are registered permanently and temporary visitors. To ensure a higher degree of long-term security and privacy, visitors are not added to the facial recognition database. As successfully verified in TC09, their access is only controlled by a time-sensitive, one-time QR code system.

As a result, the 'FAIL' in TC02 indicates that the system worked as intended and denied a guest access using facial recognition. This indicates a difference in the "Expected Outcome" column of the test case for this particular user group. Following a revised test plan, a "PASS" would have been achieved if the system had responded correctly to a "Guest/Visitor" by refusing to unlock the door. The fact that both TC09 (valid QR code access) and TC10 (invalid QR code denial) were successfully completed adds more evidence that the guest access feature is reliable, safe, and operates exactly as planned.

5.3.2 Likert Feedback

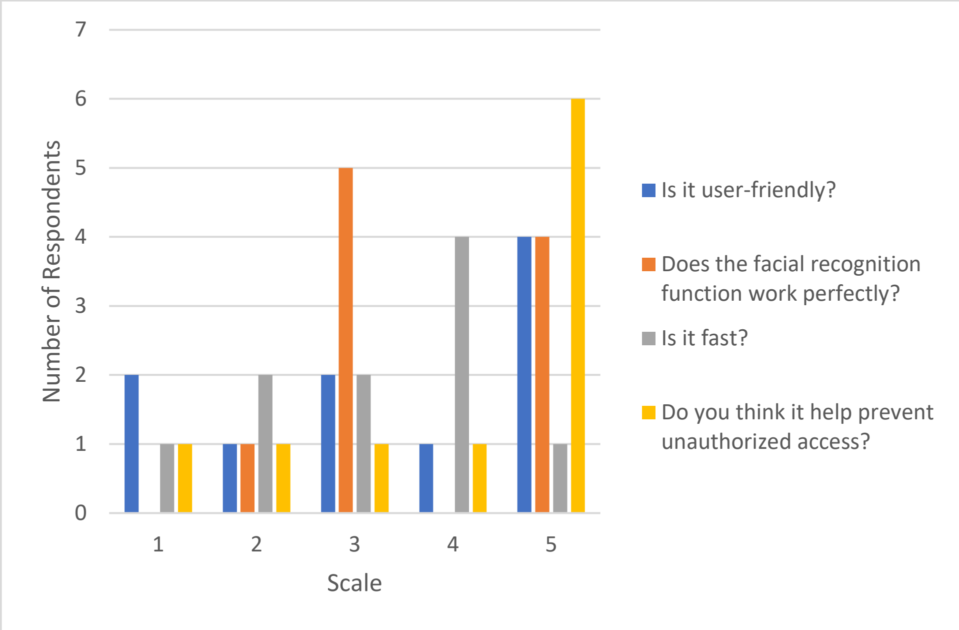


Figure 5.1: Likert scale feedback clustered bar chart.

The AIoT Smart Door Lock system's core features which is user-friendliness, facial recognition accuracy, system speed, and security effectiveness were assessed using a Likert scale in order to collect user opinions outside of functional testing. On a scale of 1 (very disagree) to 5 (very agree), each of the 10 respondents gave these aspects a rating. Figure 5.1 shows a visualization of the results.

The user-friendliness of the system received relatively positive feedbacks. On the Likert scale, the majority of users (4 out of 10) give the highest rating in the scale which is 5, 1 person give it a 4, and 2 persons give a rating of 3 which is they are feeling neutral on the user-friendliness aspects. This implies that the majority of the participants thought that the system is simple to use and intuitive. Just 3 respondents assigned a lower score of 1 or 2, suggesting that there may have been a few isolated situations in which users had difficulties, possibly due to unfamiliarity with the interface or how to interact with the device.

User feedback indicated some scepticism but was generally good regarding facial recognition accuracy. The neutral rating of 3 was chosen by the majority of respondents (5 out of 10), suggesting uncertainty or inconsistent experiences with the feature. Four users gave it the highest rating of five, indicating significant confidence in the system's identification abilities, while one respondent gave it a two, indicating slight discontent. The diverse answers indicate that while the facial recognition passed functional testing, real-world factors including user posture, processing speed, and lighting differences might have affected how dependable or "perfect" the function felt in practice.

Feedbacks on system speed were more divided. Five respondents gave lesser ratings: one person gave it a 1 (very unsatisfied), two gave it a 2, and two gave it a 3. However, four users gave performance a level 4 rating and one a level 5 rating, indicating satisfaction with response. Despite the fact that half of the users were generally satisfied with the performance, this distribution shows that the other half saw significant delays, especially in face recognition analysis and auto-unlock timing. These results confirm that even though the system functions well, performance speed optimization is still a crucial area that needs to be improved in order to increase customer happiness.

The system got the most favourable overall response in terms of security efficacy. Four respondents rated it between 1 and 4, while six gave it the maximum possible rating of 5. This shows that there is broad agreement that the system successfully prevents unauthorized access. Users' high level of confidence in the system's security was probably influenced by features like real-time face authentication, QR code validation, and the incorporation of alert systems like Telegram notifications. Despite certain areas that require improvement in terms of speed and recognition consistency, the data shows that the smart door lock is generally considered a dependable solution for home security.

Users found the system useful, particularly in terms of its security features and overall usability, according to the Likert feedback. However, concerns with processing speed and facial recognition accuracy still are present which have an impact on customer satisfaction. These results show that performance is the most crucial area for further improvement and are consistent with the qualitative recommendations.

5.3.3 Yes/No Questions Feedback

To get users' opinions swiftly and decisively, binary Yes/No questions were added in addition to the Likert scale. Two questions were asked:

- Do you find the system helpful?
- Do you think the system will enhance the security of the house?

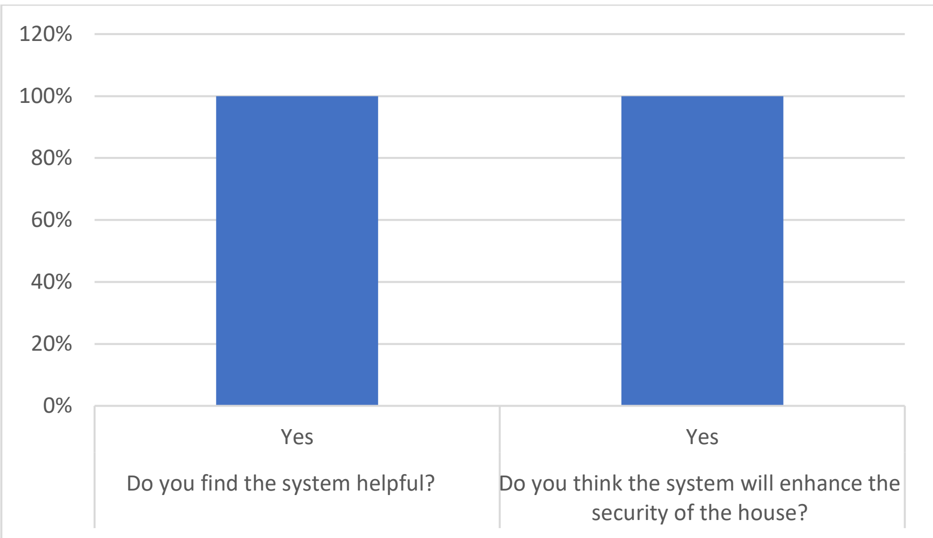


Figure 5.2: Yes/No questions feedback graph.

Figure 5.2, shows that all 10 respondents answered “Yes” for both questions, suggesting that there was 100% agreement rate. This unanimous outcome shows that users strongly agree with the systems usefulness and its ability to contribute to home safety. Despite mixed feedbacks and opinion regarding speed and performance, users can clearly see the systems value and function in a real-world application.

5.3.4 Qualitative Suggestions for Improvement

As for the qualitative suggestions improvement given by the respondent, it can be categorized into 3 category which is System Performance, New Features and No improvement. Out of 10 respondents, 3 suggests to add a new feature, 3 commented to improve the system performance speed and 4 giving no feedback on improvements.

1. System Performance

A notable concern brought up by a number of respondents related to the facial recognition system's speed and responsiveness. Users commented that the auto-lock time should be shortened and that the identification analysis takes too long to process. These recommendations point to a perceived delay between facial recognition and motion sensing. The latency impacts the overall user experience even while the system works as designed, particularly for users who anticipate fast or very instantaneous responses from the system. This input is consistent with the Likert scale result, which showed that the system speed received the most neutral or negative responses.

2. New Features

Three respondents provided feedback that focused on potential feature improvements. In order to provide more seamless and accessible experience for customers with limited mobility, an elderly user requested that the system be set up to instantly open the door after unlocking the it. Additionally, there was a comment to incorporate voice feedback or audio prompts, such as playing a “Welcome” message for approved users and a “Unauthorized user” alerting for access denials. These improvements could make the interface more engaging and help users feel more confident and clearer, especially in low-light or non-visual settings.

3. No improvements

A level of satisfaction with the system usability and efficacy is expressed by four respondents stating no desire for changes or improvements to the system current setup. This supports the finding that some users already find the core functionality to be satisfactory, despite performance issues and feature requests.

In addition to reflecting current user expectations, these insights serve as a basis for determining goals for future development iterations that strikes a compromise between everyday usability and technical performance.

5.4 Refinement

From the UAT feedbacks, the only improvement that can be done is shortening the delay on the system and auto-lock to 10 seconds. The other suggestions are not able to be done due to budgeting which require an advanced audio hardware for an audio play upon entering and a magnetic lock that is heavy and expensive for a small prototype. Aside from that, another feature is added which is a time-limit one-time QR code that will be inaccessible after 60 seconds upon generating. This feature is added due to security concern on the generated QR code that can be used anytime and still usable after months of generating when user did not use it to delete it in the system. This addition to the system proves an added security to the system making it more robust and user-friendly.

5.5 Summary

The functionality and usability of the AIoT Smart Door Lock system were assessed through thorough evaluation in this chapter. Functional testing using UAT scenarios confirmed that all important functionalities function as intended under various conditions. User feedback was used in non-functional testing to evaluate the system's dependability, responsiveness, and user experience. The results of the tests demonstrate that the system effectively uses facial recognition and real-time IoT monitoring to provide a safe, keyless entry mechanism. Despite many drawbacks, the testing results and user input provide clear guidance for future development. All things considered, the system has shown itself to be a dependable and easy-to-use smart home security solution.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 Introduction

The development of the AIoT Smart Door Lock system comes to an end in this last chapter, which also outlines possible future improvements that could be done to improve its functionality and performance. It reflects on the ability of the system to achieve the suggested goals while summarizing the project's main conclusions, results, and achievements. To make sure that all objectives have been successfully met, the chapter also compares the developed system's actual results with its intended aims. In order to give a better idea of what might be improved in the future, any limitations that were identified during the development or testing phases will also be mentioned. These restrictions provide as a roadmap for upcoming improvements, encouraging the development of a smart home security system that is more reliable, smart, and easy to use.

6.2 Objectives and Achievements

By the end of this project, the objectives outlined in Chapter 1 have been successfully achieved. The development and implementation of the AIoT Smart Door Lock system proved that combining IoT and AI technologies into a single home security solution was both feasible and efficient. By enabling secure, keyless access using facial recognition, offering real-time motion detection and alert system, and enabling remote door control and monitoring via the Blynk mobile application, the system achieved its primary objectives of developing a smart door lock with improved security and user convenience.

A thorough User Acceptance Testing (UAT) was employed to achieve the second goal, which was to assess the prototype's efficiency. Feedback from respondents was obtained to evaluate the responsiveness, usability, and dependability of the system, and functional and non-functional testing were conducted to mimic real-world circumstances. The positive results obtained from UAT confirmed the system's ability to perform well under various scenarios, reinforcing its potential for real-world implementation.

A QR code access ability was added with a specific aim of allowing temporary access in order to increase accessibility even further without compromising security. With this update, homeowners can create QR codes for guests, that enable safe, time-based entry without the need for facial recognition. This gives the system's access control mechanisms more flexibility while also making it easier for visitors to use.

To ensure that the system functions consistently under day-to-day settings, every element—from hardware configuration to software integration—was effectively implemented and tested. These achievements confirm that the proposed solution meets the intended technical and functional requirements, laying a strong foundation for future enhancements and real-world deployment.

6.3 Limitations and Constraints

While the AIoT Smart Door Lock successfully achieve its objectives, there are a few problems and limitations encountered throughout the development and testing process. The system's reliance on facial recognition technology that is sensitive to changes in environment conditions even though it works well in many situations, is one of the main challenges. For example, low light levels, harsh shadows, or the use of face masks, hats, or sunglasses that partially cover the face can all affect how accurately a person is recognized. These elements may make it more difficult for the system to accurately identify authorized users, which could result in false rejections or the inability to unlock the door.

Furthermore, the system's need on a stable internet connection for complete integration with the Blynk mobile application is another significant limitation. Remote control functions like locking, unlocking, and image notifications through the Blynk app require active internet access, even if the facial recognition capability works independently in offline mode. Users may not be able to access these essential remote features in areas where Wi-Fi is unreliable or unavailable, which restricts the system's usability and convenience.

Additionally, the prototype was tested primarily in controlled indoor environments, where it was possible to control factors like background noise, light, and distance from the camera. Unpredictable factors like weather variations, excessive background activity, or noise interference could be introduced when the system is implemented in real-world settings, particularly outside or semi-public spaces. Over time, these outside variables might affect the system's dependability and consistency.

Another limitation is power management. The system requires a constant power source, and over time, its parts—like the servo motor, Raspberry Pi, and camera—may consume a significant amount of energy. This could be an issue for installations without a reliable power source. Long-term, continuous use might not be possible without other solutions like power optimization or a battery backup. Hence, these limitations need to be address in future works in order to ensure the stability and useful implementation in future.

6.4 Future works

Even though the AIoT Smart Door Lock system prototype operates effectively and accomplishes its objectives, there are a few things that could have been improved in future work to enhance its functionality, usability, and adaptability for practical implementation. In order to overcome the constraints discovered during development and ensure that the system continues to be useful, dependable, and scalable, these future improvements are crucial.

One of the important parts that need improvements is to enhance the facial recognition algorithm by incorporating it with advanced deep learning techniques. Enhancing the facial recognition system with cutting-edge deep learning techniques is one of the main changes. At the moment, the system utilizes the standard face encoding, which works well in optimal conditions but struggles with partial occlusions or dim lighting. Using deep neural networks (DNN) from platforms like TensorFlow or OpenCV might greatly improve recognition accuracy and enable more adaptable user looks, including wearing sunglasses, face masks, or other viewpoints.

Additionally, responsiveness and speed can be enhanced. For future versions, it could improve the system's functionality and use by streamlining the face recognition process and reducing on the hardware response times, especially when unlocking or activating the sensor after detecting motions. A faster reaction time would be particularly beneficial for elderly users or users in a hurry. Furthermore, incorporating offline fallback logic for Blynk features may enhance the system's usability in areas with inconsistent or poor internet connectivity. This would enable essential functions like locking and unlocking or accessing the latest picture to continue even in a situation that the device is momentarily unavailable. Queued commands could be locally kept by the system and synchronized upon re-establishing the internet connection.

Future developments should also address the issue of power management. Implementing a battery backup or low-power standby mode could increase reliability as the system currently depends on a steady power source, particularly in areas where outages are common. More adaptable systems, like remote entries or outdoor gates, would also be possible with efficient energy use.

Finally, the system can be further enhanced to a commercial-level that provides features like mobile push alerts, user activity logging, encrypted communication, and an easy-to-use interface to manage authorized users. These future improvements will guarantee that the AIoT Smart Door Lock system is user-centric, flexible, ready for real-world use in addition to enhancing its dependability and security.

6.5 Conclusion

By combining IoT and AI technologies into a single, intelligent solution, the AIoT Smart Door Lock system shows great promise for improving home security. The system effectively accomplishes its primary goals of enabling remote control via a mobile application, real-time monitoring through motion detection, and safe, keyless access through facial recognition. The system demonstrated its use, functionality, and dependability throughout the development process, particularly for youngsters, the elderly, and homeowners. However, a few improvements may be put into consideration to make it a more practical solution for real-world deployment. These include enhancing power efficiency, strengthening the system against false alarms or environmental interference, and refining the facial recognition algorithm for a quicker reaction time. The system can develop into a more reliable, secure, and user-friendly smart home solution with improvements that prioritize scalability, accuracy, and user customization—helping to create safer and more intelligent living spaces.

References

Rahim, A., Zhong, Y., & Ahmad, T. (2023, April 9). *Enhancing Smart Home Security with Face Recognition using Deep Learning*. IJRASET. https://www.ijraset.com/research-paper/enhancing-smart-home-security-with-face-recognition

Stolojescu-Crisan, C., Crisan, C., & Butunoi, B. (2021). An IoT-Based smart home automation system. *Sensors*, *21*(11), 3784. https://doi.org/10.3390/s21113784

Zainudin, A. F. I. M., Awang, N. F., Marzukhi, S., Alsagoff, S. N., Tajuddin, T., & Jarno, A. D. (2022). A vulnerability detection framework for IoT devices: Smart lock. *AIP Conference Proceedings*. https://doi.org/10.1063/5.0119779

Balraj Balla, P., & Jadhao, K. T. (2023). Facial authentication based smart door lock system and anomaly detection using machine learning architectures integrated with IoT. *Research Square (Research Square)*. https://doi.org/10.21203/rs.3.rs-3247506/v1

Viderberg, A. (2019). *Security evaluation of smart door locks*. DIVA. https://kth.diva-portal.org/smash/record.jsf?dswid=-9520&pid=diva2%3A1336796&c=3&searchType=SIMPLE&language=en&query=Security+Evaluation+of+Smart+Locks&af=%5B%5D&aq=%5B%5B%5D%5D&aq2=%5B%5B%5D%5D&aqe=%5B%5D&noOfRows=50&sortOrder=author_sort_asc&sortOrder2=title_sort_asc&onlyFullText=false&sf=all

Rahim, A., Zhong, Y., Ahmad, T., Ahmad, S., Pławiak, P., & Hammad, M. (2023). Enhancing smart Home Security: Anomaly detection and face recognition in smart home IoT devices using Logit-Boosted CNN models. *Sensors*, *23*(15), 6979. https://doi.org/10.3390/s23156979

Y Lakshmi Venkata, N., Ch Rupa, B Dharmika, G Nithin, T., & N Vineela. (2021, August 27). *Intelligent Secure Smart Locking System using Face Biometrics*. IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9573869/authors#authors

Shanthini, M., Vidya, G., & Arun, R. (2020, August 1). *IoT enhanced smart door locking system*. IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9214288

Pacheco, J., & Miranda, K. (2020, September 1). *Design of a low-cost NFC Door Lock for a Smart Home System*. IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9216409

Aswini, D., Rohindh, R., Manoj Ragavendhara, K. S., & Mridula, C. S. (2021, October 8). *Smart door locking system*. IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/9675590

Fitriyan, R., Hamid, M., & Rajab, A. (2024). Use of IoT Technology in Home Security Monitoring Systems. *Andalas Journal of Electrical and Electronic Engineering Technology*. https://doi.org/10.25077/ajeeet.v4i1.111.

Arif, Samrah, et al. "Investigating Smart Home Security: Is Blockchain the Answer?" *IEEE Access*, vol. 8, 2020, pp. 117802–117816, https://doi.org/10.1109/access.2020.3004662.

Edu, J. S., Such, J. M., & Suarez-Tangil, G. (2020). Smart Home Personal Assistants. *ACM Computing Surveys*, *53*(6), 1–36. https://doi.org/10.1145/3412383

Hoque, M. A., & Davidson, C. (2019). Design and Implementation of an IoT-Based Smart Home Security System. *International Journal of Networked and Distributed Computing*, *7*(2), 85. https://doi.org/10.2991/ijndc.k.190326.004

Iwendi, C., Rehman, S. U., Javed, A. R., Khan, S., & Srivastava, G. (2021). Sustainable Security for the Internet of Things Using Artificial Intelligence Architectures. *ACM Transactions on Internet Technology*, *21*(3), 1–22. https://doi.org/10.1145/3448614

M. Sumathi, J Shruthi, Jain, V., G Kalyan Kumar, & Khan, Z. Z. (2023). Using Artificial Intelligence (AI) and Internet of Things (IoT) for Improving Network Security by Hybrid Cryptography Approach. *Evergreen*, *10*(2), 1133–1139. https://doi.org/10.5109/6793674

Sarhan, Q. I. (2020). Systematic Survey on Smart Home Safety and Security Systems Using the Arduino Platform. *IEEE Access*, *8*, 128362–128384. https://doi.org/10.1109/access.2020.3008610

Singh, A., Gupta, D., & Mittal, N. (2019, June 1). *Enhancing Home security systems Using IOT*. IEEE Xplore. https://doi.org/10.1109/ICECA.2019.8821833

Wu, H., Han, H., Wang, X., & Sun, S. (2020). Research on Artificial Intelligence Enhancing Internet of Things Security: A Survey. *IEEE Access*, *8*, 153826–153848. https://doi.org/10.1109/access.2020.3018170

Adiono, T., Fuada, S., Feranti, S., Gani, I., & Yusuf, M. (2019). IoT-Enabled Door Lock System. *International Journal of Advanced Computer Science and Applications*, *10*(5). https://doi.org/10.14569/ijacsa.2019.0100556

Doshi, O., Bendale, H. N., Chavan, A. C., & More, S. S. (2022). A Smart Door Lock Security System using Internet of Things. *2022 International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*. https://doi.org/10.1109/icaaic53929.2022.9792831

Kumbhar, N., & Deshmukh, P. M. (2017). Smart Door Locking System using Wireless Communication Technology. *International Journal for Research in Applied Science and Engineering Technology*, *V*(VIII), 507–512. https://doi.org/10.22214/ijraset.2017.8070

Patil, K. A., Niteen Vittalkar, Hiremath, P., & Murthy, M. A. (2020). Smart Door Locking System using IoT. *International Journal of Engineering and Technology*, *7*(5), 2395-0056. https://www.researchgate.net/publication/341508373_Smart_Door_Locking_System_using_IoT

Rahman, M., Ali, M. S., & Akther, S. (2018). Password Protected Electronic Lock System for Smart Home Security. *International Journal of Engineering Research And*, *V7*(04). https://doi.org/10.17577/ijertv7is040396

Saputra, R., & Surantha, N. (2021). Smart and real-time door lock system for an elderly user based on face recognition. *Bulletin of Electrical Engineering and Informatics*, *10*(3), 1345–1355. https://doi.org/10.11591/eei.v10i3.2955

Shaik, N., Hameed, A., Asuncion, N. R., & Prabhu, N. M. (2024). Smart Security: An IoT-NFC Lock System for Efficient Access Management. *Malaysian Journal of Science and Advanced Technology*, *4*(3), 322–329. https://doi.org/10.56532/mjsat.v4i3.313

Russell, S., & Norvig, P. (2021). *Artificial intelligence: A modern approach* (4th ed.). Pearson.

Appendices

Appendix A

Project Schedule

Table 1: Date Schedule for FYP 1

Description	Start Date	End Date	Number of Days
FYP 1	07/10/2024	12/01/2025	97
FYP 1 briefing	10/10/2024	10/10/2024	1
Identify project title and look for a supervisor	11/10/2024	13/10/2024	3
Prepare brief proposal	13/10/2024	16/10/2024	4
Submit brief proposal	16/10/2024	16/10/2024	1
Feedback from supervisor	17/10/2024	17/10/2024	1
Prepare full proposal	18/10/2024	07/11/2024	21
Submit full proposal	08/11/2024	08/11/2024	1
Analysis and quick design	09/11/2024	04/01/2025	56
Chapter 1: Introduction - Find out the scope and	09/11/2024	20/11/2024	5

significance of the project			
Submission Chapter 1	21/11/2024	21/11/2024	1
Chapter 2: Background study - Finalize full literature review	22/11/2024	12/12/2024	21
Submission of Chapter 2	13/12/2024	13/12/2024	1
Chapter 3: Requirement analysis & design - Specify methodology used - Provide requirement analysis	14/12/2024	04/01/2025	22
Submission of Chapter 3	05/01/2025	05/01/2025	1
FYP 1 report submission	12/01/2025	12/01/2025	1

Table 2: Date Schedule for FYP 2

Description	**Start Date**	**End Date**	**Number of Days**
FYP 2	10/03/2025	28/7/2025	141
Chapter 4: Design and Development - Design the system. - Develop the system. - Data collection.	10/03/2025	15/05/2025	67
Chapter 5: Testing and Refinement - Test the system using UAT feedback. - Refine the system based on the feedback received.	16/05/2025	31/05/2025	16
Chapter 6: Conclusion	16/05/2025	31/05/2025	16
FYP report submission	23/06/2025	23/06/2025	2
FYP Symposium	30/06/2025	01/07/2025	1

FYP report submission after amendment	28/07/2025	28/07/2025	1

Appendix B

Feedback of User Acceptance Testing

User Acceptance Testing

User Category: Homeowner ( ) Elderly (✓) Guests/Visitor ( )

Test Case ID	Scenario	Expected Outcome	(√/X)	User Comment
TC01	User approaches the door, and the PIR motion sensor detects movement	Camera is activated	✓	
TC02	Authorized user's face is detected	Door unlocks automatically	✓	
~~TC03~~	~~Unknown face detected~~	~~Door remains locked & plays an alert~~		
~~TC04~~	~~No movement for 30 seconds after PIR detects motion~~	~~LED lights & speaker activate an alert~~		
TC05	User unlocks the door via Blynk app	Door unlocks remotely	✓	
TC06	User locks the door via Blynk app	Door locks remotely	✓	
TC07	User approaches the door but is wearing a mask or sunglasses	System still recognizes the user and unlocks the door (if facial recognition is robust)	✓	
TC08	Multiple faces detected at once (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	✓	
TC09	Blynk app fails to connect due to no internet	User should still be able to unlock via facial recognition	✓	
TC10	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	✓	
TC11	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	✓	
TC12	Unauthorized face detected	System saves captured image and sends alert via Telegram	✓	
TC13	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	✓	

User Acceptance Testing

User Category: Homeowner ( ) Elderly (/) Guests/Visitor ( )

Test Case ID	Scenario	Expected Outcome	(√/X)	User Comment
TC01	User approaches the door, and the PIR motion sensor detects movement	Camera is activated	√	
TC02	Authorized user's face is detected	Door unlocks automatically	√	
~~TC03~~	~~Unknown face detected~~	~~Door remains locked & plays an alert~~		
~~TC04~~	~~No movement for 30 seconds after PIR detects motion~~	~~LED lights & speaker activate an alert~~		
TC05	User unlocks the door via Blynk app	Door unlocks remotely	√	
TC06	User locks the door via Blynk app	Door locks remotely	√	
TC07	User approaches the door but is wearing a mask or sunglasses	System still recognizes the user and unlocks the door (if facial recognition is robust)	√	
TC08	Multiple faces detected at once (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	√	
TC09	Blynk app fails to connect due to no internet	User should still be able to unlock via facial recognition	√	
TC10	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	√	
TC11	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	√	
TC12	Unauthorized face detected	System saves captured image and sends alert via Telegram	√	
TC13	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	√	

User Acceptance Testing

User Category: Homeowner (√) Elderly ( ) Guests/Visitor ( )

Test Case ID	Scenario	Expected Outcome	(√/X)	User Comment
TC01	User approaches the door, and the PIR motion sensor detects movement	Camera is activated	√	
TC02	Authorized user's face is detected	Door unlocks automatically	√	
~~TC03~~	~~Unknown face detected~~	~~Door remains locked & plays an alert~~		
~~TC04~~	~~No movement for 30 seconds after PIR detects motion~~	~~LED lights & speaker activate an alert~~		
TC05	User unlocks the door via Blynk app	Door unlocks remotely	√	
TC06	User locks the door via Blynk app	Door locks remotely	√	
TC07	User approaches the door but is wearing a mask or sunglasses	System still recognizes the user and unlocks the door (if facial recognition is robust)	√	
TC08	Multiple faces detected at once (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	√	
TC09	Blynk app fails to connect due to no internet	User should still be able to unlock via facial recognition	√	
TC10	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	√	
TC11	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	√	
TC12	Unauthorized face detected	System saves captured image and sends alert via Telegram	√	
TC13	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	√	

User Acceptance Testing

User Category: Homeowner (√) Elderly ( ) Guests/Visitor ( )

Test Case ID	Scenario	Expected Outcome	(√/X)	User Comment
TC01	User approaches the door, and the PIR motion sensor detects movement	Camera is activated	√	
TC02	Authorized user's face is detected	Door unlocks automatically	√	
~~TC03~~	~~Unknown face detected~~	~~Door remains locked & plays an alert~~		
~~TC04~~	~~No movement for 30 seconds after PIR detects motion~~	~~LED lights & speaker activate an alert~~		
TC05	User unlocks the door via Blynk app	Door unlocks remotely	√	
TC06	User locks the door via Blynk app	Door locks remotely	√	
TC07	User approaches the door but is wearing a mask or sunglasses	System still recognizes the user and unlocks the door (if facial recognition is robust)	√	
TC08	Multiple faces detected at once (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	√	
TC09	Blynk app fails to connect due to no internet	User should still be able to unlock via facial recognition	√	
TC10	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	√	
TC11	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	√	
TC12	Unauthorized face detected	System saves captured image and sends alert via Telegram	√	
TC13	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	√	

User Acceptance Testing

User Category: Homeowner (✓) Elderly ( ) Guests/Visitor ( )

Test Case ID	Scenario	Expected Outcome	(√/X)	User Comment
TC01	User approaches the door, and the PIR motion sensor detects movement	Camera is activated	✓	
TC02	Authorized user's face is detected	Door unlocks automatically	✓	
~~TC03~~	~~Unknown face detected~~	~~Door remains locked & plays an alert~~		
~~TC04~~	~~No movement for 30 seconds after PIR detects motion~~	~~LED lights & speaker activate an alert~~		
TC05	User unlocks the door via Blynk app	Door unlocks remotely	✓	
TC06	User locks the door via Blynk app	Door locks remotely	✓	
TC07	User approaches the door but is wearing a mask or sunglasses	System still recognizes the user and unlocks the door (if facial recognition is robust)	✓	
TC08	Multiple faces detected at once (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	✓	
TC09	Blynk app fails to connect due to no internet	User should still be able to unlock via facial recognition	✓	
TC10	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	✓	
TC11	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	✓	
TC12	Unauthorized face detected	System saves captured image and sends alert via Telegram	✓	
TC13	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	✓	

User Acceptance Testing

User Category: Homeowner (√) Elderly ( ) Guests/Visitor ( )

Test Case ID	Scenario	Expected Outcome	(√/X)	User Comment
TC01	User approaches the door, and the PIR motion sensor detects movement	Camera is activated	√	
TC02	Authorized user's face is detected	Door unlocks automatically	√	
TC03	Unknown face detected	Door remains locked & plays an alert		
TC04	No movement for 30 seconds after PIR detects motion	LED lights & speaker activate an alert		
TC05	User unlocks the door via Blynk app	Door unlocks remotely	√	
TC06	User locks the door via Blynk app	Door locks remotely	√	
TC07	User approaches the door but is wearing a mask or sunglasses	System still recognizes the user and unlocks the door (if facial recognition is robust)	√	
TC08	Multiple faces detected at once (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	√	
TC09	Blynk app fails to connect due to no internet	User should still be able to unlock via facial recognition	√	
TC10	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	√	
TC11	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	√	
TC12	Unauthorized face detected	System saves captured image and sends alert via Telegram	√	
TC13	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	√	

User Acceptance Testing

User Category: Homeowner ( ) Elderly ( ) Guests/Visitor (√)

Test Case ID	Scenario	Expected Outcome	(√/X)	User Comment
TC01	User approaches the door, and the PIR motion sensor detects movement	Camera is activated	√	
TC02	Authorized user's face is detected	Door unlocks automatically	√	
TC03	Unknown face detected	Door remains locked & plays an alert		
TC04	No movement for 30 seconds after PIR detects motion	LED lights & speaker activate an alert	X	
TC05	User unlocks the door via Blynk app	Door unlocks remotely	√	
TC06	User locks the door via Blynk app	Door locks remotely	√	
TC07	User approaches the door but is wearing a mask or sunglasses	System still recognizes the user and unlocks the door (if facial recognition is robust)	√	
TC08	Multiple faces detected at once (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	√	
TC09	Blynk app fails to connect due to no internet	User should still be able to unlock via facial recognition	√	
TC10	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	√	
TC11	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	√	
TC12	Unauthorized face detected	System saves captured image and sends alert via Telegram	√	
TC13	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	√	

User Acceptance Testing

User Category: Homeowner ( ) Elderly ( ) Guests/Visitor (✓)

Test Case ID	Scenario	Expected Outcome	(√/X)	User Comment
TC01	User approaches the door, and the PIR motion sensor detects movement	Camera is activated	✓	
TC02	Authorized user's face is detected	Door unlocks automatically	✓	
TC03	Unknown face detected	Door remains locked & plays an alert		
TC04	No movement for 30 seconds after PIR detects motion	LED lights & speaker activate an alert	x	
TC05	User unlocks the door via Blynk app	Door unlocks remotely	✓	
TC06	User locks the door via Blynk app	Door locks remotely	✓	
TC07	User approaches the door but is wearing a mask or sunglasses	System still recognizes the user and unlocks the door (if facial recognition is robust)	✓	
TC08	Multiple faces detected at once (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	✓	
TC09	Blynk app fails to connect due to no internet	User should still be able to unlock via facial recognition	✓	
TC10	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	✓	
TC11	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	✓	
TC12	Unauthorized face detected	System saves captured image and sends alert via Telegram	✓	
TC13	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	✓	

User Acceptance Testing

User Category: Homeowner ( ) Elderly ( ) Guests/Visitor (✓)

Test Case ID	Scenario	Expected Outcome	(√/X)	User Comment
TC01	User approaches the door, and the PIR motion sensor detects movement	Camera is activated	✓	
TC02	Authorized user's face is detected	Door unlocks automatically	✓	
TC03	Unknown face detected	Door remains locked & plays an alert		
TC04	No movement for 30 seconds after PIR detects motion	LED lights & speaker activate an alert		
TC05	User unlocks the door via Blynk app	Door unlocks remotely	✓	
TC06	User locks the door via Blynk app	Door locks remotely	✓	
TC07	User approaches the door but is wearing a mask or sunglasses	System still recognizes the user and unlocks the door (if facial recognition is robust)	✓	
TC08	Multiple faces detected at once (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	✓	
TC09	Blynk app fails to connect due to no internet	User should still be able to unlock via facial recognition		
TC10	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	✓	
TC11	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	✓	
TC12	Unauthorized face detected	System saves captured image and sends alert via Telegram	✓	
TC13	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	✓	

User Acceptance Testing

User Category: Homeowner ( ) Elderly (√) Guests/Visitor ( )

Test Case ID	Scenario	Expected Outcome	(√/X)	User Comment
TC01	User approaches the door, and the PIR motion sensor detects movement	Camera is activated	√	
TC02	Authorized user's face is detected	Door unlocks automatically	√	
~~TC03~~	~~Unknown face detected~~	~~Door remains locked & plays an alert~~		
~~TC04~~	~~No movement for 30 seconds after PIR detects motion~~	~~LED lights & speaker activate an alert~~		
TC05	User unlocks the door via Blynk app	Door unlocks remotely	√	
TC06	User locks the door via Blynk app	Door locks remotely	√	
TC07	User approaches the door but is wearing a mask or sunglasses	System still recognizes the user and unlocks the door (if facial recognition is robust)	√	
TC08	Multiple faces detected at once (e.g., family approaching together)	System correctly identifies at least one authorized user and unlocks	√	
TC09	Blynk app fails to connect due to no internet	User should still be able to unlock via facial recognition	√	
TC10	Guest scans valid QR code shown on their phone	System grants access and unlocks the door	√	
TC11	Guest scans an **expired or invalid** QR code	System denies access and triggers alert	√	
TC12	Unauthorized face detected	System saves captured image and sends alert via Telegram	√	
TC13	Manual alarm triggered from Blynk app	Buzzer activates but no Telegram message is sent	√	

Appendix C

Feedbacks of User Likert Scale

Likert Scale Feedback Form

Please (√) in the box.

1 = Very Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Very Agree

Questions	1	2	3	4	5
Is it user-friendly?			✓		
Does the facial recognition function work perfectly?			✓		
Is it fast?				✓	
Do you think it help prevent unauthorized access?					✓

Do you find the system helpful? Yes (✓) No ( )

Do you think the system will enhance the security of the house? Yes (✓) No ( )

Do you have any suggestions for improvement?

-

Likert Scale Feedback Form

Please (√) in the box.

1 = Very Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Very Agree

Questions	1	2	3	4	5
Is it user-friendly?			✓		
Does the facial recognition function work perfectly?			✓		
Is it fast?					✓
Do you think it help prevent unauthorized access?					✓

Do you find the system helpful? Yes (✓) No ( )

Do you think the system will enhance the security of the house? Yes (✓) No ( )

Do you have any suggestions for improvement?

[illegible]

Likert Scale Feedback Form

Please (√) in the box.

1 = Very Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Very Agree

Questions	1	2	3	4	5
Is it user-friendly?					/
Does the facial recognition function work perfectly?					/
Is it fast?			/		
Do you think it help prevent unauthorized access?					/

Do you find the system helpful? Yes (/) No ( )

Do you think the system will enhance the security of the house? Yes (/) No ( )

Do you have any suggestions for improvement?

MAKE IT FASTER.

Likert Scale Feedback Form

Please (√) in the box.

1 = Very Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Very Agree

Questions	1	2	3	4	5
Is it user-friendly?				/	
Does the facial recognition function work perfectly?			/		
Is it fast?		/			
Do you think it help prevent unauthorized access?			/		

Do you find the system helpful? Yes (/) No ( )

Do you think the system will enhance the security of the house? Yes (/) No ( )

Do you have any suggestions for improvement?

MAKE it faster

Likert Scale Feedback Form

Please (√) in the box.

1 = Very Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Very Agree

Questions	1	2	3	4	5
Is it user-friendly?		√			
Does the facial recognition function work perfectly?		√			
Is it fast?		√			
Do you think it help prevent unauthorized access?		√			

Do you find the system helpful? Yes (√) No ( )

Do you think the system will enhance the security of the house? Yes (√) No ( )

Do you have any suggestions for improvement?

–

Likert Scale Feedback Form

Please (√) in the box.

1 = Very Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Very Agree

Questions	1	2	3	4	5
Is it user-friendly?	√				
Does the facial recognition function work perfectly?			√		
Is it fast?	√				
Do you think it help prevent unauthorized access?	√				

Do you find the system helpful? Yes (√) No ( )

Do you think the system will enhance the security of the house? Yes (√) No ( )

Do you have any suggestions for improvement?

–

Likert Scale Feedback Form

Please (√) in the box.

1 = Very Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Very Agree

Questions	1	2	3	4	5
Is it user-friendly?	✓				
Does the facial recognition function work perfectly?			✓		
Is it fast?				✓	
Do you think it help prevent unauthorized access?				✓	

Do you find the system helpful? Yes (✓) No ( )

Do you think the system will enhance the security of the house? Yes (✓) No ( )

Do you have any suggestions for improvement?

Pendekkan masa pintu tutup.

Likert Scale Feedback Form

Please (√) in the box.

1 = Very Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Very Agree

Questions	1	2	3	4	5
Is it user-friendly?					✓
Does the facial recognition function work perfectly?					✓
Is it fast?				✓	
Do you think it help prevent unauthorized access?					✓

Do you find the system helpful? Yes (✓) No ( )

Do you think the system will enhance the security of the house? Yes (✓) No ( )

Do you have any suggestions for improvement?

make the recognition analysis faster.

Likert Scale Feedback Form

Please (√) in the box.

1 = Very Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Very Agree

Questions	1	2	3	4	5
Is it user-friendly?					✓
Does the facial recognition function work perfectly?					✓
Is it fast?				✓	
Do you think it help prevent unauthorized access?					✓

Do you find the system helpful? Yes (✓) No ( )

Do you think the system will enhance the security of the house? Yes (✓) No ( )

Do you have any suggestions for improvement?

Add audio

Likert Scale Feedback Form

Please (√) in the box.

1 = Very Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Very Agree

Questions	1	2	3	4	5
Is it user-friendly?					✓
Does the facial recognition function work perfectly?					✓
Is it fast?			✓		
Do you think it help prevent unauthorized access?					✓

Do you find the system helpful? Yes (√) No ( )

Do you think the system will enhance the security of the house? Yes (√) No ( )

Do you have any suggestions for improvement?

No improvement needed

Appendix D

Source Code (main.py)

```
import os

import time

import threading

from datetime import datetime

import RPi.GPIO as GPIO

from BlynkLib import Blynk

import qrcode

import requests

import telegram

from reportlab.lib.pagesizes import letter

from reportlab.pdfgen import canvas

from reportlab.lib.units import inch

from camera import image

from frecog import face

from lock import unlock, locked, setup as lock_setup

from alert import full_alert

from user import capture_image


BLYNK_AUTH = 'ytuqosWw1pHZEflhNxlpkWMDeojmYQrF'

IMGBB_API_KEY = '229c2f16447f438ebbcde023bd6d1b98'

BOT_TOKEN = "8134479864:AAGSBwlWt5cxfY3E6y6qYjEBGhiENUnbT6U"

CHAT_ID = "539181152"


AUTH_DIR = 'authorized_access'
```

```
UNAUTH_DIR = 'unauthorized_access'
QR_DIR = 'qr'
os.makedirs(QR_DIR, exist_ok=True, mode=0o777)
os.makedirs(AUTH_DIR, exist_ok=True, mode=0o777)
os.makedirs(UNAUTH_DIR, exist_ok=True, mode=0o777)

blynk = Blynk(BLYNK_AUTH)

activity_log = []
log_lock = threading.Lock()

def update_activity_log(event):
    with log_lock:
        timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
        log_entry = f"[{timestamp}] {event}"
        print(log_entry)
        activity_log.append(log_entry)

#BLYNK EVENT HANDLERS
@blynk.on("V0") # DOOR LOCK CONTROL
def blynk_lock_control(value):
    if int(value[0]) == 1:
        print("Locking door (app)")
        locked()
        update_activity_log("Door locked from app")
    else:
        print("Unlocking door (app)")
```

```
        unlock(with_autolock=False)
        update_activity_log("Door unlocked from app")


@blynk.on("V1") # MANUAL ALARM
def blynk_manual_alert(value):
    if int(value[0]) == 1:
        print("Manual alarm triggered (app)")
        full_alert()
        update_activity_log("Manual alarm triggered")


@blynk.on("V3") # REGISTER NEW USER
def register_new_user(value):
    if int(value[0]) == 1:
        print("Registering new authorized user...")
        success = capture_image()
        if success:
            print("New user added.")
            update_activity_log("New authorized user registered")
        else:
            print("Failed to add user.")


@blynk.on("V7") # GUEST QR GENERATOR
def generate_qr(value):
    if int(value[0]) == 1:
        timestamp = datetime.now().strftime('%Y%m%d_%H%M%S')
        guest_id = f"guest_{timestamp}"
        qr_data = f"guest_access:{guest_id}"
```

```
        qr_path = os.path.join(QR_DIR, f"{guest_id}.png")

        img = qrcode.make(qr_data)
        img.save(qr_path)
        print(f"[INFO] QR saved: {qr_path}")

        url = upload_to_imgbb(qr_path)
        if url:
            blynk.virtual_write(2, url)
            print(f"[SUCCESS] QR uploaded. URL: {url}")
            update_activity_log("Guest QR generated")
        else:
            print("[ERROR] QR upload failed.")
            blynk.virtual_write(2, "Upload failed.")

@blynk.on("V8") # GENERATE PDF REPORT
def generate_report_trigger(value):
    if int(value[0]) == 1:
        print("[REPORT] Generating PDF report...")
        timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
        pdf_path = f"/home/srmzl/fyp/report_{timestamp}.pdf"

        generate_pdf_report(pdf_path)

        try:
            bot = telegram.Bot(token=BOT_TOKEN)
            with open(pdf_path, 'rb') as f:
```

```
                bot.send_document(chat_id=CHAT_ID,    document=f,
filename=f"SmartLock_Report_{timestamp}.pdf")
            print("[REPORT] PDF sent to Telegram.")
            update_activity_log("PDF report generated")
        except Exception as e:
            print(f"[REPORT] Failed to send PDF: {e}")

        with log_lock: # Clear the log after reporting
            activity_log.clear()
            update_activity_log("Activity log cleared after report.")


def upload_to_imgbb(file_path):
    try:
        with open(file_path, 'rb') as f:
            response = requests.post(
                'https://api.imgbb.com/1/upload',
                params={'key': IMGBB_API_KEY},
                files={'image': f},
                timeout=15
            )
        if response.status_code == 200:
            return response.json()['data']['url']
        else:
            print(f"[ERROR] imgbb upload failed: {response.text}")
            return None
    except Exception as e:
```

```
        print(f"[ERROR] Exception in imgbb upload: {e}")
        return None


def update_authorized_list_on_blynk():
    files = os.listdir(AUTH_DIR)
    names = "\n".join([f for f in files if f.endswith(('.jpg', '.png'))])
    blynk.virtual_write(4, names if names else "No authorized users.")
    print("Authorized list on Blynk updated.")


def update_unauthorized_image_on_blynk():
    files = sorted([f for f in os.listdir(UNAUTH_DIR) if f.endswith(('.jpg', '.png'))], reverse=True)
    if files:
        image_path = os.path.join(UNAUTH_DIR, files[0])
        url = upload_to_imgbb(image_path)
        if url:
            blynk.virtual_write(5, url)
            print(f"[SUCCESS] Unauthorized image link sent to Blynk: {url}")
        else:
            blynk.virtual_write(5, "Upload failed.")
    else:
        blynk.virtual_write(5, "No unauthorized data.")
    print("Unauthorized image on Blynk updated.")


def generate_pdf_report(pdf_path):
    c = canvas.Canvas(pdf_path, pagesize=letter)
    width, height = letter
    line_height = 14
```

```
y = height - 1 * inch


c.setFont("Helvetica-Bold", 16)
now = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
c.drawString(1 * inch, y, f"Smart Door Lock System Report ({now})")
y -= 0.5 * inch


c.setFont("Helvetica-Bold", 12)
c.drawString(1 * inch, y, "Authorized Users:")
y -= line_height
for file in os.listdir(AUTH_DIR):
    if file.endswith(('.jpg', '.png')):
        c.setFont("Helvetica", 10)
        c.drawString(1.2 * inch, y, f"✔ {file}")
        y -= line_height
        if y < inch: c.showPage(); y = height - inch


y -= 0.3 * inch
c.setFont("Helvetica-Bold", 12)
c.drawString(1 * inch, y, "Unauthorized Access Log:")
y -= line_height
for file in os.listdir(UNAUTH_DIR):
    if file.endswith(('.jpg', '.png')):
        c.setFont("Helvetica", 10)
        c.drawString(1.2 * inch, y, f" {file}")
        y -= line_height
        if y < inch: c.showPage(); y = height - inch
```

```
    y -= 0.3 * inch
    c.setFont("Helvetica-Bold", 12)
    c.drawString(1 * inch, y, "Activity Log:")
    y -= line_height
    with log_lock:
        for log in activity_log[-20:]:
            c.setFont("Helvetica", 10)
            c.drawString(1.2 * inch, y, f"- {log}")
            y -= line_height
            if y < inch: c.showPage(); y = height - inch

    c.save()
    print(f"PDF report generated at {pdf_path}")

def motion_sensor_loop():
    PIR_PIN = 4
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(PIR_PIN, GPIO.IN)
    lock_setup()

    print("Motion sensor thread started...")
    while True:
        if GPIO.input(PIR_PIN):
            update_activity_log("Motion detected!")

            img_path = image()
```

```
            print("Analyzing image...")
            is_authorized = face(img_path)

            if is_authorized:
                unlock(with_autolock=True)
                update_activity_log("Door unlocked via face/QR")
            else:
                locked()
                full_alert("Unauthorized access detected!", img_path)
                update_activity_log("Unauthorized access attempt")

            time.sleep(5)
        time.sleep(0.2)


def run_main_app():
    last_auth_files = set()
    last_unauth_file = ""
    last_log_count = 0

    update_authorized_list_on_blynk()
    update_unauthorized_image_on_blynk()
    update_activity_log("System started")

    while True:
        blynk.run()
```

```
        current_auth_files = set(f for f in os.listdir(AUTH_DIR) if f.endswith(('.jpg', '.png')))
        if current_auth_files != last_auth_files:
            update_authorized_list_on_blynk()
            last_auth_files = current_auth_files

        unauth_files = sorted([f for f in os.listdir(UNAUTH_DIR) if f.endswith(('.jpg', '.png'))],
reverse=True)
        if unauth_files and unauth_files[0] != last_unauth_file:
            update_unauthorized_image_on_blynk()
            last_unauth_file = unauth_files[0]

        with log_lock:
            if len(activity_log) > last_log_count:
                blynk.virtual_write(6, "\n".join(activity_log[-5:]))
                last_log_count = len(activity_log)

        time.sleep(0.1)


if __name__ == "__main__":
    sensor_thread = threading.Thread(target=motion_sensor_loop, daemon=True)
    sensor_thread.start()

    try:
        run_main_app()
    except KeyboardInterrupt:
        print("\nExiting application...")
```

```
finally:
    GPIO.cleanup()
```

Source Code (camera.py)

```
import os
import time


def  image():
    timestamp = time.strftime("%Y%m%d-%H%M%S")
    img_path = f"/home/srmzl/fyp/images/{timestamp}.jpg"


    os.system(f"libcamera-still -n -o {img_path}")
    print(f"Image saved to {img_path}")


    return img_path
```

Code Script (frecog.py)

```
import face_recognition
import os
import shutil
```

```
import time
from pyzbar import pyzbar
from PIL import Image

authorized_access = "/home/srmzl/fyp/authorized_access/"
unauthorized_access = "/home/srmzl/fyp/unauthorized_access/"
os.makedirs(unauthorized_access, exist_ok=True)

known_face_encodings = []
known_face_names = []


for img_file in os.listdir(authorized_access):
    if img_file.endswith(".jpg") or img_file.endswith(".png"):
        img_path = os.path.join(authorized_access, img_file)
        image = face_recognition.load_image_file(img_path)
        encodings = face_recognition.face_encodings(image)
        if encodings:
            known_face_encodings.append(encodings[0])
            known_face_names.append(os.path.splitext(img_file)[0])

print(f"Loaded {len(known_face_encodings)} known faces.")

def scan_qr(img_path):
    used_qr_file = "used_qr.txt"
    try:
        image = Image.open(img_path)
```

```
decoded_objects = pyzbar.decode(image)
for obj in decoded_objects:
    qr_data = obj.data.decode('utf-8')
    print(f"[QR] Detected: {qr_data}")

    if qr_data.startswith("guest_access:"):
        with open(used_qr_file, 'a+') as f:
            f.seek(0)
            used_codes = set(line.strip() for line in f.readlines())
            if qr_data in used_codes:
                print("[QR] QR already used.")
                return False

            f.write(qr_data + "\n")
            print("[QR] Guest QR accepted")

            guest_id = qr_data.split(":")[1]
            qr_path = os.path.join("qr", f"{guest_id}.png")
            if os.path.exists(qr_path):
                os.remove(qr_path)
                print(f"[QR] Deleted used QR image: {qr_path}")

            try:
                import BlynkLib
                blynk = BlynkLib.Blynk('ytuqosWw1pHZEflhNxlpkWMDeojmYQrF')
                blynk.virtual_write(2, "")
                print("[QR] Cleared Blynk V2 display")
```

```
                except Exception as e:
                    print(f"[QR] Could not clear Blynk V2: {e}")

                return True
    except Exception as e:
        print(f"[QR] Error reading QR: {e}")
    return False


def face(img_path):
    print(f"[FR] Checking: {img_path}")
    image = face_recognition.load_image_file(img_path)
    encodings = face_recognition.face_encodings(image)

    if not encodings:
        print("[FR] No face found.")
        if scan_qr(img_path):
            print("[ACCESS] Guest access granted via QR")
            return True
        timestamp = time.strftime("%Y%m%d-%H%M%S")
        filename = f"unauth_noface_{timestamp}.jpg"
        dest_path = os.path.join(unauthorized_access, filename)
        shutil.copy(img_path, dest_path)
        print(f"[FR] Saved to: {dest_path}")
        return False

    for encoding in encodings:
```

```
            results = face_recognition.compare_faces(known_face_encodings, encoding,
tolerance=0.5)
        if True in results:
            matched_idx = results.index(True)
            print(f"[FR] Authorized user: {known_face_names[matched_idx]}")
            return True

    print("[FR] No matching authorized face.")
    if scan_qr(img_path):
        print("[ACCESS] Guest access granted via QR")
        return True

    timestamp = time.strftime("%Y%m%d-%H%M%S")
    filename = f"unauth_{timestamp}.jpg"
    dest_path = os.path.join(unauthorized_access, filename)
    shutil.copy(img_path, dest_path)
    print(f"[FR] Unauthorized! Image saved to: {dest_path}")
    return False
```

Code Script (lock.py)

```
import RPi.GPIO as GPIO

import time


GPIO.setmode(GPIO.BCM)


SERVO_PIN = 18

GPIO.setup(SERVO_PIN, GPIO.OUT)


servo = None


def setup():

    global servo

    if servo is None:

        GPIO.setup(SERVO_PIN, GPIO.OUT)

        servo = GPIO.PWM(SERVO_PIN, 50)

        servo.start(0)

        print("Servo initialized")


def unlock(with_autolock=True):

    setup()

    print("Unlocking door...")

    servo.ChangeDutyCycle(7.5)

    time.sleep(1)

    servo.ChangeDutyCycle(0)
```

```
    if with_autolock:
        print("10 seconds before auto-lock...")
        time.sleep(10)
        print("Locking door...")
        servo.ChangeDutyCycle(2.5)
        time.sleep(1)
        servo.ChangeDutyCycle(0)


def locked():
    setup()
    print("Locking door...")
    servo.ChangeDutyCycle(2.5)
    time.sleep(1)
    servo.ChangeDutyCycle(0)
```

Source Code (alert.py)

```
import RPi.GPIO as GPIO
import time
import os
import requests
import socket


BUZZER_PIN = 16
```

```
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(BUZZER_PIN, GPIO.OUT)
buzzer_pwm = GPIO.PWM(BUZZER_PIN, 1000)


BOT_TOKEN = "8134479864:AAGSBwlWt5cxfY3E6y6qYjEBGhiENUnbT6U"
CHAT_ID = "539181152"
IMGBB_API_KEY = '229c2f16447f438ebbcde023bd6d1b98'


def is_connected():
    try:
        socket.create_connection(("8.8.8.8", 53), timeout=3)
        return True
    except:
        return False

def beep(duration=0.3, freq=1000):
    buzzer_pwm.ChangeFrequency(freq)
    buzzer_pwm.start(50)
    time.sleep(duration)
    buzzer_pwm.stop()

def t_alert():
    print("[ALERT] Triggering buzzer...")
    for _ in range(5):
        beep(0.3, 1000)
```

```
        time.sleep(0.2)
    print("[ALERT] Buzzer alert complete.")


def send_telegram_alert(message, image_path=None):
    if not is_connected():
        print("[TELEGRAM] No internet connection.")
        return

    try:
        if image_path and os.path.exists(image_path):
            print("[IMGBB] Uploading image...")
            with open(image_path, 'rb') as f:
                res = requests.post(
                    "https://api.imgbb.com/1/upload",
                    params={"key": IMGBB_API_KEY},
                    files={"image": f},
                    timeout=10
                )
            if res.status_code == 200:
                img_url = res.json()["data"]["url"]
                message += f"\n\nUnauthorized image:\n{img_url}"
            else:
                print(f"[IMGBB] Failed to upload image: {res.text}")

        data = {'chat_id': CHAT_ID, 'text': message}
        response = requests.post(
            f"https://api.telegram.org/bot{BOT_TOKEN}/sendMessage",
```

```
            data=data, timeout=10
        )

        if response.status_code == 200:
            print("[TELEGRAM] Message sent successfully.")
        else:
            print(f"[TELEGRAM] Failed: {response.status_code}, {response.text}")

    except Exception as e:
        print(f"[TELEGRAM] Error:\n{e}")


def full_alert(message="Intruder detected!", image_path=None):
    try:
        t_alert()
        send_telegram_alert(message, image_path)
    finally:
        buzzer_pwm.stop()


if __name__ == "__main__":
    try:
        test_image = "/home/srmzl/fyp/unauthorized_access/sample.jpg"
        full_alert("ALERT! from Smart Lock!", test_image)
    except KeyboardInterrupt:
        print("\n[EXIT] Interrupted by user.")
    finally:
        GPIO.cleanup()
```

Source Code (user.py)

```
import os
import subprocess
from time import sleep


def get_next_filename(folder):
    os.makedirs(folder, exist_ok=True)
    count = 1
    while True:
        filename = f"user{count}.jpg"
        if not os.path.exists(os.path.join(folder, filename)):
            return filename
        count += 1


def capture_image():
    save_dir = "/home/srmzl/fyp/authorized_access/"
    filename = get_next_filename(save_dir)
    filepath = os.path.join(save_dir, filename)

    print("🔥 Warming up camera (3 seconds)...")
    sleep(3)

    try:
        subprocess.run([
            "libcamera-still",
            "-o", filepath,
```

```
            "--nopreview",
            "--timeout", "1000",
            "--width", "3280",
            "--height", "2464",
            "--brightness", "0.2",
            "--contrast", "1.2"
        ], check=True)
        print(f"Image saved as {filename}")
        return True
    except subprocess.CalledProcessError as e:
        print(f"Capture failed: {e}")
        return False


if __name__ == "__main__":
    print("Press ENTER to capture...")
    input()
    capture_image()
```

Appendix E

Images Collected