DOORBELL MONITORING SYSTEM BASED ON IOT

RAWEZH RIZGAR KARIM

QAIWAN INTERNATIONAL UNIVERSITY

PSZ 19:16 (Pind. 1/13)

UNIVERSITI TEKNOLOGI MALAYSIA

DECLARATION OF THESIS / UNDERGRADUATE PROJECT REPORT AND COPYRIGHT

Author's full name : Rawezh Rzgar Karim

Date of Birth : June 30th 2001

Title : I.O.T Doorbell and Security Recognition System

Academic Session : 2023, Semester one.

I declare that this thesis is classified as:

☐ **CONFIDENTIAL** (Contains confidential information under the Official Secret Act 1972)*

☐ **RESTRICTED** (Contains restricted information as specified by the organization where research was done)*

☐ **OPEN ACCESS** I agree that my thesis to be published as online open access (full text)

1. I acknowledged that Universiti Teknologi Malaysia reserves the right as follows:
2. The thesis is the property of Universiti Teknologi Malaysia
3. The Library of Universiti Teknologi Malaysia has the right to make copies for the purpose of research only.
4. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

SIGNATURE OF STUDENT

SIGNATURE OF SUPERVISOR

QU201SCSR001

MATRIX NUMBER

Mr. Kanar Tariq

NAME OF SUPERVISOR

Date: 26 June 2023

Date: 26 June 2023

"I hereby declare that we have read this thesis and in my opinion this thesis is sufficient in term of scope and quality for the award of the degree of Bachelor of Computer Science (Computer Networks & Security)"

Signature : 

Name of Supervisor : Kanar R. Tariq

Date : 26/6/2023

DOORBELL MONITORING SYSTEM BASED ON IOT

RAWEZH RIZGAR KARIM

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Bachelor of Computer Science (Computer Networks & Security)

School of Computing
Faculty of Engineering
Qaiwan International University

June 2023

DECLARATION

I declare that this thesis entitled *"*DOORBELL MONITORING SYSTEM BASED ON IOT *"* is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature : .....

Name : Rawezh Rizgar Karim

Date : 26 JUNE 2023

DEDICATION

This thesis is dedicated to my father, who has always been there for me and dedicated his life to leading me to exellence. Without his understanding, love, patience and hard work. I wouldn't be where I am today.

ABSTRACT

This thesis details the design and implementation of an IoT doorbell device featuring facial recognition technology, developed using the Agile methodology and the Flutter framework. The primary goal is to enhance home security and user convenience through advanced IoT solutions.

The project employed Agile practices to iteratively design, develop, and refine the device, while the Flutter framework provided cross-platform compatibility for a cohesive user interface. Facial recognition capabilities were integrated using a Raspberry Pi, ensuring accurate visitor identification.

The development process included problem identification, a literature review of similar devices, technology stack selection, and iterative prototyping, implementation, and testing. The results show that the device effectively improves home security through reliable facial recognition.

This research contributes to smart home technology by showcasing the practical application of biometric technologies in IoT devices. It demonstrates how such integration can enhance home security systems and improve user experience in smart home solutions.

ABSTRAK

Tesis ini merincikan reka bentuk dan pelaksanaan peranti loceng pintu IoT yang menggunakan teknologi pengecaman wajah, dibangunkan menggunakan metodologi Agile dan rangka kerja Flutter. Matlamat utama adalah untuk meningkatkan keselamatan rumah dan kemudahan pengguna melalui penyelesaian IoT yang maju.

Projek ini menggunakan amalan Agile untuk merekabentuk, membangunkan, dan memperhalusi peranti secara iteratif, manakala rangka kerja Flutter menyediakan keserasian merentas platform untuk antara muka pengguna yang menyeluruh. Keupayaan pengecaman wajah diintegrasikan menggunakan Raspberry Pi, memastikan pengenalan pengunjung yang tepat.

Proses pembangunan termasuk pengenalpastian masalah, tinjauan literatur mengenai peranti serupa, pemilihan teknologi, serta prototaip, pelaksanaan, dan ujian iteratif. Keputusan menunjukkan bahawa peranti ini berkesan meningkatkan keselamatan rumah melalui pengecaman wajah yang boleh dipercayai.

Penyelidikan ini menyumbang kepada teknologi rumah pintar dengan menunjukkan aplikasi praktikal teknologi biometrik dalam peranti IoT. Ia menunjukkan bagaimana integrasi sedemikian dapat meningkatkan sistem keselamatan rumah dan memperbaiki pengalaman pengguna dalam penyelesaian rumah pintar.

LIST OF TABLES

TABLE NO.	TITLE	PAGE
Table 1.1	Table showing 2nd generation doorbell specs	13
Table 1.2	Comparison table showing differences between Eufy and Ring	11.
Table 1.3	Table showing advatages and disadvantages of agile **Error! Bookmark not defined.**	
Table 1.4	Raspberri Pi, hardware specs	24

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
Figure 2.2.1.1	Ring 2nd Generation device	7
Figure 2.2.1.2	Ring app interface	8
Figure 2.2.2.1	Eufy S330, best selling model	9
Figure 2.2.1.1	Eufy phone app interface	10
Figure 2.5.1	Gantt Chart	14
Figure 3.3.1	Agile Model	19
Figure 4.1	Use case diagram	26

Contents

DECLARATION ii

DEDICATION iii

ABSTRACT v

ABSTRAK vi

LIST OF TABLES vii

LIST OF FIGURES viii

Chapter One 1

1.1 Introduction 1

1.2 Problem Background 2

1.3 Project Aim 2

1.4 Project Objectives 3

1.5 Project Scope 4

1.6 Project Importance 5

1.7 Report Organization 5

CHAPTER 2 LITERATURE REVIEW 6

2.1 Introduction 6

2.2 Case Studies 6

2.2.1 Ring Doorbell 7

2.2.2 Eufy Doorbell 9

2.3 Current System Analysis 10

2.4 Comparison between existing systems 11

2.5 Literature review of technology used 11

2.6 PSM Gantt Chart 14

2.7 Chapter Summary 15

CHAPTER 3 SYSTEM DEVELOPMENT METHODOLOGY 16
3.1 Introduction 16
3.2 Methodology Choice and Justification 17
3.3 Phases of the Chosen Methodology 19
3.4 Technology Used Description 21
3.5 System Requirement Analysis 23
3.6 Chapter Summary 24
CHAPTER 4 REQUIREMENT ANALYSIS AND DESIGN 25
4.1 Introduction 25
4.2 Requirement Analysis 25
4.2.1 Requirement Analysis 26
4.2.2 Actor Description 27
4.2.3 Use Case Description 28
4.2.4 Sequence Diagrams 28
4.2.5 Requirement Analysis 35
4.3 Project Design 36
4.3.1 System Architecture Design 36
4.4 Interface Design 37
4.4.1 Live Feed Page 38
4.4.2 Storage Management Page 39
4.5 Chapter Summary 39
CHAPTER 5 IMPLEMENTATION AND TESTING 40
5.1 Introduction 40
5.2 Coding of System Main Functions 40
5.3 Testing 41

5.3.1 Black box Testing 41
5.4 Chapter Summary 43
CHAPTER 6 CONCLUSION 44
6.1 Introduction 44
6.2 Achievement of Project Objectives 44
6.3 Suggestions for Future Improvement 44

CHAPTER 1

INTRODUCTION

1.1 Introduction

Everyone has the right to feel secure in their home. It's nice to have peace of mind as a homeowner and relax knowing that your house is secure. Most of the time, just having a visible working camera can be a deterrent to any malicious actors who want to conduct a burglary, or home invasion. Unfortunately, installing security systems can get pricy and technical. In terms of cost, home security can be expensive to get into, and often require you to subscribe to a security company's services, which is not cheap. When it comes to technicality, it will take a lot of time an expertise to properly set up a camera system.

Thanks to these two factors, not everybody will be able to afford comprehensive security solutions, making home security unapproachable to the average person. This is harmful, as depending on where a person lives, home security might be a very essential requirement to achieve peace of mind.

This project aims to provide an inexpensive security device with I.O.T capabilities. The device is a doorbell that automatically detects movement at the user's front door and sends an image of the person to the user. The user then can approve entry or ignore their visitor depending on whether or not they were expecting them or not. The doorbell can also be used detect trespassers or package thieves. As well as a front door security camera.

1.2 Problem Background

Many people do not feel like they're protected and secure inside their homes, or they are afraid to leave their house unattended for fear of someone breaking in. This is inconvenient and anxiety inducing in citizens. The best recourse against home invasions is to install a security system, unfortunately they are quite pricy to setup and require a lot of technical skill that the average person does not have, making them inaccessible to most of the population.

This inaccessibility leaves many people vulnerable to crimes such as home invasions, burglary, and package theft. As most times, having a visible security device can act as a deterrent against such crimes. A package thief will not attempt to steal a package on a homeowner's front door if there was blatantly a security camera there, for example. However, as we discussed, current security solutions are both expensive, and technically challenging to setup, making this deterrent out of reach of people who don't have the means to afford them

1.3 Project Aim

Create an easy to use I.O.T security device, accessible to the average person.

1.4 Project Objectives

The objectives of the project are:

1. Building an affordable I.O.T solution to home security using Raspberry PI computer, that is affordable and easy to set up

2. Conduct preliminary research on components and how to best integrate them into the project, such as the tech stack, necessary hardware components, and specialized software as well as their compatibility with each other. Determine the functional requirements and system requirements of the system, as well as the user requirements.

3. Create an easy to use mobile app that allows the user to control the device. The aim of the application is to give the user a smooth and easy UX, that is convenient to use with the aim of easily setting up the security device, without much tech expertise of background

4. Conduct thorough testing of various system and device functionalities, such as the mobile app, and system features such as motion detection, and cloud storage. Blackbox and Whitebox testing should be done to ensure the system is working as intended and is not frustrating to use by the target user.

1.5 Project Scope

The scopes of the project are:

1. **I.O.T connectivity:** The device can connect to other devices on the internet, such as the user's smartphone.
2. **User Authentication:** The system must authenticate the user(s) devices before they can connect and use the device.
3. **Video capturing:** The system can capture a live feed through its Camera, the user can select whether or not the device will continuously record or record based on motion detection.
4. **Motion detection:** The system is capable of motion detection, and will begin recording if it detects any motion within a particular range of the monitoring device, as well as send an alert to the user through the mobile app. This is done through motion sensors.
5. **Video and voice transmission:** The system can transmit the feed through the internet to the user's smart device, and this can be used to conduct video-calls alongside voice transmission. This is done through a mobile app that that allows the user to control the device's various features including voice calls.

The combination of motion detection and voice transmission can help deter potential threats to the homeowner's property. In the event of an attempted break in, the home owner can transmit his voice to the device and inform the criminals that the authorities have been called and are on their way.

1.6 Project Importance

People find it difficult to setup home security systems and devices to feel secure in their own home. This is due to the inaccessibility of current home security solutions to the average person, due to expense and technical expertise. This is inconvenient to people who want a simple home security solution, without much of an aptitude for technology or network solutions.

It is much easier to have an easy to setup, set and forget device than can provide home security than to go through the complicated process of finding a home security service provider to set up a network of cameras. With the I.O.T device paired with an app, all the user need do is to install the device, and configure it through a variety of easy to understand settings on their phone through an application's interface.

1.7 Report Organization

- **Chapter One:** Chapter one is an introduction and general discussion of the problem to discuss the benefits and why there is a need for the project.
- **Chapter two:** Literature review
- **Chapter Three:** Approach, tech-stack and framework of the project
- **Chapter Four:** System requirements analysis, and system design**.**
- **Chapter Five:** Implementation and testing
- **Chapter Six:** Conclusion

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, we conduct a literature review to find out about the inner workings of existing systems, as well as current technologies used in similar I.O.T tech solutions. Preliminary-research is an important element of project to find out what tech stacks and devices are best to pull-of a similar project. In this chapter, we'll also analyse existing I.O.T Doorbell security systems to find out how they compare and what features they have that are useful to the modern user.

2.2 Case Studies

For the case study, we'll be examining two I.O.T doorbell security devices, Ring Doorbell and Eufy Doorbell. We'll be comparing their specs, features, pricing, and mobile apps respectively to get a better idea on how they work.

2.2.1 Ring Doorbell

The Ring Doorbell is a smart home security device that consists of a doorbell with a camera, and two-way communication between the visitor and homeowner. It also allows homeowners to see visitors remotely through a mobile app. The device has many other features depending on the model, which includes motion detection, cloud storage, allowing for the device to be stolen but for the footage to remain intact, smart home integration through devices like Amazon's Alexa, allowing the use of voice commands with the device, as well as other advanced features such as setting a field of motion for motion detected, night vision and so on. The primary method for controlling the doorbell is through mobile app that can be used to be paired with the device.

Figure 2.2.1.1, showing the 2nd generation Ring doorbell device

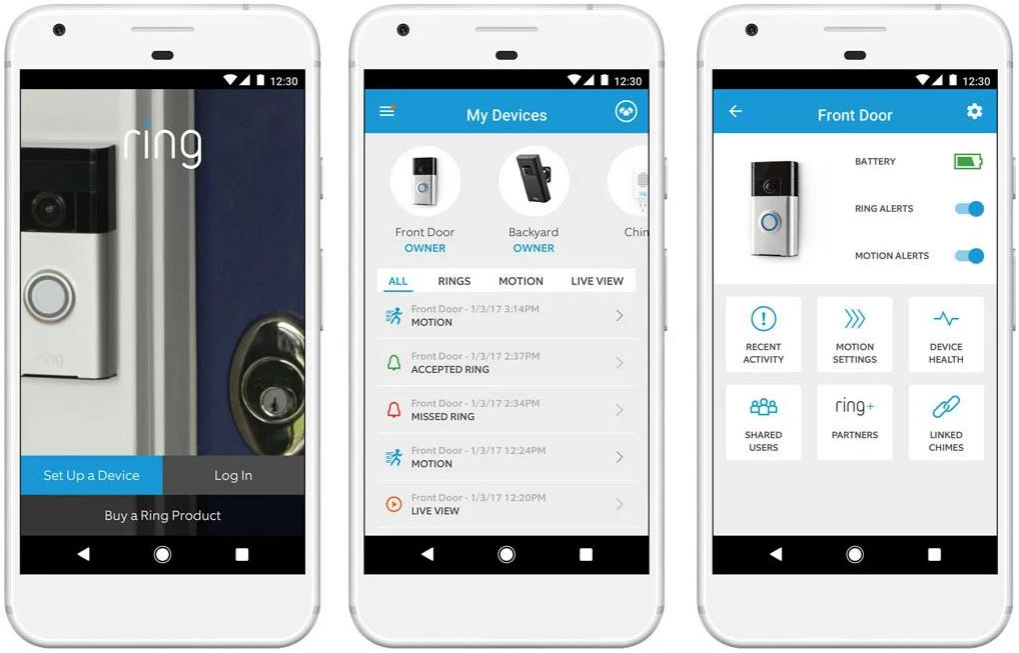

Figure 2.2.1.2, showing the Ring app, device pairing and controls.

Table 1.1 showing specs of the 2nd generation doorbell.

Ring, 2nd Generation doorbell Specs.	
Camera resolution	1080p HD
Field of view	155 degrees horizontal, 90 degrees vertical
Power options	Rechargeable battery (non-removable), hardwired to existing doorbell or transformer, solar charger or connected to mains via Plug-In Adapter (2nd Gen)
Connectivity	2.4GHz Wireless 802.11 b/g/n
Connection speed	2mbps
Audio specs	Two-way audio with noise cancellation
System requirements	Current system requirements for the Ring app
Size	12.6 cm x 6.2 cm x 2.8 cm

2.2.2 Eufy Doorbell

Eufy doorbell also boasts a lot of similar features to the ring doorbell device. With it having night vision, smart home integration, two-way communication between the homeowner and visitors and so on. However, there are some differences between it and its main competitor. Eufy doorbell, in addition to offering cloud storage as a subscription service, also offers local storage. While this might not be ideal if you fear that the footage from your device will get stolen alongside the device, it will help on saving costs compared to ring doorbells, as there is no need to pay a for a cloud subscription if you don't have the means to. Eufy doorbell pricing is also overall cheaper than ring devices, although this will differ from model to model.

Figure 2.2.2.1, showing S330, the bestselling Eufy doorbell model.

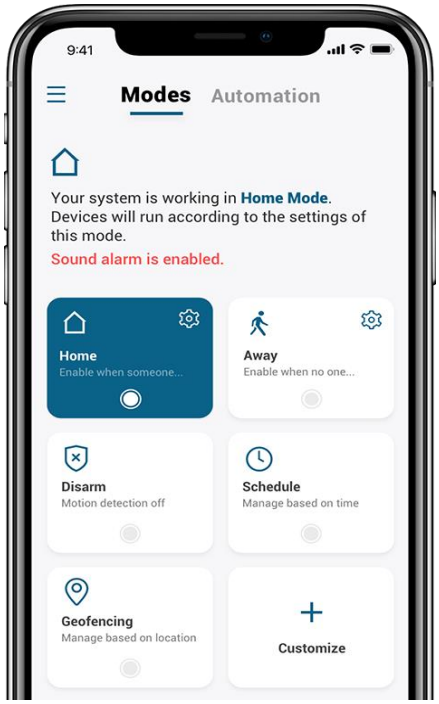

Figure 2.2.2.2, showing Eufy phone app.

2.3 Current System Analysis

Analysis of current systems reveals that there a bunch of key features that users look for when it comes to these types of devices. These features all combine to make an affordable security solution that is easy to install and to use.

The device should be easy to setup and not require a technician to mount for home security. The primary point of these devices is accessibility for the common user, meaning that they should work out of the box and not be overly complicated. After installing and turning on the device, the UX should be smooth and not demand too much from the end user. The device should come with a mobile app that allows for easy pairing with the device and an easy to navigate menu to control the device's features, such as motion detection, cloud storage, reviewing past footage and so on.

People on the market for such solutions do not desire an expensive and overly complicated device, and are not tech savvy enough to install their own security systems, with multiple cameras and monitors which will cost a lot of money.

2.4 Comparison between existing systems

In the table below, we'll compare the features of both existing I.O.T Doorbell security solutions.

Table 1.2 showing comparison between Eufy and Ring

Characteristics	**Eufy**	**Ring**
Cloud Storage	√	√
Motion Detection	√	√
Local Storage	√	X
Mobile app pairing	√	√
Subscription model	√	√

The primary difference besides pricing between Eufy and ring is the subscription model and local storage. Thanks to Eufy's built in local storage, users are able to review captured footage without the need of a cloud subscription. However, this feature is not present in ring doorbells, meaning the only way to review old footage or store footage is through paying a monthly subscription to the company, which is a major inconvenience to users who desire more control over their security device.

2.5 Literature review of technology used

The project will have both a Raspberry Pi programmed as the IOT device, as well as a mobile application to go along with it that the user will use to control the device. The various technologies used for development and deployment are as follows.

- **Flutter**: Flutter, an open-source UI framework developed by Google, enables developers to create natively built software for mobile, web, and desktop platforms using a unified codebase. Leveraging the Dart programming language, programmers can design responsive, aesthetically pleasing, and high-performance applications. Flutter offers an extensive selection of pre-made widgets and components, facilitating swift UI development while ensuring a native experience on each platform. The framework's focus on expressive and customizable UIs guarantees visually pleasing apps for a better UX, while its compilation to native code ensures excellent performance and seamless integration with the respective systems. Thanks to its versatility, Flutter is now a top choice for modern app development across various devices and platforms.

- **Dart**: Flutter utilizes the Dart programming language as its core for app development, introducing distinctive features while incorporating syntax elements from Java, JavaScript, and C#. Dart's fully object-oriented nature enables the use of essential concepts such as classes, inheritance, and interfaces, promoting a modular and reusable approach to code organization. Moreover, Dart supports both Just-in-Time (JIT) and Ahead-of-Time (AOT) compilation, facilitating fast development with hot reload and improved performance for production releases. With its strong type system, Flutter developers benefit from enhanced code reliability, readability, and maintainability.

- **Visual Studio Code:** Microsoft develops Visual Studio Code, an IDE that enhances productivity through various tools like syntax highlighting, code completion, formatting, and navigation. It intelligently adapts suggestions to the programming language used, providing a flexible environment tailored to user preferences. Visual Studio Code offers extensive customization and extensibility options, boasting a considerable collection of community-contributed extensions. These extensions introduce new functionalities, support diverse languages, add themes, and integrate with various tools and frameworks. Developers (Cont.)

- can install and personalize these extensions according to their specific needs, allowing them to optimize and customize their coding experience.

  Additionally, Visual Studio Code boasts a user-friendly interface that caters to both beginners and experienced developers alike. Its intuitive layout and ease of use make it a popular choice among programmers across different skill levels. The IDE's built-in version control integration, such as Git, facilitates seamless collaboration and code management, further streamlining the development process.

- **Raspberry PI:** The Raspberry Pi Foundation is the mastermind behind Raspberry Pi, a series of single-board computers (SBCs). These credit card-sized, affordable computers are versatile and can be employed for numerous tasks and programs. Running on a specialized version of Linux, they can be tailored to fulfill a wide range of functions. The primary programming language for Raspberry Pi is Python, though it supports the use of other programming languages as well.

  Due to its flexibility and programmability, the Raspberry Pi has become a popular choice for building prototypes and proof-of-concept projects. It also serves as an energy-efficient solution for internet-of-things (IoT) applications. Additionally, the Raspberry Pi's GPIO (General Purpose Input/Output) pins allow it to interact with external hardware, making it a great choice for projects that require sensor integration, robotics, and automation.

- **Firebase:** Firebase is a platform developed by Google that offers a comprehensive set of tools and services for building web and mobile applications. It provides developers with easy-to-use backend services, including real-time database, authentication, cloud storage, hosting, and cloud messaging. With Firebase, developers can focus on app development without worrying about infrastructure management. Moreover, Firebase is known for its real-time capabilities, making it ideal for building collaborative and interactive applications. With the Realtime Database and Cloud Firestore, developers can create real-time synchronization between clients and the backend, facilitating instant updates and data sharing across devices.

2.6 PSM Gantt Chart

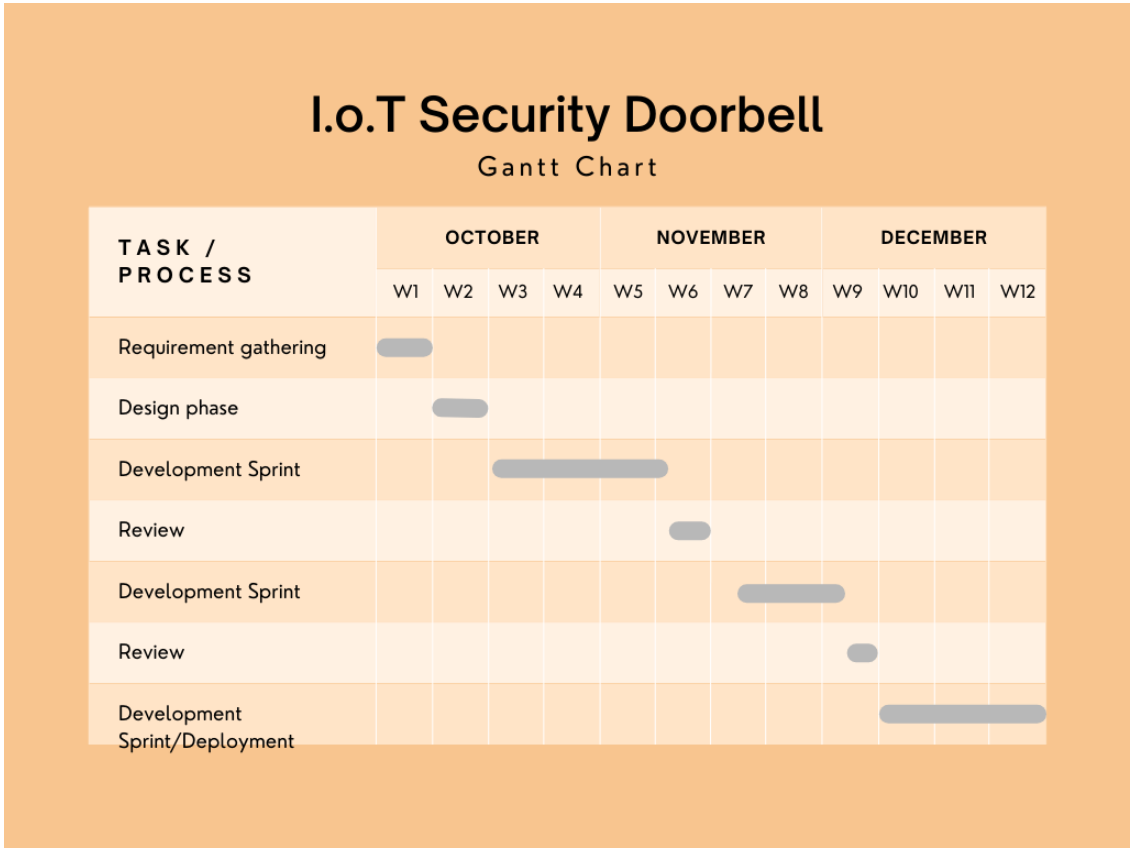

Figure 2.5.1, Gantt Chart

2.7 Chapter Summary

In this chapter, we explored current I.O.T security doorbell solutions and their features, to help gather some requirements for own device, and learn what features customers will expect from the security solution. After comparison the two dominating companies in the market, them being Ring and Eufy, and comparing their features, we found that users will expect an easy to install security solution that does not require much hassle, and is relatively inexpensive compared to setting up a whole network of security cameras for example.

CHAPTER 3

SYSTEM DEVELOPMENT METHODOLOGY

3.1 Introduction

Software development methodologies consists of a systematic approach used to dictate the development of software. The methodology will dictate how long the project will take, which features are able to be delivered, what the lifecycle of development and sometimes maintenance will be. Each methodology has its own advantages and disadvantages, as they each specialize in a certain type of software development.

A well-defined methodology gives guidelines that will help with productivity, meeting deadlines, teamwork and communication, as well as help with any issues that arise with development. Some methodologies like waterfall are rigid, with clear deadlines to meet are better for longer term development with fixed software, while others focus on continuous improvement and fast delivery and deployment of systems such as agile.

This chapter will discuss the chosen development methodology, the justification for choosing it, its advantages and disadvantages as well as the phases used for development. We'll also discuss all tools and the tech stack used for the project, and explain what they are and what they're for in detail.

3.2 Methodology Choice and Justification

For this project, I have chosen the agile methodology for system development. Agile development is a type of iterative software development that is flexible and focuses on fast delivery and deployment of projects. Rigid methodologies like waterfall, have the development requirements collected and defined upfront, which can be advantageous for a project that will take a long time, agile however is not like this.

In Agile development, the development process has multiple iterations. Agile encourages incremental development of system, and contentiously collects feedback to change or add extra requirements during the development cycle. It's a very flexible methodology for fast development of systems. The development iterations in agile are called "Sprints". Each sprint typically takes multiple weeks and the sprint focuses on delivering part of the requirements. Before a new sprint begins, the development team will meet with the stakeholders to define what the requirements for the sprint will be, as well as which requirements get priority, this requirement list is called a "backlog".

During each sprint, the development team will have daily meetings to discuss their progress, any issues that have risen. Even if all the requirements aren't met, it will just get postponed to the next sprint, and at the end of each sprint a portion of the project will be completed and sent for review by the stakeholders, which might lead to changes or additional requirements for the next sprint. Overall, the main focus of agile development is flexibility, continuous communication and collaboration, focus of stakeholders, and dynamic and speedy delivery of software project, with ever changing requirements.

3.2.1 Advantages and disadvantages of agile

Agile has many advantages, especially when it comes to flexibility and speedy delivery. However, it is not without disadvantages either. We'll discuss the advantages and disadvantages of agile below.

Table 1.3 showing advantages and disadvantages of agile.

Advantages	Disadvantages
1) Flexibility: Agile does not define rigid, unchanging requirements for its projects. Which allows the requirements to change during the development process to adapt to customer needs. 2) Stakeholder collaboration: Agile is based on close collaboration with stakeholders during system development, which helps more accurately deliver a product within customer expectations. 3) Quick system delivery: Agile development delivers the project in increments, allowing for earlier deployment and releasing of the product. Even if the product is missing some features, they continuously improve it through incremental refinements.	1) Unpredictability: The requirements for agile projects change or increase with each increment, making it difficult to define an exact timeline for complete delivery. 2) Scope creep: The flexibility of requirements might lead to scope creep, which might end up causing an issue with things like the budget and allocated resources. 3) Demand for customer availability: Agile development methodologies need constant feedback from stakeholders, which might get too demanding or become unfeasible for the customer. This is an issue especially with customers who are geographically apart from the development tea, due to time zones.

3.3 Phases of the Chosen Methodology

Agile development phases typically follow the same structure, as shown in the graph below.

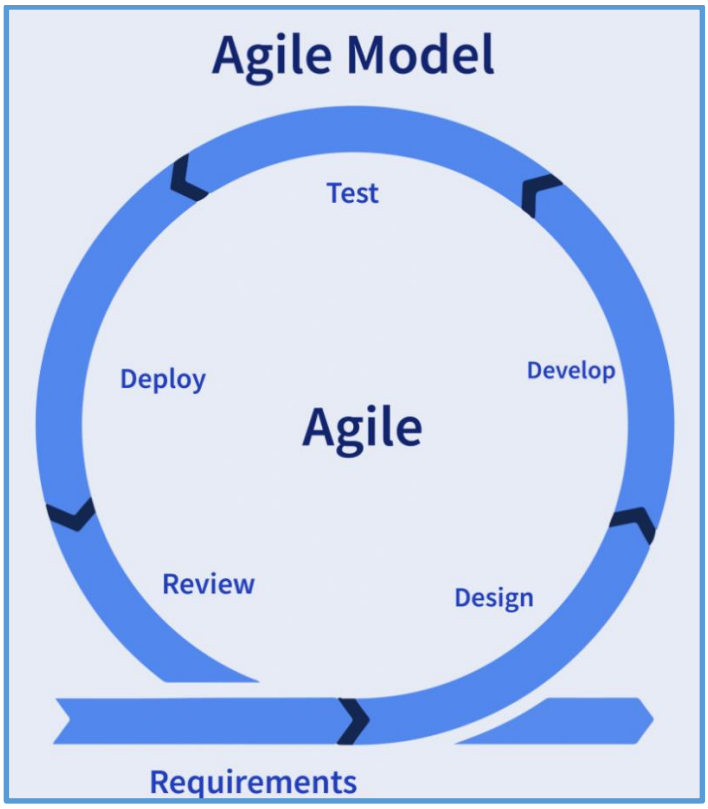

Figure 3.3.1, Showing agile model.

- **Requirements:** During this phase, the development team will gather the requirements with the stake holders. During this phase, developers will understand the needs and expectations that the stakeholders have for the project, this phase will repeat itself with each development increment. The aim of this phase will be to define the user needs as clear as possible, to focus on later during development

Requirement gathering for the project will be done through interviews and user stories with stakeholders, this is to know what the stakeholder expectation will be, and what to focus on during development. In the requirements phase, I'll also examine and define several use cases of the system, to clearly define functional requirements and for testing later.

- **Design:** Designing a software solution's design and architectural framework is the primary goal of the design phase in agile development, In Agile, the design phase is integrated into all other phases of the development lifecycle rather than existing as a separate and independent stage. Meaning that his phase will also repeat with each increment.

  In this phase, I'll make clear which features will take priority, and which features will make it into the development phase. This phase will repeat after each sprint to gather more requirements, and stakeholder feedback until the software requirements meet stakeholder expectations.

- **Development:** During the development phase, the requirements gathered in the previous phases that were added to backlog will get developed, iteration by iteration. This helps with continuously improving the software for the stakeholders, as later on after delivery, it will get reviewed and evaluated by the stakeholders. This phase will last 1-4 week(s), after which I will present my workings for review for the stakeholders, whether to improve the current system or to gather more requirements for the next development phase.

- **Testing:** Agile encourages early and continuous testing, which means that testing is integrated into each sprint to ensure that issues will get identified and rectified quickly, which will help satisfy the stakeholders while ensuring timely delivery.

  In the testing phase, each feature developed in the development phase will get tested by me using blackbox and whitebox testing methods. After ensuring that the added features during the development phase are working as intended, we can

then move on to the review phase, or if the project is ready for deployment, the deployment phase.

- **Deployment:** After the development software meets a sufficient amount of requirements, it will get deployed to be used by the stakeholders. Even if all the requirements are not met, the software will still get continually improved upon via further development iterations.

- **Review:** At the end of each development cycle, the developer must hold a review with the stakeholders to present and discuss the work developed and delivered during the sprint. The development team will show the stakeholder which functionalities have been fulfilled by the requirements and collect feedback and validate if the work is satisfactory. This process will repeat itself multiple time before the system enters the deployment phase, and even after the deployment phase. This phase will be done by presenting my work to the stakeholders.

3.4 Technology Used Description

The project will have both a Raspberry Pi programmed as the IOT device, as well as a mobile application to go along with it that the user will use to control the device. The various technologies used for development and deployment are as follows.

- **Flutter**: Flutter is a Google open-source UI (User Interface) framework. It allows developers to produce natively built software from a single codebase for mobile, web, and desktop platforms. It enables programmers to create responsive, aesthetically pleasing, high-performance applications using the Dart programming language. Flutter offers a comprehensive selection of pre-made widgets and components that facilitate quick UI development. For a native experience on each platform.
- **Dart**: Flutter uses the programming language Dart. It serves as the main programming language for creating applications with the Flutter framework. Dart introduces its own distinctive features while combining syntax from (cont.)

languages such as Java, JavaScript, and C#. Everything in Dart is an object because it is a fully object-oriented language. It is compatible with object-oriented development theory and concepts such as classes, inheritance, interfaces etc. This facilitates the modular and reusable organization and structure of code.

- **Visual Studio Code:** Visual studio code is an IDE made by Microsoft. It includes tools that increase productivity, including as syntax highlighting, code completion, formatting, and navigation. Based on the programming language, it makes intelligent suggestions and offers a flexible and adaptable environment to accommodate user preferences. Numerous customization and extensibility options are available in Visual Studio Code. It features a sizable library of community-contributed extensions that add new functionalities, support other languages, add themes, and integrate with different tools and frameworks. Installing and customizing these extensions to suit their individual requirements allows developers to improve and customize their coding experience.

- **Raspberry PI:** The Raspberry Pi Foundation created a line of single-board computers (SBCs) called Raspberry Pi. It is a little, reasonably priced computer the size of a credit card that can be used for many different tasks and programs. It runs on its own particular version of Linux and may be designed to perform a wide range of jobs. Its primary programming language is Python, though it can also be used with other languages.

- **Firebase:** Firebase is a platform developed by Google that offers a comprehensive set of tools and services for building web and mobile applications. It provides developers with easy-to-use backend services, including real-time database, authentication, cloud storage, hosting, and cloud messaging. With Firebase, developers can focus on app development without worrying about infrastructure management.

3.5 System Requirement Analysis

The system will be able to run on any contemporary smartphone to allow it to be accessible to every user, regardless of whether or not they own a top of line phone. The unique hardware requirements for the project is the Raspberry pi device.

Table 1.4 showing specifications of Raspberry Pi B

NO	Hardware	Specification
1	RAM	4GB
2	Operating system	Raspberry-pi OS (Linux distribution)
3	Network Connection	• 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, • Bluetooth 5.0 • BLE Gigabit Ethernet
4	Processor	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.8GHz

3.6 Chapter Summary

In this chapter, we went through the chosen development methodology, why development methodologies are important, the characteristics of agile development, including its six main phases and their details. As well went through the technology stack that will be used for project development as well as the system requirements. For the backend solution, we picked firebase, as it is easy to manage due to the lack of infrastructure needed by developers, and we picked flutter for the mobile app development as well as raspberry pi to program the I.O.T

CHAPTER 4

REQUIREMENT ANALYSIS AND DESIGN

4.1 Introduction

This chapter analyzes the system design of the project based on the gathered requirements; It consists of describing the workflow of requirements analysis results and design phase. This chapter will contain the use case diagram, activity diagrams, and database design using UML modeling. As well as the interface design

4.2 Requirement Analysis

The requirement analysis will be done through multiple UML diagrams that will cover the use cases that the user will go through when using the program, as well as cover the back-end and front-end design of the project via UML design of the database, and interface design from the front end parts of the project that the user will interact with.

4.2.1 Requirement Analysis

Use case diagram summarizes some of the relationships between use cases, and the user (Actor). Actors are the users which will use the functionality of the system. While the use cases are the functionalities of the system. The below use case diagram will showcase both:

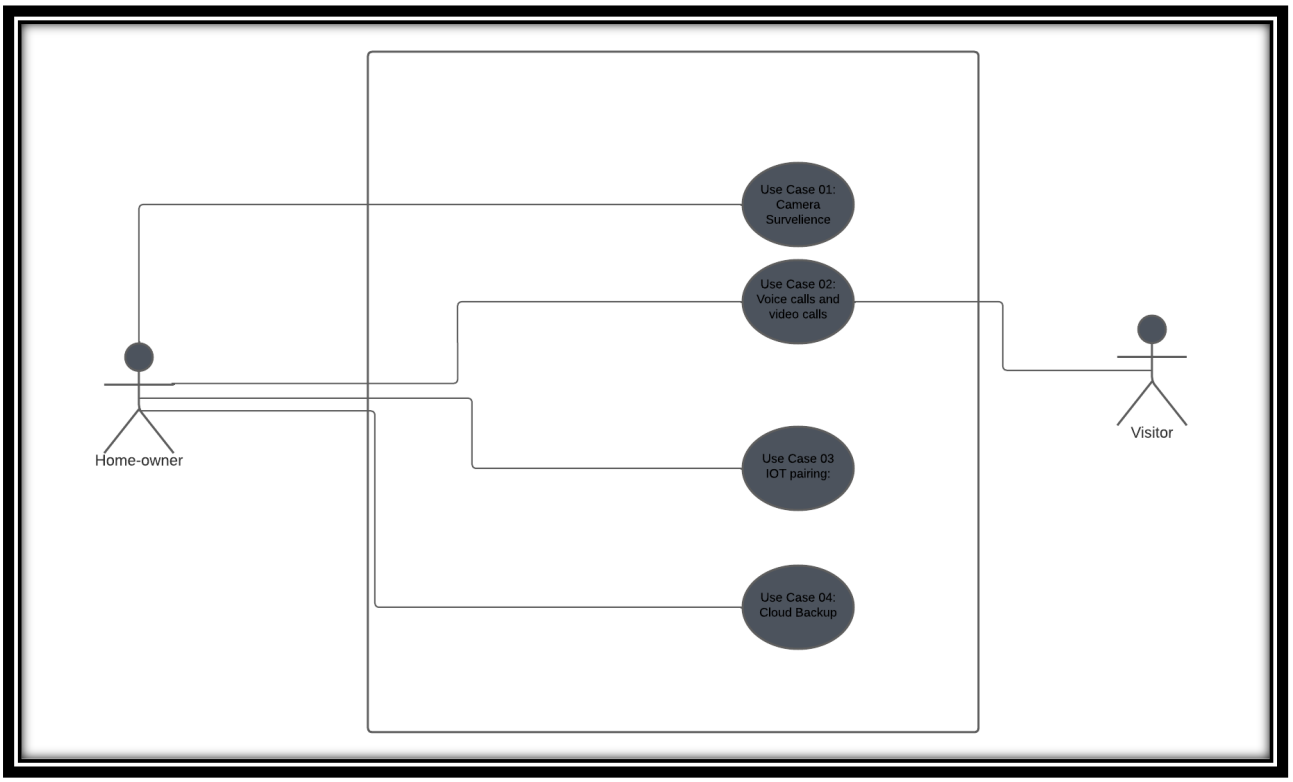

Figure 4.1 showing use case diagram

- **Camera surveillance:** The user will be able to use the device as a surveillance device of wherever it is deployed. Allowing the user to view live footage as well as backed up footage in the cloud to see the goings on of where the user deployed the device. The device will allow for motion detection video capturing as well as a configuration for live feed video capture and streaming.
- **Voice calls:** The user will be able to transmit their voice to the IOT device through the internet, if they have a working internet connection. This allows for remote communication with intended and unintended guests, a user can tell a potential robber that they are being recorded for example, which might deter them(cont.)

from breaking into the home. Or, the user might be expecting someone and will check if the intended person(s) are the ones who are at the door.

- **IOT pairing:** The user will be able to pair the I.O.T device with their phone to allow for remote viewing wherever the user will be, as well as to allow access to the backup footage from the device stored in the cloud. The pairing process will be intuitive and easy to understand for the user.

- **Cloud backup:** The user will be able to manage the device's cloud backup and view the captured footage whenever they wish, as well all delete backups or configure how backups are taken or set a temporary timer on them to be delete after a certain time has passed. This way the user will have full control over the backup functionality of the device.

4.2.2 Actor Description

According to the use case diagram, two main actors will use the system, and we'll discuss how they interact with the system and each other

Table 4.2.1.1 Actor Description

No	Actor	Role
1	Home Owner	Home Owners will use the system to view the live camera feed, interact with visitors, and manage cloud storage.
2	Visitor	Visitors will use the system to interact with homeowners.

4.2.3 Use Case Description

The actors of the system can perform various functionalities. Table 4.2.1.2 Use Case Description describes the use cases of the doorbell system based on the actor use cases.

Table 4.2.1.2 Use Case Description

Actor	Use Case	Description
Home-Owner	Register	To allow home owners to register to authenticate their mobile device.
	Login	To allow home owners to log into the system.
	Surveillance	To allow home owners to view the live feed of the doorbell camera
	View and manage stored videos	To allow homeowners to view the stored videos in the cloud system, as well as delete them.
	Voice over call	To allow homeowners to speak to the visitors over the internet
Visitor	Voice over call	To allow visitors to speak to the homeowners over the internet, with the homeowner's permission.

4.2.4 Sequence Diagrams

Sequence diagrams are interaction diagrams that show how the system activities, processes, or tasks are being carried out for each function. The sequence diagram also shows in detail how each function work.

4.2.4.1 Register Sequence Diagram

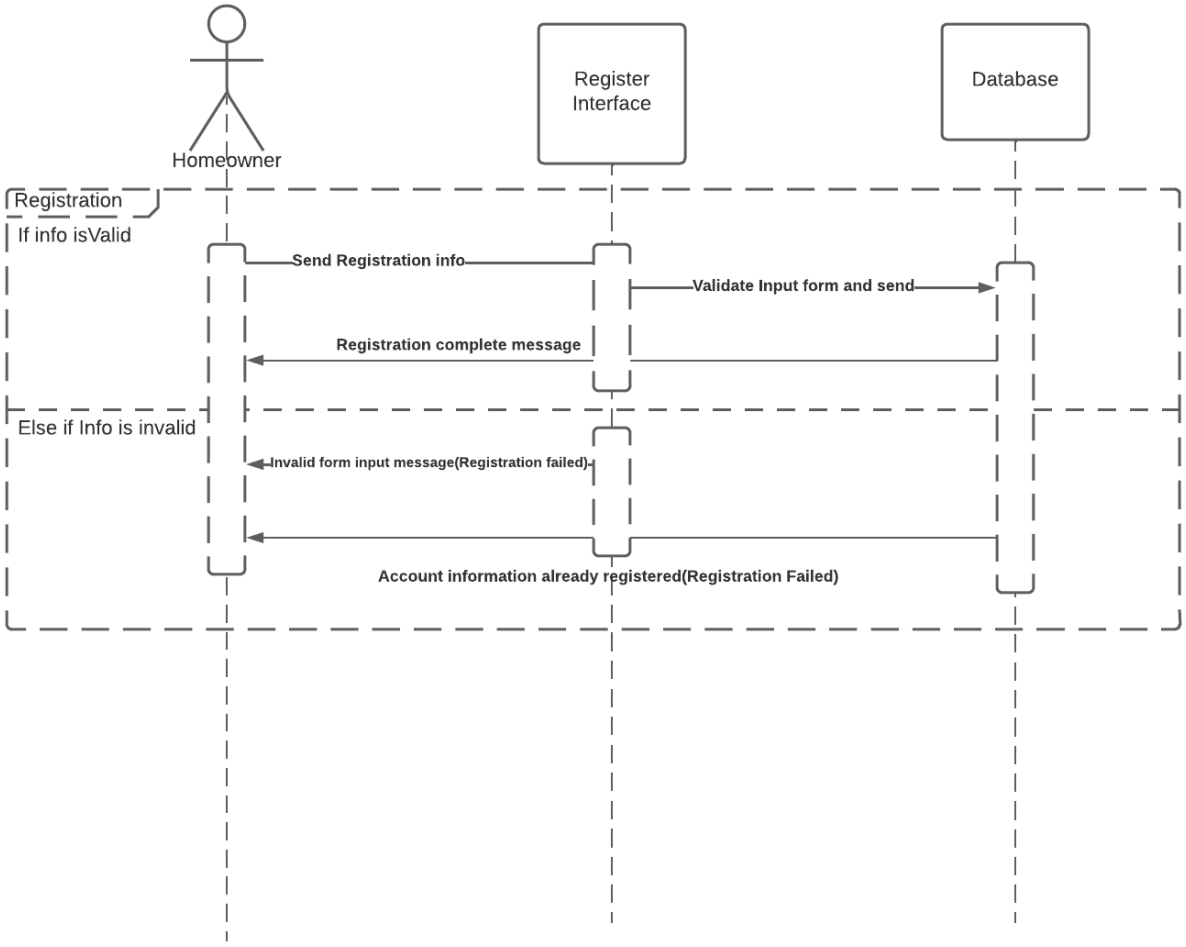

Figure 4.2.4.1 (Registration Sequence diagram)

4.2.4.2 Login Sequence Diagram

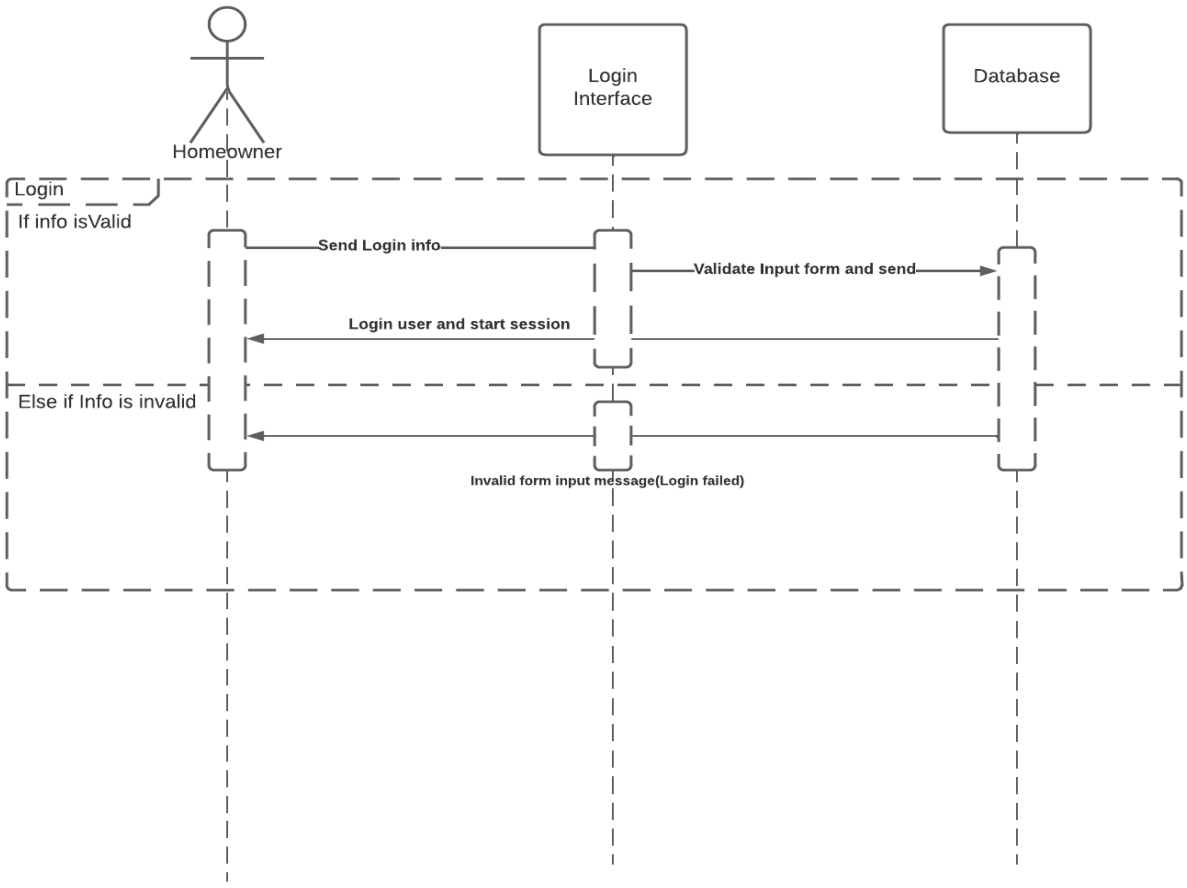

Figure 4.2.4.2 (Login Sequence diagram)

4.2.4.3 Live Surveillance Sequence Diagram

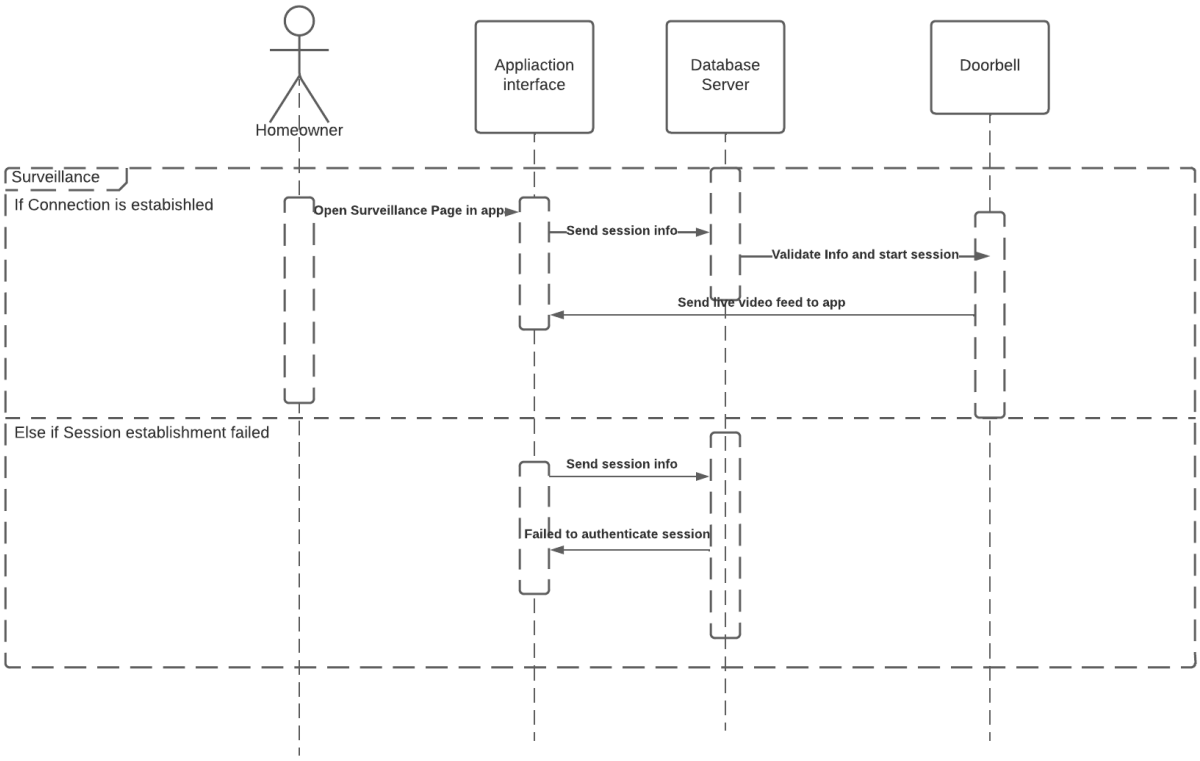

Figure 4.2.4.3 (Surveillance Sequence diagram)

4.2.4.4 Video Playback and Storage Management Sequence Diagram

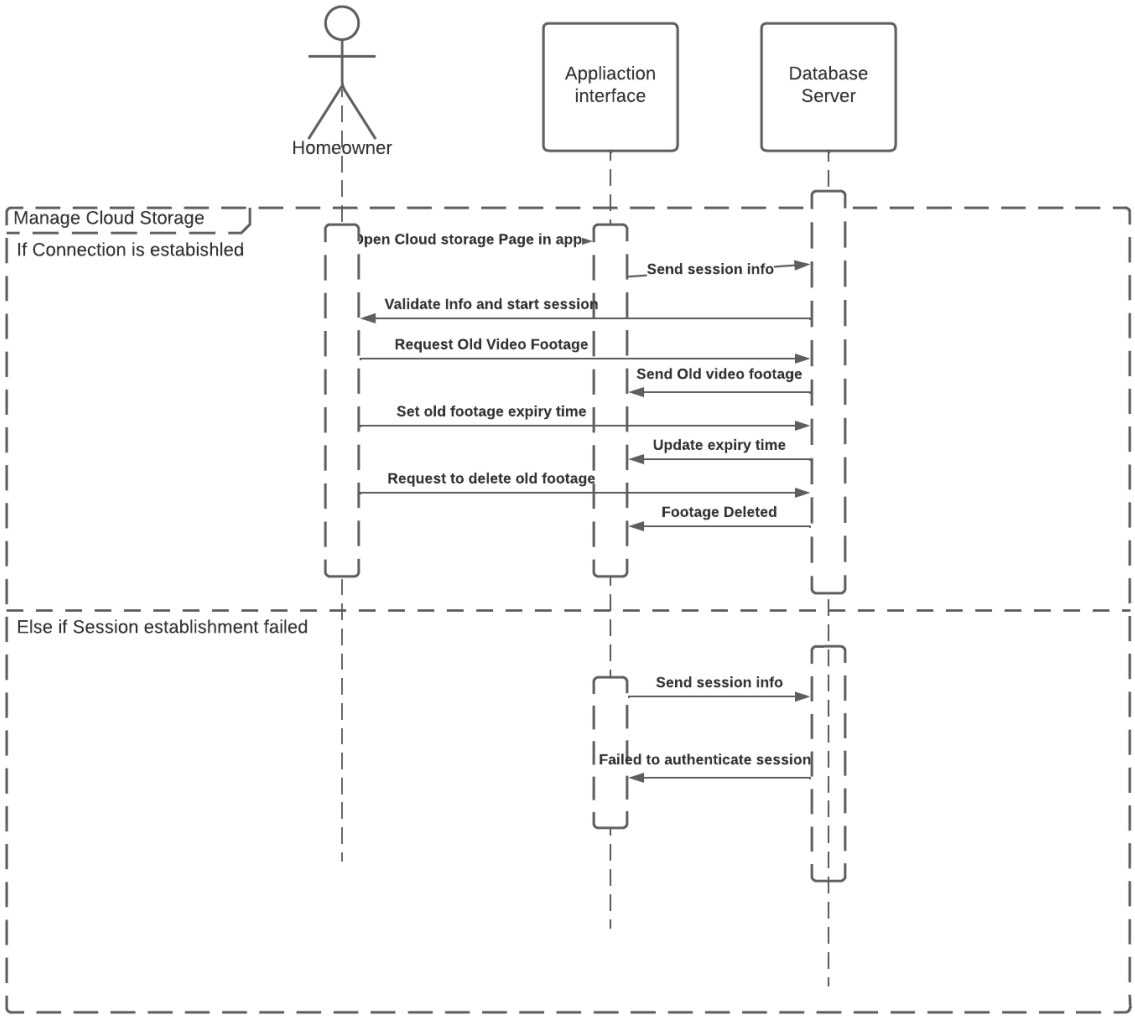

Figure 4.2.4.4 (Playback management Sequence diagram)

4.2.4.5 Voice Calls Sequence Diagram

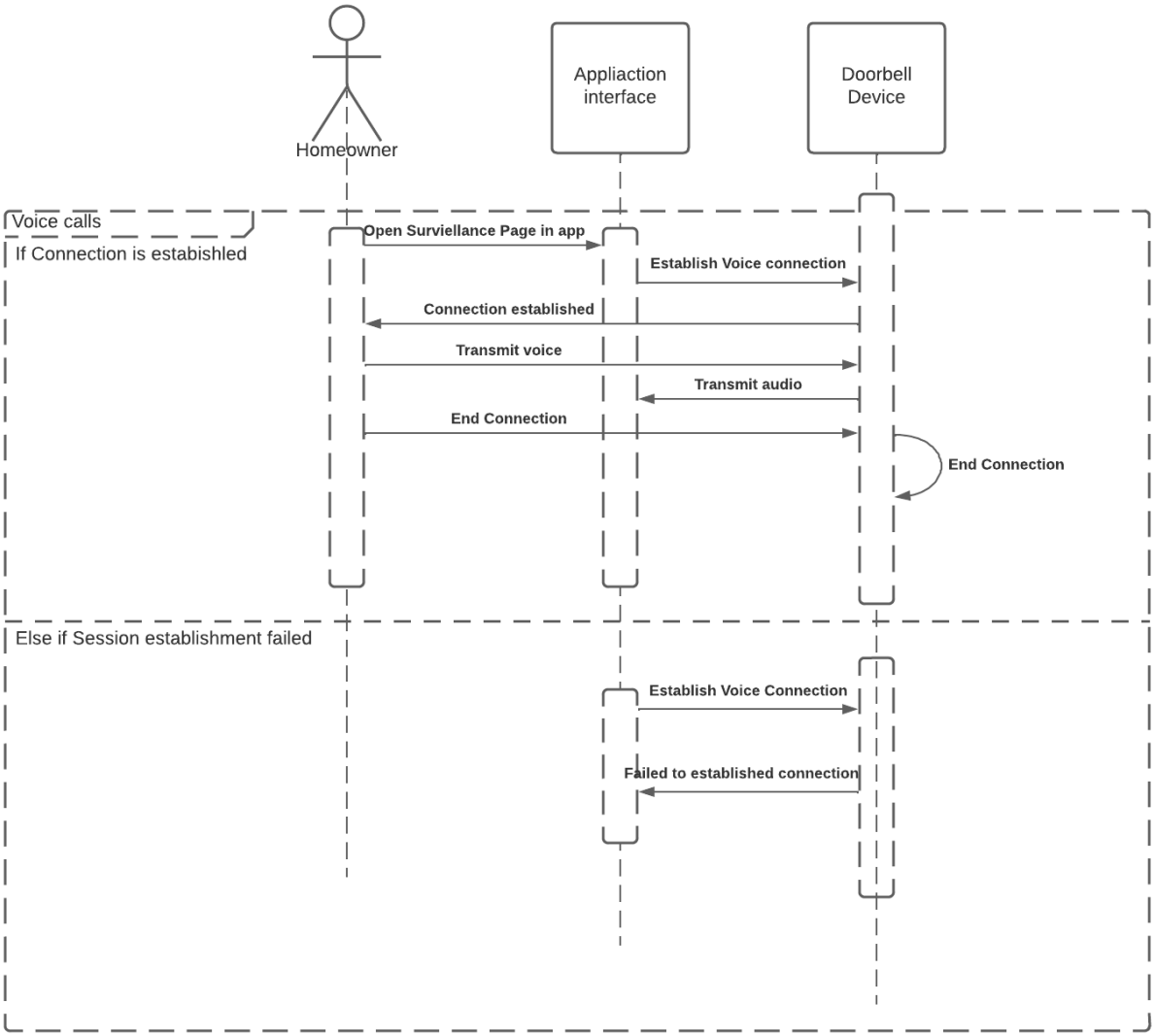

Figure 4.2.4.5 (Voice Call Sequence diagram)

4.2.4.6 Voice calls sequence diagram (visitor)

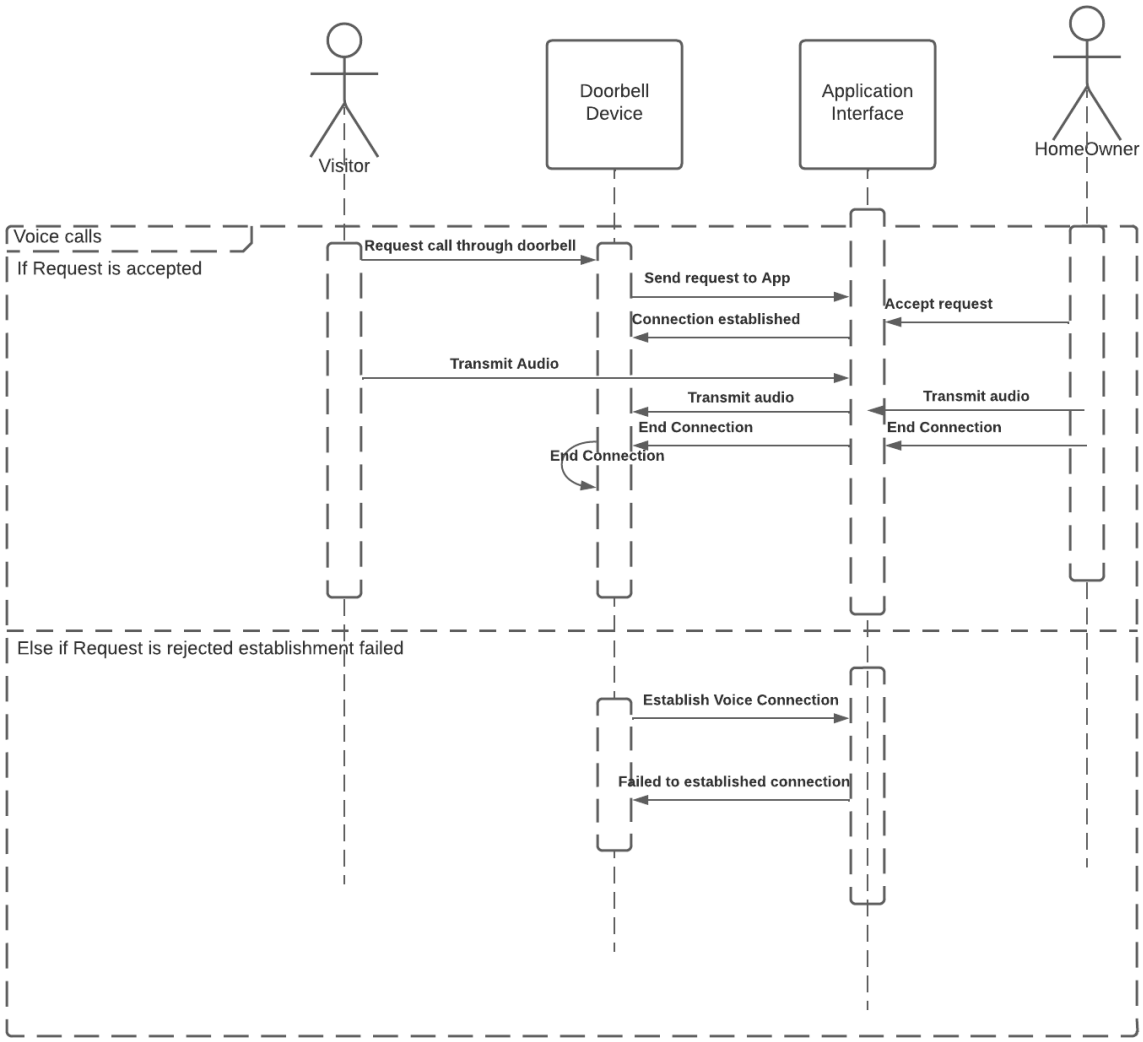

Figure 4.2.4.6 (Voice Call Sequence diagram(Visitor))

4.2.5 Requirement Analysis

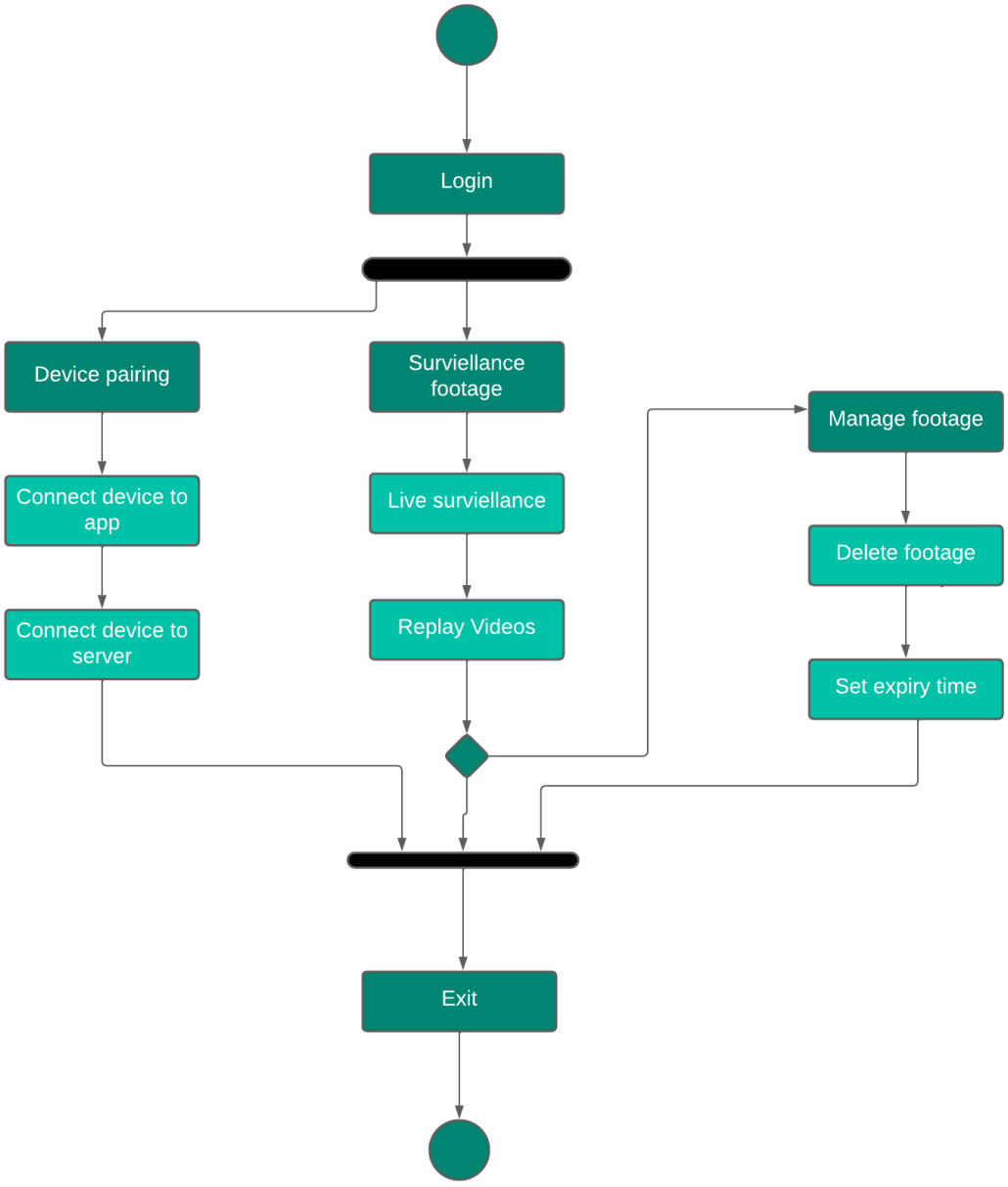

Figure 4.2.5 (Activity Diagram)

As shown in the figure above, the activity diagram for homeowner explains the functions and ways that the user can interact and use the system. The user can view and manage surveillance footage, and pair the device with their smartphone, and system server. This includes viewing old footage, setting and setting an expiry time on stored footage so it deletes after a specified period of time.

4.3 Project Design

4.3.1 System Architecture Design

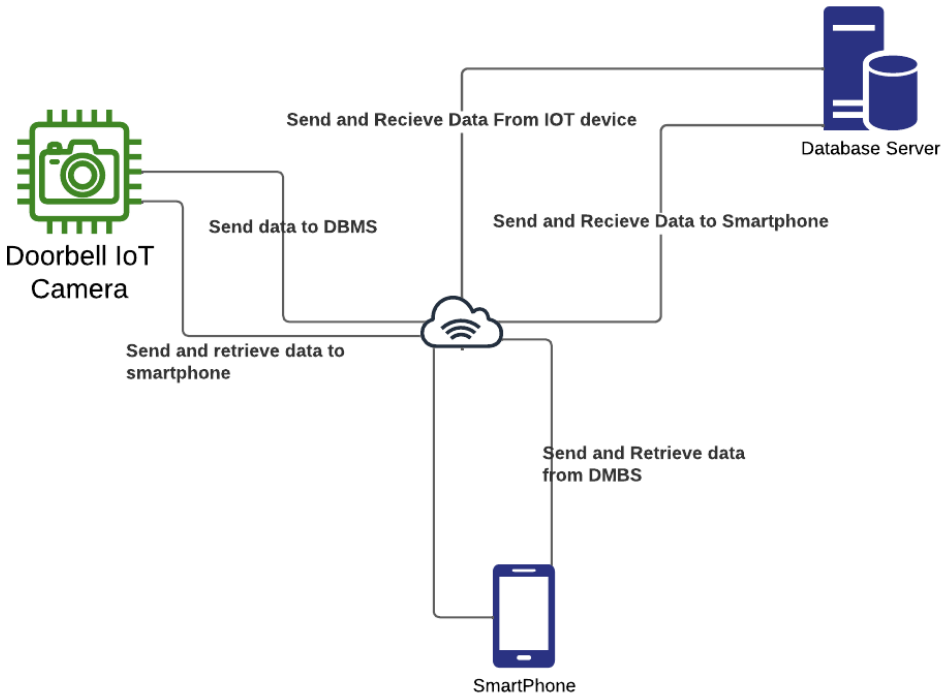

Figure 4.3.1 Architecture Diagram

The system architecture and design show the relationship between all the hardware and software used in the system. The diagram specifies how the devices will interact with each other and which devices will send and retrieve data to each other. The user's smartphone will be able to send and retrieve data from both the I.o.T device, and the database server, while the IOT device can only send data to the DMBS, not retrieve data from it.

4.4 Interface Design

I have chosen to do a low fidelity wireframe prototype for the interface design. This will help map out the general idea and pages behind the interfaces in the phone application, we'll go through what each page will do one by one in the following part. The phone application will have 4 primary pages.

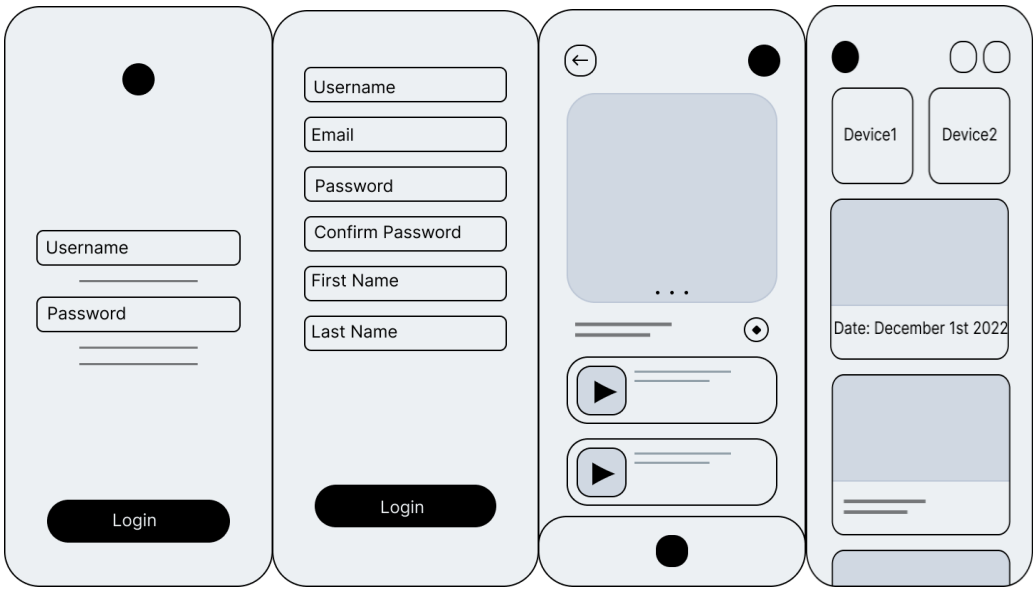

Figure 4.5.1, showing the low fidelity interface prototype.

4.4.1 Live Feed Page

In this page, the user can view the live feed of their device, as well as view captured events by the device. Events are triggered by the motion trigger sensor, where the device will especially store and send a notification to the owner of the motion event that happened at the homeowner's front door.

Figure 4.5.2, showing the low fidelity interface prototype for the live feed page

I have chosen to do a low fidelity wireframe prototype for the interface design. This will help map out the general idea and pages behind the interfaces in the phone application, we'll go through what each page will do one by one in the following part. The phone application will have 4 primary pages.

4.4.2 Storage Management Page

In this page, the user can view old videos from the cloud storage. As well as delete them, and set a time for them to expire, so the database server auto deletes it after a certain specified period of time. The user can view all the footage from all the smart devices paired to their smartphone.

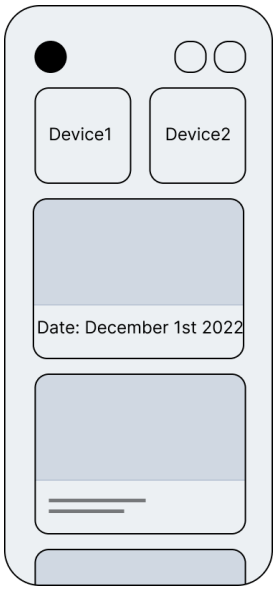

Figure 4.5.3, showing the low fidelity interface prototype for all of the stored videos.

4.5 Chapter Summary

This chapter focuses on the use cases, development and functional requirements of the program as well as how the user will interact with them. This modelled using UML diagrams to ensure that the functional requirements are well defined and to accurately model and develop the UX of the program, when it comes to user interaction with the I.O.T device. There's also a low fidelity prototype to show that the application's user interface will look like.

CHAPTER 5

IMPLEMENTATION AND TESTING

5.1 Introduction

The system has been implemented using a Raspberry pi computer as well as a Raspberry pi camera module. The raspberry pi camera module will be recording footage to send to cloud storage via the internet, and send notifications to the user's phone application through facial recognition. If the camera module recognizes a human face it will notify the user of their being a visitor near the smart doorbell device and the user can view the footage of the camera module.

5.2 Coding of System Main Functions

The coding for the main functionality of facial detection is done through python, as it is the most compatible language with the raspberry pi minicomputer. The raspberry pi facial detection relies on a learning module that draws from a data set of images of people's face to recognize it and differentiate it from background footage that is being recorded.

The raspberry pi sends the footage firebase cloud storage where the user can review the footage that the device has captured and view the pushed notification footage through the phone application. The application, as mentioned before, is developed using flutter and dart framework and language. The functionalities of the system include:

1. Facial Recognition: The system uses a deep learning algorithm do differentiate human faces in the video footage and recognize them. The system can also detect masked individuals, making it not interfere with the functionality of the facial recognition
2. Real time monitoring: The system provides a real time live video feed for constant monitoring wherever the user sets up the device. The user can save the video footage and view the live feed whenever they wish.
3. Notification System: The device is able to send notifications to the user's smart device showcasing any person detected by the device. The user can open the notification and view the live feed to determine who they are.

5.3 Testing

5.3.1 Black box Testing

These black-box testing scenarios cover various aspects of the Doorbell Monitoring System without delving into the specifics of its internal workings. They help ensure that the system meets functional requirements and behaves as expected from a user's perspective.

User Authentication: Test the doorbell system's authentication process.

Test Steps:

- Attempt to log in with valid credentials.
- Attempt to log in with invalid credentials.
- Test password recovery/reset functionality.

Notification System: Test the system's notification capabilities.

Test Steps:

- Verify that users receive timely notifications.
- Test notification content for accuracy.
- Check if notifications are received when the doorbell is pressed and when it's not pressed.

Remote Monitoring: Test the ability to remotely monitor the doorbell feed.

Test Steps:

- Access the live video feed from a remote device.
- Check the quality and clarity of the video.

Mobile App Functionality: Test the functionalities of the mobile app.

Test Steps:

- Verify that the mobile app can connect to the doorbell.
- Test app responsiveness and usability.
- Check if app settings are correctly reflected in the doorbell system.

5.4 Chapter Summary

The Doorbell Monitoring System, powered by a Raspberry Pi, utilizes facial recognition to record and send footage to Firebase cloud storage. Notifications are sent to the user's mobile app, built with Flutter and Dart. Python-coded facial detection relies on a learning module for recognizing human faces. Black-box testing ensures user authentication, timely notifications, and effective remote monitoring. The mobile app undergoes testing for connectivity, responsiveness, usability, and accurate reflection of settings in the doorbell system.

CHAPTER 6
CONCLUSION

6.1 Introduction

The Smart IoT Doorbell Monitoring Device integrates IoT technology into traditional doorbell systems. Utilizing a Raspberry Pi computer and camera module, the device employs facial recognition for secure user authentication. Video footage is sent to cloud storage, enabling remote monitoring, while users receive prompt notifications on their mobile applications developed with Flutter and Dart.

6.2 Achievement of Project Objectives

The project was an overall success with it being able to achieve IOT capability using cloud computing to transmit video to the user's application to for live monitoring as well as be able to store old videos for them to view at their convenience. The main functionality of the device was achieved. However, due to technical limitations some of the features of the device had to be cut such as touch screen capability as well as video call capability between the home owner and the interacting user. This was due to me being only able to locate a raspberry pi module with only 1GB of memory capacity.

6.3 Suggestions for Future Improvement

The device could have benefited from having a touchscreen and video call functionality. Unfortunately, I was unable to get my hands on a 4GB Raspberry pi and only had access to a 1GB one, meaning that the raspberry pi computer didn't have enough capability for video calls and touchscreen support. A custom 3D printed case would have also made the assembly of the device and its components a lot easier, instead I relied on a pre-made raspberry pi case that I had to modify to specification, which made the device look crude.

References

A review: IoT based power & security management for smart home system. (2017, April 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/8203598

A Review on Techniques used for Face Authentication based Smartdoor Bell System using IOT. (2023, January 23). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/10128225

Forensic Analysis of the August Smart Device Ecosystem. (2020, October 20). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9297346

Gomathy, C. K., & Satya, D. (2008). A STUDY ON IOT SMART DOORBELLS. *ResearchGate*. https://www.researchgate.net/publication/354884963_A_STUDY_ON_IOT_SMART_DOORBELLS

IoT application development: Home security system. (2017, April 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/8273688

IoT smart bell notification system: Design and implementation. (2017). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/7890101

Never Ending Story: Authentication and Access Control Design Flaws in Shared IoT Devices. (2020, May 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9283861

Proceedings, C. S. &. I. T. &. C. S. C. (2020). AN INTELLIGENT INTERNET-OF-THINGS(IOT) DOOR BELL SYSTEM FOR SMART NOTIFICATION ALERT. *www.academia.edu*. https://www.academia.edu/43015934/AN_INTELLIGENT_INTERNET_OF_THINGS_IOT_DOOR_BELL_SYSTEM_FOR_SMART_NOTIFICATION_ALERT

Ranking Security of IoT-Based Smart Home Consumer Devices. (2022). IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/document/9698229

Smart Home with Wireless Smart Doorbell with Smart Response. (2021, October 7). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9590865

State of the Art - Smart Doorbell Systems. (2021, December 21). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/967731

Appendix A PI Code Main

```
    from imutils.video import VideoStream

    from imutils.video import FPS

    import face_recognition

    import imutils

    import pickle

    import time

    import cv2


    currentname = "unknown"

    encodingsP = "encodings.pickle"

    print("[INFO] loading encodings + face
detector...")

    data = pickle.loads(open(encodingsP,
"rb").read())

    vs = VideoStream(usePiCamera=True).start()

    time.sleep(2.0)

    fps = FPS().start()
```

```
    while True:
        frame = vs.read()
        frame = imutils.resize(frame, width=500)
        boxes                                    =
face_recognition.face_locations(frame)
        encodings                                =
face_recognition.face_encodings(frame, boxes)
        names = []

        for encoding in encodings:

            matches =
face_recognition.compare_faces(data["encodings"]
,
                encoding)
            name = "Unknown" #
            if True in matches:
```

```
                matchedIdxs = [i for (i, b) in
enumerate(matches) if b]
                counts = {}
                for i in matchedIdxs:
    name = data["names"][i]
counts[name] = counts.get(name, 0) + 1
name = max(counts, key=counts.get)
if currentname != name:
currentname = name
print(currentname)
names.append(name)

for ((top, right, bottom, left), name) in
zip(boxes, names):
            cv2.rectangle(frame, (left, top),
(right, bottom),
                (0, 255, 225), 2)
    y = top - 15 if top - 15 > 15 else top + 15
```

```
cv2.putText(frame, name, (left, y),cv2.FONT_HERSHEY_SIMPLEX, .8, (0, 255, 255), 2)

        cv2.imshow("Facial Recognition is Running", frame)
        key = cv2.waitKey(1) & 0Xff

        if key == ord("q"):
            break
    fps.update()
    fps.stop()
    print("[INFO] elasped time: {:.2f}".format(fps.elapsed()))
    print("[INFO] approx. FPS: {:.2f}".format(fps.fps()))

    cv2.destroyAllWindows()
    vs.stop()
```

Appendix B PI Code Training Module

```
    from imutils import paths
    import face_recognition
    #import argparse
    import pickle
    import cv2
    import os

    # our images are located in the dataset folder
    print("[INFO] start processing faces...")
    imagePaths                                    =
list(paths.list_images("dataset"))

    # initialize the list of known encodings and
known names
    knownEncodings = []
    knownNames = []
```

```
    for (i, imagePath) in enumerate(imagePaths):
        print("[INFO] processing image {}/{}".format(i + 1,
            len(imagePaths)))
        name = imagePath.split(os.path.sep)[-2]
# load the input image and convert it from RGB (OpenCV ordering)
# to dlib ordering (RGB)

image = cv2.imread(imagePath)

rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

boxes = face_recognition.face_locations(rgb,
model="hog")
    encodings = face_recognition.face_encodings(rgb, boxes)
```

```
for encoding in encodings:
                knownEncodings.append(encoding)
                knownNames.append(name)
    print("[INFO] serializing encodings...")
    data = {"encodings": knownEncodings, "names":
knownNames}
    f = open("encodings.pickle", "wb")
    f.write(pickle.dumps(data))
    f.close()
```