UHOME

CHANIPORN WENGWEERAKEAT
NERAMIT SINGH
SETHAWIT SUWINCHARAT

A PROJECT SUBMITTED IN PARTIAL
FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE
BACHELOR OF ENGINEERING IN COMPUTER ENGINEERING

FACULTY OF ENGINEERING & INTERNATIONAL COLLEGE
MAHIDOL UNIVERSITY

2019

UHOME

CHANIPORN WENGWEERAKEAT
NERAMIT SINGH
SETHAWIT SUWINCHARAT

A PROJECT SUBMITTED IN PARTIAL
FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE
BACHELOR OF ENGINEERING IN COMPUTER ENGINEERING

FACULTY OF ENGINEERING & INTERNATIONAL COLLEGE
MAHIDOL UNIVERSITY
2019

Computer Engineering Project
entitled
UHOME

.....
Miss Chaniporn Wengweerakeat
Researcher

.....
Mr. Neramit Singh
Researcher

.....
Mr. Sethawit Suwincharat
Researcher

.....
Asst.Prof. Lalita Narupiyakul,
Ph.D. (Computer Science)
Ph.D. (Computer Engineering)
Advisor

Computer Engineering Project
entitled
UHOME

was submitted to the Faculty of Engineering & International College,
Mahidol University
for the degree of Bachelor of Engineering (Computer Engineering)
on
April 13th, 2020

.....
Asst.Prof. Lalita Narupiyakul,
Ph.D. (Computer Science)
Ph.D. (Computer Engineering)
Chair Committee

.....
Dr. Vasin Suttichaya
Ph.D. (Computer Science)
Committee

.....
Dr. Konlakorn Wongpatikaseree
Ph.D. (Information Science)
Committee

BIOGRAPHY

NAME	Chaniporn Wengweerakeat
DATE OF BIRTH	February 23, 1997
BIRTHPLACE	Bangkok, Thailand
EDUCATION BACKGROUND	Mahaprutharam Girls' School 2011-2015
MOBILE PHONE	+66 – 91 – 710 – 4971
E-MAIL	jungchaniporn@gmail.com

NAME	Neramit Singh
DATE OF BIRTH	August 10, 1998
BIRTHPLACE	Nong Khai, Thailand
EDUCATION BACKGROUND	St Andrews International School (Phrakhanong Campus) 2014 - 2016
MOBILE PHONE	+66 – 89 – 042 – 2269
E-MAIL	neramit.singh@gmail.com

NAME	Sethawit Suwincharat
DATE OF BIRTH	September 26, 1997
BIRTHPLACE	Bangkok, Thailand
EDUCATION BACKGROUND	Saint Gabriel's College 2011-2015
MOBILE PHONE	+66 – 83 – 461 – 5149
E-MAIL	big.sethawit@gmail.com

ACKNOWLEDGEMENT

Foremost, we would like to express our sincere gratitude to our advisor Asst.Prof. Lalita Narupiyakul for the continuous guidance. Moreover, we would like to express our sincere thanks to our co-advisors, Dr. Vasin Suttichaya and Dr. Konlakorn Wongpatikaseree who gave us valuable advice on improving our project. With the help from our advisors, our project is successfully completed.

Last but not the least, we would like to thank and appreciate the support and the encouragement from our family, and friends.

Miss Chaniporn Wengweerakeat
Mr. Neramit Singh
Mr. Sethawit Suwincharat

UHOME

STUDENTS	Miss Chaniporn Wengweerakeat
	Mr. Neramit Singh
	Mr. Sethawit Suwincharat
DEGREE	Bachelor of Engineering (Computer Engineering)
PROJECT ADVISOR	Asst.Prof. Lalita Narupiyakul
DATE OF GRADUATION	April 13th, 2020

ABSTRACT

An easy to use smart home Android and web application will be developed. The main target of the project is to support elderly people. The application is based on Kotlin language with Android Studio, Nodejs and Vuejs with Visual Studio Code. Users can use the mobile application to control Philip Hue light bulbs and Estimote Location Beacons as normal users and the web application as admins. The devices include Philip Hue lightbulbs and Estimote Location Beacons. The Estimote Location Beacons are used to detect the current location of elderly or each user in the house. The Philip Hue lightbulbs are automatically turned on or off according to routine setting in order to facilitate eldery. The application queries data from the database via Restful API. The result is sent back to alert users on mobile phones.

KEYWORDS: uHome platform/ mobile application/ web application/ Philip Hue light bulbs/ Estimote Location Beacons/ elderly

109 Pages

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (cont.)

LIST OF FIGURES (cont.)

CHAPTER I
INTRODUCTION

1.1 Background and Motivation

Self-automated homes or Smart Homes used to be science fiction until recent developments in technology. This particular industry has been seeing a lot of innovation and is continuing to grow [1]. Today, smart homes are in widespread use and are continuing to grow.

The definition of a smart home is " A term that refers to modern homes that have appliances, lighting and/or electronic devices that can be controlled remotely by the owner, often via a mobile app. Smart home-enabled devices can also operate in conjunction with other devices in the home and communicate information to other smart devices."[2]

The problem with the current applications of a smart home is that they are produced by multiple businesses. Since most businesses develop their own mobile application due to be used with their product, one smart home is likely to have multiple appliances from multiple brands. This results in a multitude of mobile applications that the user must install in order to control these appliances. Although the smart devices work well with their own mobile applications, some users might find it to be a hassle. For example, the users are unable to perform two tasks simultaneously on devices that do not belong to the same brand without switching between the mobile applications.

Another issue for smart homes is that it might not be user friendly, especially to elderly people. Due to their lack of technical knowledge, elderly people might find it hard to use mobile applications or the devices they are hosted on.

With these issues that are being faced today, it would be extremely beneficial for smart homes to have one mobile application that controls all the devices in their household, but also being more user friendly towards households with elderly people.

The reason that a mobile application should be used instead of a web application is due to the widespread access that people have to mobile phones. Having

a native application on mobile phones will also result in better performance of the application which also contributes to a better user experience.

In order to make a smart home feel more like home, it should be able to help out families in more efficient ways, especially families with elderly people living with them. Sometimes, the elderly people are left home alone on their own while their family members have to constantly check on them or are constantly worried about them. The solution would be to have the smart home become the eyes and ears and monitor the behaviour and conditions of the elderly person at home.

1.2 Objectives

1.2.1 To develop a platform that can support the use of Philip Hue light bulbs and Estimote Location Beacons in the home environment including add, remove or manage Philip Hue light bulbs and Estimote Location Beacons.

1.2.2 To implement a mobile application to control the Philip Hue lightbulbs in a smart home.

1.2.3 To integrate the mobile application with devices including Bluetooth devices and light.

1.2.4 To use monitoring devices, Bluetooth devices, in order to keep track of the elderly person at home.

1.2.5 To use the data from other sources in order to automate the house based on their profile.

1.2.5.1 Collect data from devices that connect through the internet connection.

1.2.5.2 Collect data from the internet for sunrise and sunset time.

1.2.5.3 Store the data on the cloud.

1.3 Scope

1.3.1 An Android mobile application with Kotlin programming that can control Philip Hue lightbulbs in a smart home.

1.3.2 Control electronic devices such as LEDs light bulbs and Bluetooth enabled-device.

1.3.3 A platform serving as a backbone for the Philip Hue light bulbs and Estimote Location Beacons that can be called by the mobile and web application.

1.3.3 A mobile device must be carried by the user at all times.

1.4 Expected results

1.4.1 Expected results

1.4.1.1 A user friendly dashboard on Android mobile application.

1.4.1.2 The application can control all the smart home devices.

1.4.1.3 A smart home system which can detect and alert the users in the risk event of elder at home as a precaution.

1.4.1.4 A web application that users can add, remove and manage devices in the house.

1.4.2 Testing methods

1.4.2.1 User testing

1.4.2.2 Scenario testing

1.5 Timeline

Table 1.1 Timeline

	May	June	July	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar
Research											
Exploring Test											
Setup hardware											
Test API											
Application Development											
Application Testing											
Feature Development											
Testing											
Report compilation											

Table 1.1 shows a timeline of the planning and developing process of uHome Platform and application. It shows the workflow from the start of planning in May 2019 to the final product in March 2020.

CHAPTER II
LITERATURE REVIEW

2.1 Arlo Security Camera

Arlo security cameras are the one we choose specifically for the feature that created for elderly. They can work without any wire connected because they use battery as power supply. The cameras will record when motion or sound is detected. These cameras also can link or sync with other devices through WIFI which needs upload speed more than 1Mbps. They come with Arlo base station which is connected to cameras via WIFI and this base station will stream video recorded to Arlo account in the cloud. Once the base station connects to the user's home network, the user will be able to watch their camera streams anywhere [3].

However, the base station is not wireless. It needs to connect to home router via an Ethernet cable and wire-power supply. The WIFI router needs to be optimized for streaming video. The base station should be at least 2 meters between cameras and far from WIFI-enable devices such as telephones and microwaves. In order to sync cameras at the beginning, cameras and base stations need to be close to each other within 30-100 centimetres proximity. Moreover, Arlo requires firmware which is a software that controls specific devices from a specific manufacturer [4]. For positioning, the field of view of cameras is approximately 110-degree both horizontally and vertically.

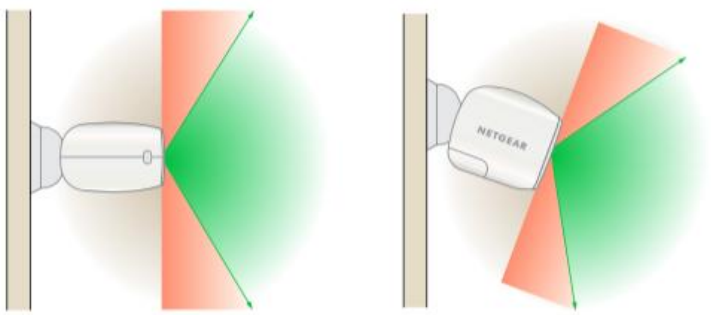

Figure 2.1 View of Arlo Security Camera

As shown in Figure 2.1, the cameras can be placed or mounted on any flat surface such as walls or ceilings with plastic drywall anchors. The maximum distance that cameras can be away from base station is 90 meters without anything obscuring the WIFI signal. Obstacles can be thick walls, brick, concrete material, stone, mirror, etc. The best position to mount the cameras is 2 meters above ground while aiming downwards. Cameras should not be placed above any entry point to bad motion detection because the cameras do not detect well when there is movement toward or away from them.

Arlo can be integrated with other smart home platforms with IFTTT (If This Then That) which is a free cloud-based service. [5] IFTTT is a platform that is used for connecting apps, devices and services from different developers or manufacturers. Applets are the methods that make automation. Applets are a kind of macros that link multiple applications to run assigned tasks automatically.

Hence, Arlo cameras record video once their motion sensors are activated. The video will be streamed through WIFI to the cloud system. It can be integrated with other devices or applications by IFTTT services.

2.2 Philips Hue

Philips Hue is a product with LED bulbs which can be changed up to 16 million colours and a bridge to connect multiple bulbs in one system [6]. The setup of the product is to connect the bulbs into existing light fitting and turn on the switch. Then, connecting the bridge with the Wi-Fi router via network cable is required. The last step is to connect the mobile application to the bridge. The Philips Hue works based on Zigbee, an IEEE 802.15.4-based specification for the low rate wireless personal area network. The ZigBee is primarily used between a sensor and a control system in a two-way communication [7].

Figure 2.2 Connection of Philip Hue

Philips Hue can integrate with other smart devices such as Alexa for automating a smart home. It can work under a low-power with safe and reliable technology even if the internet goes down. However, a Hue bridge requires a connection with wireless router via Ethernet cable. The router does not need to have an internet connection because the bridge only uses the router to send wireless signals to the Hue bulbs.

Philips has just announced a new Philips Hue bulb that can be controlled via Bluetooth and it does not need a hub to connect between them [8]. This new product provides an advantage that the user has to buy only the Hue bulb instead of buying both hub and the bulb. Many companies have taken this approach in their development of smart bulbs. It is also better for people who want only a few smart lights in the room rather than a full functioned smart home.

However, the Bluetooth connection in the new Hue bulb has a shorter range than the wireless system in the old Hue bulb that has a used of ZigBee. This means the users can control the Bluetooth Hue bulbs only when the phone is in range of the bulbs. The Philips solves this problem by remaining the Zigbee connection in the new product. The users can buy a few amounts of Bluetooth bulb and buy a hub if they want to add more bulbs in the system.

In this project, the wireless Philips Hue bulb is used instead of Bluetooth bulb in order to affirm the strong communication of smart devices in one smart home system through mobile application.

2.3 Alexa

Alexa, Amazon Echo, basically works with speech using natural language processing or NLP. Alexa gets input from users as speech and sends it to Amazon voice service. Analysis will be analysed in a server which splits the user's order into individual sound and querying databases to identify significant words. The sound input will be converted into text format. Then it will be able to indicate a task to perform an action corresponding function. Amazon's server sends back responding information and the Alexa will speak as it needed.

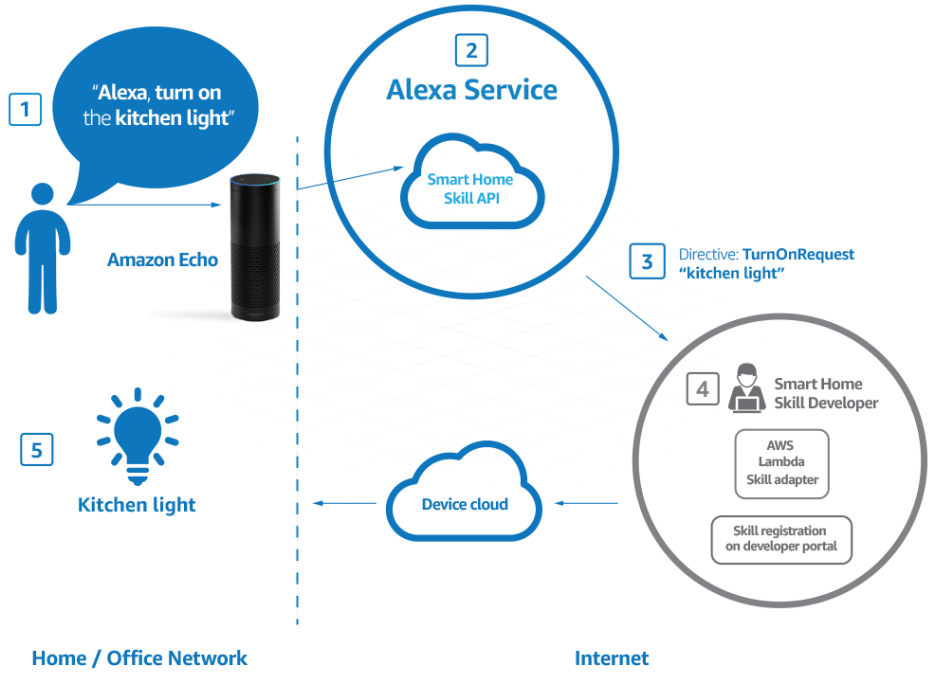

Figure 2.3 Flow of Alexa

According to Amazon Alexa's website [9], Alexa is built on cloud-based service which can be a web service or AWS Lambda. It can communicate using request-response mechanism. These requests and responses will be sent in JSON format.

2.4 Bluetooth-enabled Device

A Bluetooth tracker will be used with smartphones with Bluetooth Low Energy 4.0 (BLE). The good thing is that Bluetooth can communicate with other devices without host networks such as satellites for GPS [10]. Data that is transmitted can be in any form. In this case, Bluetooth signal will be transmitted to alert missing items. The Bluetooth tracker itself has a battery and a chip inside. The ship is the brain of the device that sends and receives signals to make alert sound. Chipolo Plus will be used as a Bluetooth enabled device with replaceable battery. Chipolo also has an API service that we can use with our project. With the Chipolo Plus, users can find items by ringing the Bluetooth tracker or ringing their smartphone using the Bluetooth tracker. Quickly pressing the internal button in the Bluetooth tracker can help users find their phone. The Chipolo Plus can make 100dB sound and the battery lasts up to 12 months.

2.5 Database

Database is essential for this project because we need storage for both data of users and Philip Hue light bulbs and Estimote Location Beacons. Database is more suitable than physical storage in terms of internet accessibility. It can also store a large amount of data and set grant permission. There are 2 types of database which are SQL and NoSQL database.

SQL database or relational database requires SQL language. Structured query language (SQL) is the language that is used to manage databases. Data structure must be defined before. The data is added according to the predefined structure. It is appropriate for organising structured data that has relation with others. MySQL is one of the popular SQL databases.

NoSQL is the opposite way. It requires no SQL language and is used with unstructured data. It is more flexible compared to SQL databases. It can store documents, graph, column oriented, etc without pre definition. The stored document also can have its unique structure. The syntax also can vary in different databases. MongoDB is one of the most useful NoSQL databases [11]. In this project, we will use

MongoDB (a NoSQL database) as our main storage for the alert system and SQL for keeping users and devices.

From the MongoDB website [12], MongoDB is flexible with JSON-like documents that can be various. The data structure in MongoDB can be adjusted over time. It is also easy to work with since it requires NoSQL. It supports a variety of data and queries. Moreover, the document model can be mapped to objects in our application code, so our code will define schema for our database. MongoDB also supports reaching by regex and fields. The MongoDB can be stored on local computers or automated with a cloud service by MongoDB Atlas.

Therefore, MongoDB is a NoSQL database that is easy and flexible to use but it cannot handle some information that has a relationship to others, so MySQL will be used to handle data that requires relationships. They can work with many programming languages with JSON format.

2.6 Application Program Interface (API)

An application program interface (API) is a set of routines, protocols, and tools for building software applications [13]. For this project, it essentially allows the application to get information from external services without having to develop it from scratch. The API that will be used is the Video Intelligence API by Google.

The Video Intelligence API is essentially a cloud-based service with a pre-trained machine learning models that automatically recognize a vast number of objects, places, and actions in stored and streaming video [14]. It is highly efficient for common use cases and improves over time as new concepts are introduced.

The video footage from the camera will be spliced and sent to Google's cloud for processing using the API and it will allow and the processed data from Google's cloud will be sent back to the program to detect whether the action detected in the video footage is considered as a normal behaviour or an abnormal behaviour. The parameters of what is considered to be a normal behaviour or an abnormal behaviour will be preset by the developers.

2.7 Mobile Application

The mobile application in this project is based on an Android operation with the Kotlin programming language. Kotlin is a cross-platform and general-purpose programming language. It is designed to fully interoperate with the old Java. The Java Virtual Machine or JVM version of its library depends on Java Class Library.

The Java Virtual Machine (JVM) is a virtual machine that enables computers to run Java programs and other languages which are compiled to Java bytecode. It acts as an abstraction between the executed code and the operating system. The program will use the instruction sets provided by the JVM and then the JVM will translate those instructions to machine specific instructions later. The Java Virtual Machine will see the file containing machine instructions or bytecode as a binary format class and needs to be executed [15].

Kotlin is officially supported by Google for Android development. It is fully supported in Android Studio and is compatible with the Android system. Additionally, Kotlin applications can run as fast as Java does. Kotlin can run even faster than the same code in Java by coding using lambdas because of features that support inline functions [16].

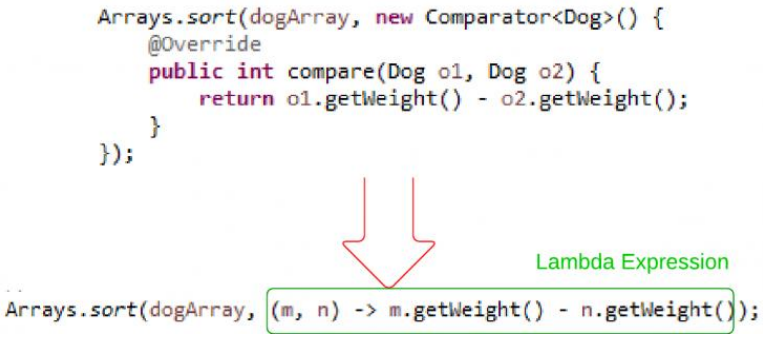

Figure 2.4 Lambda Expression of Kotlin

Kotlin provides some features to solve some Java issues. For example, in Java, accessing a member of a null reference will result in NullPointerException and the Kotlin is aimed to eliminate this issue [17]. The NPE can cause in many ways such as an explicit call to throw NullPointerException() and java interoperation to access a member on a null reference. The system in Kotlin differentiates between nullable references and non-null references.

```
fun main() {
   var a: String = "abc"
   a = null // compilation error
}
```

Result: Null cannot be a value of a non-null type String

To allow nulls, a nullable string must be declared which is written as ‘String?’. The extra character to make a variable be nullable is the question mark after the type of the variable.

```
fun main() {
   var b: String? = "abc"
   b = null // ok
   print(b)
}
```

Result: null

The application will let the user control each device through only one application. By integrating all devices, the application will be used to control the Philip Hue light bulbs manually and perform some specific functions. New devices can be added to the system by connecting the device’ signal to the receiver. Disconnecting the device can be done by performing the action in the application.

The application will receive different types of information from each device. From the default, it can perform turning on and turning off the light for Philips Hue and control the Arlo Security Camera. The application will receive the information from those devices by calling the services and choose what operation to respond to that type of information.

2.8 Smart Home Technical Report

In this paper [n], the Telecommunication Engineering Centre Department of Telecommunications Ministry of Communications Government of India aims to create

a generic platform that accepts multiple technologies as its inputs and somehow combines these inputs at the Home Gateway level and sends the information about various sensors from all the vendors on a single high speed communication link.

The IoT gateway working with Smartphone and Cloud Application is connected to Things on 6LoWPAN network. High quality open source mesh networking stacks such as Contiki has helped the proliferation of IoT. In this gateway it is possible to have sensors connected to the cloud that can communicate using Bluetooth, Wi-Fi, Sub-GHz and NFC. This concept is generic and can be extended to any of different wireless and wireline standards. The architecture is based on a Microcontroller Platform. Likewise, an MPU based platform can be designed with operating system ported as well using the same approach.

Figure 2.5 Gateway device architecture

Home Gateway device is MCU based IoT Platform having different connectivity. The system includes Bluetooth, Wi-Fi, Sub-GHz and Near Field Communication. Wi-Fi is used for exchanging Things or Node data with the Cloud Platform through IoT Home Gateway Platform. Bluetooth is used for communicating the Things or Node data with the Android App through Gateway Platform and Sub-GHz is used for exchanging data between Gateway Platform and Things. An Application layer is added on the Gateway solution that acts as a bridge between the Cloud Application and Things.

The idea of connecting the devices in one system from this paper can be developed in the project along with the cloud and the mobile application. For the project, devices such as Philips Hue and Arlo Security Camera have different types of communication. In order to let the user be able to communicate with those devices on mobile phone or let the devices communicate with each other, the standard type of communication in this system is required.

2.9 Environmental Hazards in the homes of older people

The objective of this paper is mainly focused on the safety of elderly people [18]. The authors aim to investigate environmental hazards in homes, elders' knowledge of injuries and relationship between socio-demographic. The method is a cross-sectional survey of 425 elders with age around 70 and above living in a restricted area. Therefore, the result is that 80% of participants had at least 1 hazard while 39% had more than 5 hazards. Moreover, the result shows that the most hazardous room is the bathroom (66%). More than 60% of the homes had hazards relating to floor surfaces and inappropriate grab or handrails. Most elderly people (88%) can identify falls as the most common injured cause and they know at least one safety measure. Even though older people have self-assessment, their homes still contain more than 5 hazards. The possible hazards such as

- Poor lighting problems such as too dim, too bright, too dark, hard to reach or find switch
- Unaware of furniture or obstacles on walkways
- Slippery floor for the bathroom, grab rails and walkways

In this case, we can use the statistics from this paper to classify the dangerous areas in order to send warning notifications to our users. For example, when cameras detect that the user is going to the bathroom, there will be a notification sent from our system to warn the user to be aware of the wet floor. The obstacles in walkways are also concerned as hazards, so we can implement to our system, for instance, using cameras to detect the obstacles and notice the users when they are approaching. The warning message can also be notified by Alexa for more convenient usage.

2.10 Assistive social robots in elderly care: a review

The author explores the use of assistive social robots in elderly care and the effect that it has on the elderly people. These robots refer to companion type robots such as Aibo and Paro. The research was a secondary research which collected data from databases and Google Scholar to get records in order to draw a conclusion.

The author concludes that there is some evidence that these companion type robots have positive effects on the elderly people. This is with regard to areas such as mood, loneliness and social connections with others.

This can be applied to this project as well. Despite not using any companion type robots in order to give the elderly people some company, the idea of having someone or something to talk to can be used. Amazon's Alexa which will be used can help to fulfill this role. Even though Alexa might not provide the sense of physical touch to the elderly people, it can help provide the emotional support and make the elderly people feel less lonely or bored when they are home alone.

2.11 Estimote Location Beacon

Figure 2.6 Estimote Location Beacons

The Estimote Location Beacon is an indoor location Bluetooth device. The Estimote computes based on nRF52 SoC and DW1000 [19]. The devices that support Estimote are Bluetooth Smart or Bluetooth Low Energy (BLE). This device broadcasts the Bluetooth signal that can be detected by mobile phones (iOS 7 and Android 4.3 or higher). However, it is not a smartphone itself that detects Bluetooth signal, but an

application in the smartphone using Estimote SDK or native frameworks, iBeacon by Apple and Eddystone by Google.

The Estimote Location Beacon fleet data can be accessed by using the RESTful API with Estimote Cloud API. The response from Estimote Cloud API is sent in the form of JSON format which is compatible with many programming languages. In order to use Estimote Cloud API, authentication is needed to be confirmed by unique API credential which requires App ID and App token from Apps section in Estimote Cloud website. Moreover, every request needs to pass authentication using the standard HTTP Basic Authentication. App ID and App Token are used as the username and password in HTTP Basic Authentication respectively.

2.12 Web Application

The web application in this project is based on Vue.js [20]. Vue is a progressive framework that is used for creating user interface. Vue utilizes a virtual DOM or VDOM.

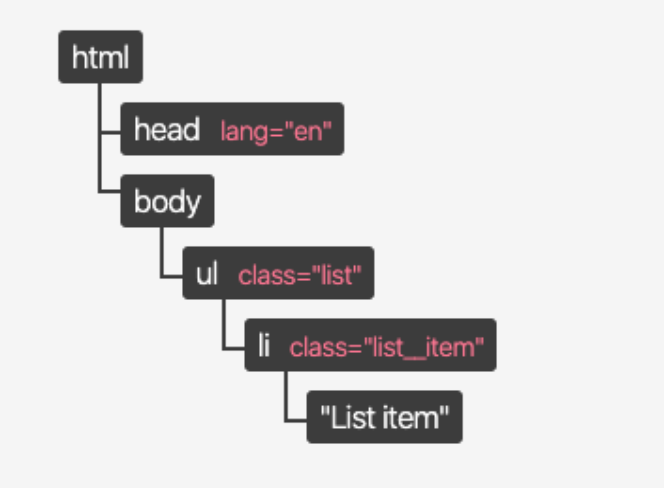

Figure 2.7 DOM tree data structure

```
const listItemOne = document.getElementsByClassName("list__item")[0];
listItemOne.textContent = "List item one";

const list = document.getElementsByClassName("list")[0];
const listItemTwo = document.createElement("li");
listItemTwo.classList.add("list__item");
listItemTwo.textContent = "List item two";
list.appendChild(listItemTwo);
```

Figure 2.8 DOM APIs

DOM, as shown in Figure 2.7, is the object-based representation of the HTML document [21]. In order to manipulate DOM elements, DOM APIs are needed. As shown in Figure 2.8, DOM API (document.getElementByClassName) is used to find a specific element and make some action, such as add, update or delete, on DOM element itself. However, DOM is suitable for small scale and updating the whole data, for example, replacing the entire list is easier than replacing a specific element in the list. In this case virtual DOM can be used to solve this problem. VDOM can be frequently and specifically manipulated and updated without using DOM APIs.

```
const vdom = {
    tagName: "html",
    children: [
        { tagName: "head" },
        {
            tagName: "body",
            children: [
                {
                    tagName: "ul",
                    attributes: { "class": "list" },
                    children: [
                        {
                            tagName: "li",
                            attributes: { "class": "list__item" },
                            textContent: "List item"
                        } // end li
                    ]
                } // end ul
            ]
        } // end body
    ]
} // end html
```

Figure 2.9 VDOM structure

Vue also provides reactive and composable view components and maintains focus in the core library. Companion libraries will handle routing and global state management. Vue is outstanding in the term of ecosystem and abundance of custom renderers. Runtime performance of Vue is quite fast, as good as React.

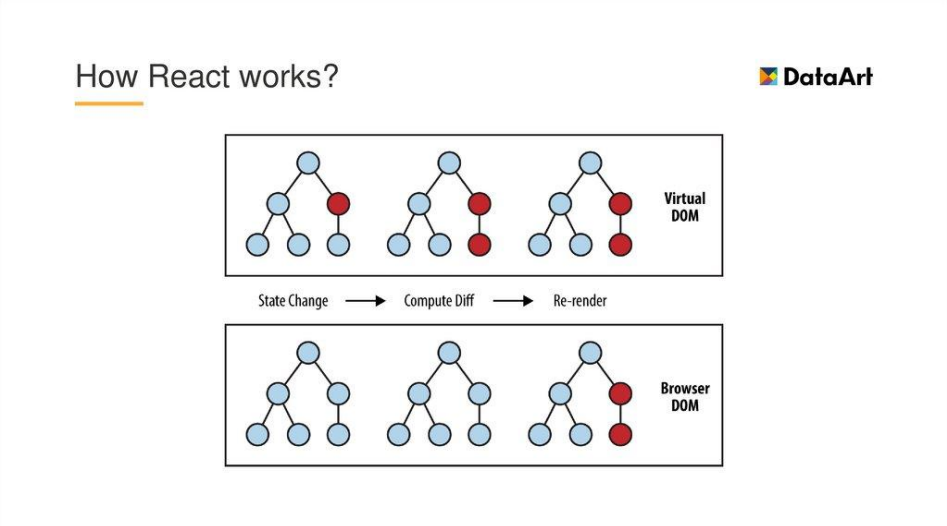

Figure 2.10 Comparison between DOM and VDOM

In comparison, React component's state changes trigger the re-render of the entire component sub-tree. In this case, immutable data structure may be needed in order to make state changes more optimal and friendly. Nevertheless, Vue component's dependencies are tracked automatically during its render. It means that the system knows exactly which components require re-rendering when state changes.

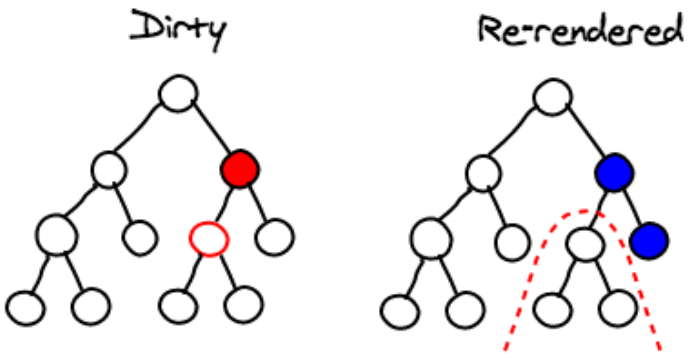

Figure 2.11 Comparison between Dirty and Re-render state changes

Compared to AngularJS, Vue has less complexity and more flexibility than AngularJS. Furthermore, Vue provides the flexibility to tweak the configuration which benefits time consumption for configuration. On one hand, AngularJS uses two-way binding between scopes. On the other hand, Vue imposes one-way data flow between components which makes data flow easier. In terms of runtime performance, AngularJS is much slower than Vue due to the dirty checking. Every time scopes change, all watchers in the system need to be re-evaluated and there are lots of watchers for AngularJS. Therefore, Vue is much faster than AngularJS since it uses a transparent dependency tracking observation system with async queueing. Any scope changes are triggered independently unless the scopes have explicit dependency relationship.

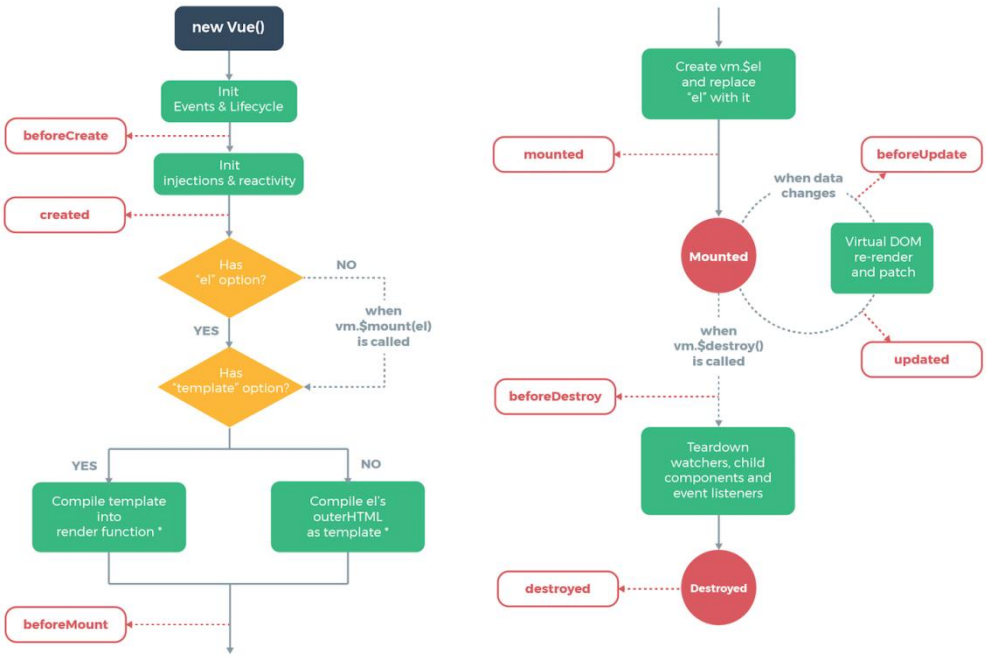

Figure 2.12 Vue lifecycle

Hence, Vue is suitable for this project due to the simplicity, flexibility, time consumption and compatibility. Figure 2.12 represents Vue life-cycle which tells the order or sequence of components in Vue.

2.13 Backend Server

The backend in this project is based on Node.js. [22] Node.js, as an asynchronous event-driven JavaScript runtime, is designed for easily building fast and scalable network applications. Node.js is considered as a server-side platform that is built on Google Chrome's JavaScript Engine (V8 Engine). This platform is also based on a non-blocking I/O model which means that nothing blocks or performs directly with I/O. This benefits Node.js to be free of dead-locking process, lightweight and efficient.

Even though Node.js uses a single threaded model with event looping, it can provide service responding to more requests than traditional servers such as Apache HTTP Server. With the event mechanism, it used non-blocking I/O technique to help the server respond which increases server's scalability, unlike the traditional servers that create limited numbers of threads to handle requests.

```
const http = require('http');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Figure 2.13 Example code of Node.js

In addition, Node.js can handle several connections concurrently, however, if there is no work to do, Node.js will sleep. As a result, Node.js is suitable for data-intensive real-time applications that run across distributed devices.

2.14 Firebase

2.14.1 Firebase Authentication

Firebase Authentication is a service provided by Firebase which provides applications with an easy to use way to authenticate users in order to get the identity of the user currently on the application.

Firebase can be used as a complete drop-in auth solution by using FirebaseUI or by using the Firebase Authentication SDK to manually integrate one or several sign-in methods into an application.

Firebase Authentication creates an OAuth token which can be passed to a Firebase Authentication SDK and the token will be verified and a response of the user's profile will be sent back [23].

2.14.1 Firebase Cloud Messaging

Firebase Cloud Messaging (FCM) is a cross-platform messaging solution that allows messages to be sent between devices.

FCM can be used to notify client applications about anything. Furthermore, it can be used in order to send notification messages to drive user re-engagement and retention. For use cases such as instant messaging, a message can transfer a payload of up to 4KB to a client app.

An FCM implementation includes two main components for sending and receiving:

1. A trusted environment such as Cloud Functions for Firebase or an app server on which to build, target, and send messages.
2. An iOS, Android, or web (JavaScript) client app that receives messages via the corresponding platform-specific transport service [24].

2.15 Amazon Web Services

Amazon web service is a platform that offers flexible, reliable, scalable, easy-to-use and cost-effective cloud computing solutions.

AWS is a comprehensive, easy to use computing platform offered by Amazon. The platform is developed with a combination of infrastructure as a service (IaaS), platform as a service (PaaS) and packaged software as a service (SaaS) offerings.

2.15.1 Amazon EC2

According to Amazon [25], the features of Amazon EC2 are as follows:

- Virtual computing environments, known as instances
- Preconfigured templates for your instances, known as Amazon Machine Images (AMIs), that package the bits needed for the server (including the operating system and additional software)
- Various configurations of CPU, memory, storage, and networking capacity for instances, known as instance types
- Secure login information for instances using key pairs (AWS stores the public key, and user stores the private key in a secure place)
- Storage volumes for temporary data that's deleted when an instance is stopped or terminated, known as instance store volumes
- Persistent storage volumes for data using Amazon Elastic Block Store (Amazon EBS), known as Amazon EBS volumes
- Multiple physical locations for resources, such as instances and Amazon EBS volumes, known as Regions and Availability Zones
- A firewall that enables the user to specify the protocols, ports, and source IP ranges that can reach the instances using security groups
- Static IPv4 addresses for dynamic cloud computing, known as Elastic IP addresses
- Metadata, known as tags, that can be created and assigned in Amazon EC2 resources
- Virtual networks you can create that are logically isolated from the rest of the AWS cloud, known as virtual private clouds (VPCs)

CHAPTER III
METHODOLOGY

3.1 Research summary

In this project, an Android mobile application, web application and backend server will be developed. The mobile application is based on the Kotlin language by using Android Studio software for coding. Moreover, the web application and backend server are based on the Vue.js and Node.js respectively which both of them will be coded using Visual Studio Code. The smart devices can be controlled through the mobile application as normal users if the devices have already been added and permission of the users is granted by admin of the house via web application. MongoDB and MySQL are used as databases to store users' data such as devices in the house, privacy data and time spent in each room. Both web and mobile application will fetch and manipulate data in databases through backend which means that only backend can directly access and perform action with all databases.

3.2 System scenario

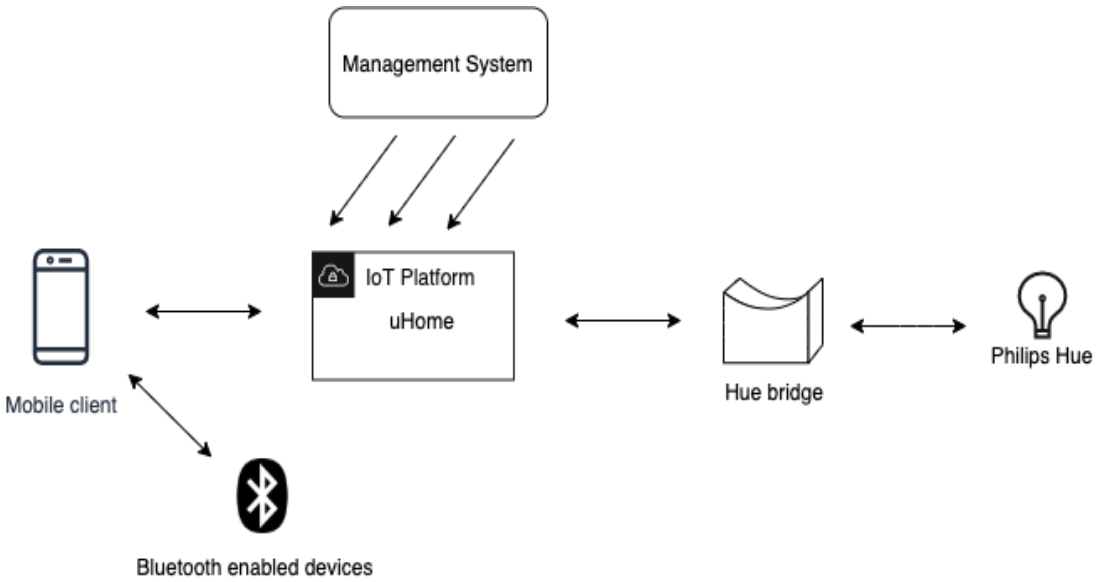

Figure 3.1 System Scenario

As shown in Figure 3.1, Bluetooth enabled devices will be detected on the client side or mobile phones. Web application or management systems work with uHome platform, as well as mobiles. Philip Hue lightbulbs are controlled by the Hue bridge which is manipulated by uHome platform.

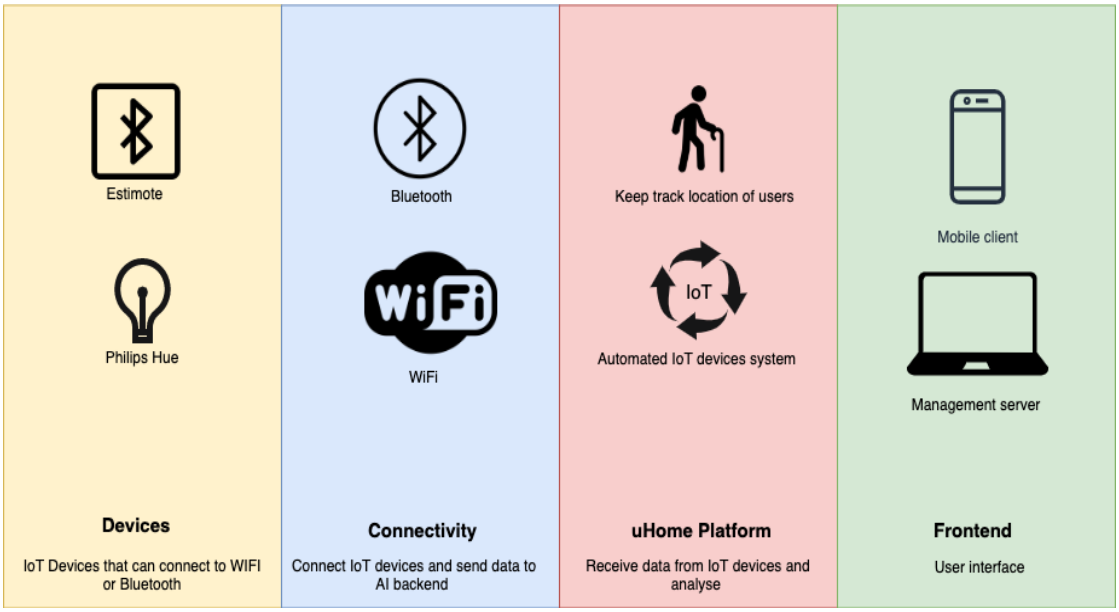

Figure 3.2 System Segmentation

Figure 3.2 illustrates the different segments that the system is divided into. The devices involved is under the “Devices” segment. “Connectivity” segment depicts what type of connection protocols will be needed in order for the system to work. “uHome Platform” segment depicts what functions will the platform perform. “Frontend” segment shows the type of devices that are supported as client devices for the system to operate optimally.

3.3 Virtual perimeter

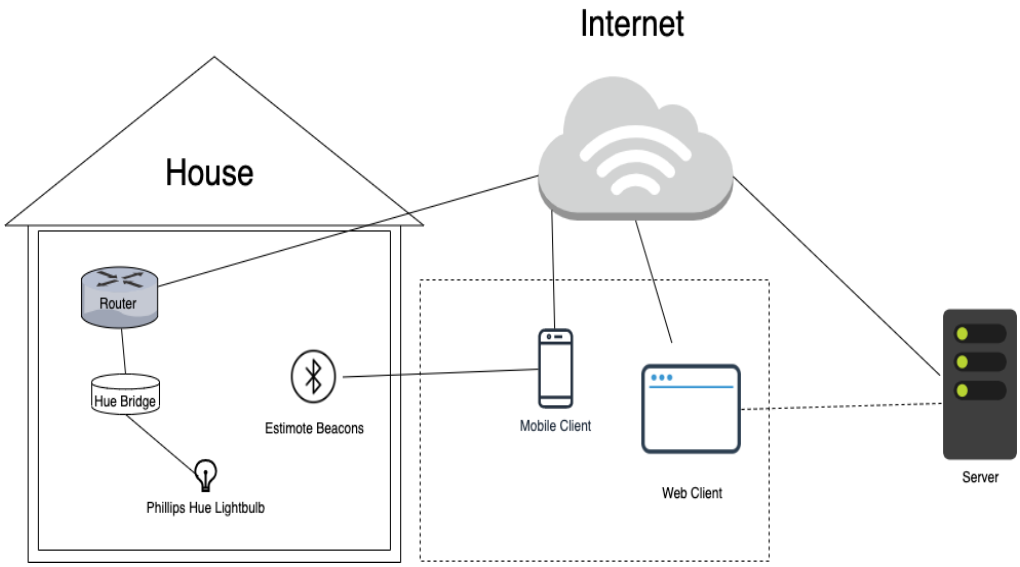

Figure 3.3 Virtual perimeter

Figure 3.3 illustrates how each component of the system will be located when in use. In the figure, the label “House” represents the house of the user. The elements under it shows the devices that are supposed to be physically located in the house. In the dotted square, consists of client devices that can be used to access the system. These client devices can be used to access the system both inside the house and outside the house. The “Server” is located in the cloud. The lines in the figure shows how each device are connected with each other in order to form the system.

3.4 Server functions

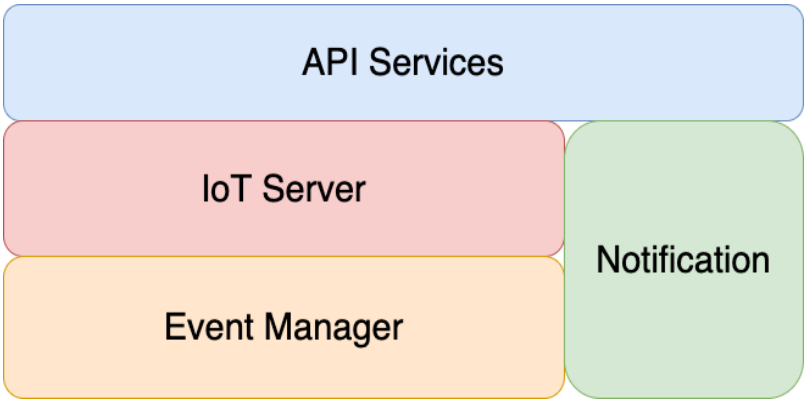

Figure 3.4 Server functions

Figure 3.4 shows the different functions that the server will be able to perform. All the other functions are layered under API services, which acts like a middleware for services provided to clients.

3.5 Design scenario

According to Horgas [18], elderly people with age around 70-103 are home alone more than 60 percent of a day. When they are home, their activities can be

- Personal care such as medical and self-treatment
- Communication
- Eating and drinking
- Washing and dressing
- Toileting
- Mobilization
- Sleeping
- Relaxing such as reading, watching TV and hand craft

If elderly users of the application are not so active, their routine will be pretty much the same every day. Our project focuses on basic routine that can be done within a room (regular condo room).

Elders usually spend time watching TV longer than personal care. However, the frequency of personal care activity is much higher. Moreover, resting also occupies a substantial portion of daytime. Toileting also frequently occurs in a day with a short period of time each round.

3.5.1 Time spent in the bathroom

Toilets can be a dangerous place for elderly people especially if the floor is wet and slippery. Since putting cameras inside the toilet is a huge invasion of privacy, monitoring the elderly persons activity will need another solution. Hence, the elderly persons time in the toilet will be recorded and monitored instead. If the elderly person is in the toilet for more than 30 minutes, it will alert the users to check on the elderly person.

3.5.2 Missing phone

Sometimes people forget their phones and cannot remember where they have put them before they forget. The admin of the uHome platform can use the web application to search for the mobile devices of that user. This means the admin can choose which smartphone's user to find. The target smartphones will start to ring and vibrate until the user clears the notification. The notification will alert whether the application is open or not.

3.5.3 Routine

When there is no one at home, the admin wants to turn on and turn off the light in any area for safety reasons. The admin can choose to set the routine of that house from the web application. The sunrise and sunset time can be chosen as a time to do the operation. The admin of the web application will need to accept the location permission to send the precise location to calculate the time. The custom time is also allowed to set the routine.

3.6 Gantt Chart

Table 3.1 Gantt Chart

Task	16-20 June	21-27 June	28-30 June
Research CH2			
-Light Phillip			
-Bluetooth Device			
-Database			
-API			
-Mobile Application			
Methodology CH3			
-Research Summary			
-Design Scenario			
-Gantt Chart			
Overall			

3.7 Sequence Diagram

3.7.1 Mobile application

The mobile application will be used by the residents in the house including both admin of the uHome platform and the normal user. The user needs to have access to the uHome platform to be able to use the mobile application. In this case, the mobile application will let the user sign-in using the Google authentication which is connected to the firebase database.

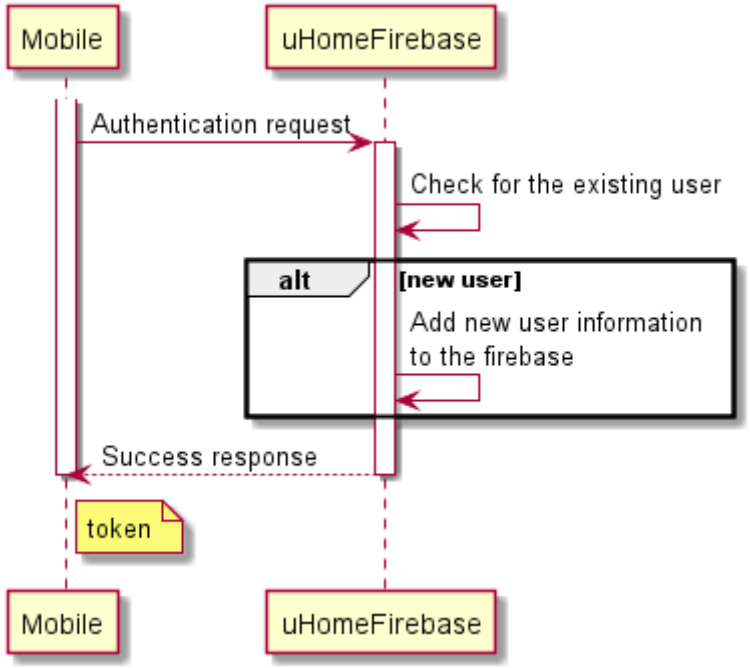

Figure 3.5 Sequence Diagram for login

In Figure 3.5, when the user chooses to login, the mobile application will send a login request to the firebase in order to have access to the application. On the firebase, if the login user already exists in the firebase database, the firebase will send back the token to the user. On the other hand, if the login user is the new user, it then saves the information in the firebase database and sends back the token to the user in the same way as the existing user.

The token from the authentication will be used to send the request to the backend service. Without the token, any request via the mobile application will not be successful. Once the uHome platform has received the information of the credential or

the token, the admin can add the user to the existing house to let the user use the smart device in the house.

Back to the mobile application, the user can choose to start the application or log-out of the application. Normally, the application will keep the token and the credential of the login user of the devices. When the user quits the application and uses the application, the user does not need to sign-in again. On the other hand, if the user chooses to log-out of the application, the application will not keep the token and release the resource for the new log-in session.

Before the user can use the application, the mobile application will send an API request to get the backend server status and display on the screen. In Figure 3.6, the response body from the backend will be “online” if the backend service is ready to use.

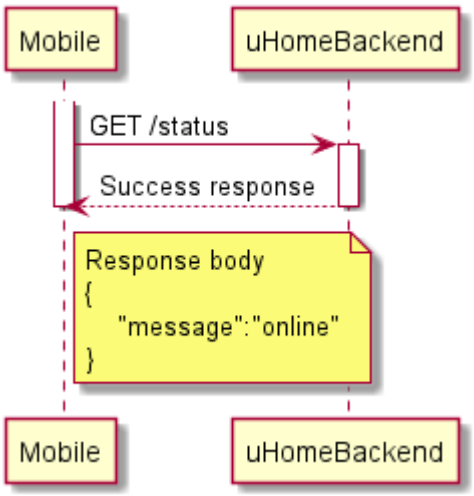

Figure 3.6 Sequence Diagram for checking status

On the first page, the user would see the list of the house that can access. If the user has access to only one house, the user would probably see one accessible house on the list.

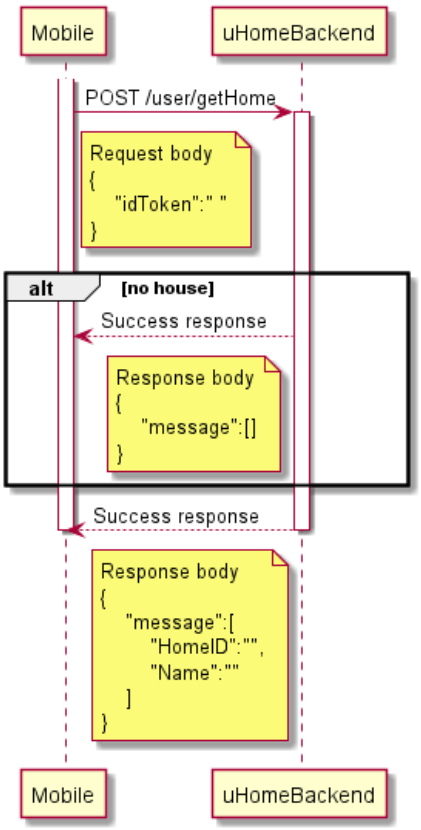

Figure 3.7 Sequence Diagram for getting house list

In Figure 3.7, the mobile application will try to send the request to backend service to get the house list. It will send the API POST request with the request body. The token from the authentication will be attached in the body to be authorized and the backend will verify that the token is valid and has the information. If the requested user has no access to the house, the backend will return as an empty list. On the other hand, the backend service will send responses including home id and the home name to the user, if the user is the resident of that house.

While starting the home activity, the mobile application begins sending the request to the backend to receive the Estimote credential to let the mobile application detect and work with Estimote beacons. The request with a body consisting of a token and home id is required to let the backend service know what Estimote in the house is being called. In Figure 3.8, the backend will send back id and token of the Estimote credential and the Estimote key exists will prove that there is an Estimote key existing in the database.

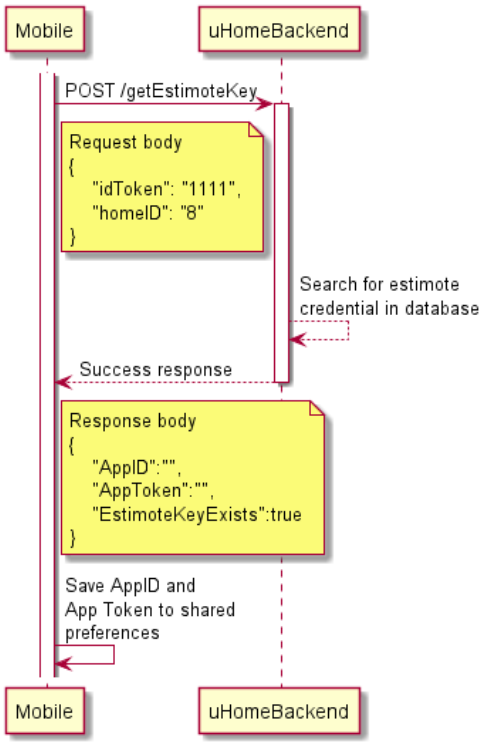

Figure 3.8 Sequence Diagram for getting Estimote credential

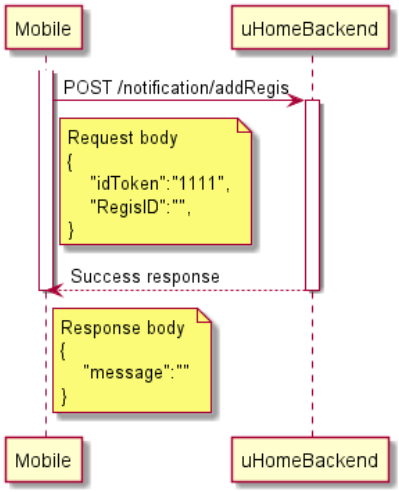

Figure 3.9 Sequence Diagram for sending firebase cloud messaging token

The mobile application will additionally send the API request to add the registration id of the user smartphone to the firebase in order to be able to use the firebase cloud messaging. In Figure 3.9, the mobile application will send the POST API to the backend service. A token must be included with the registration's id to let the request be successful. After adding the id to the firebase database, the backend will

return success response as a message of “record is inserted”. Therefore, the admin can now send the message to the target device as a notification of the activity in the house.

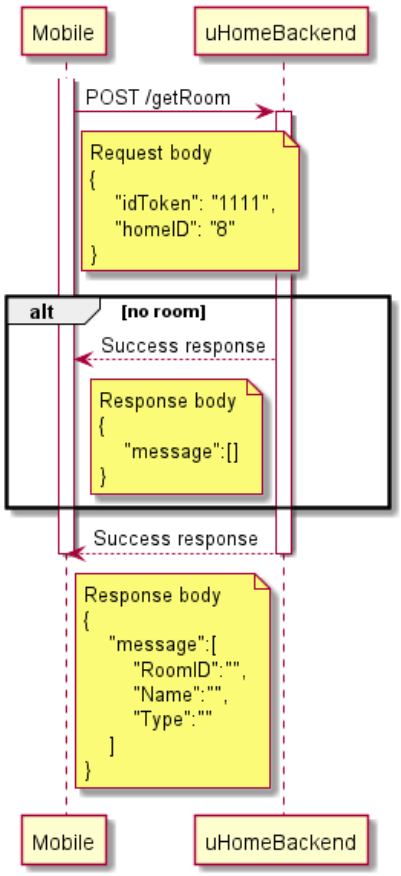

Figure 3.10 Sequence Diagram for getting room list

When the user selects the house by clicking it on the list, the application will direct the user to the room list page. The list of the rooms on this page come from the API request to get the room list from the backend service. This request will send the token of the user and the home id from the previous page to let the backend know which house the user wants to use. As same as the previous API request, the backend will return an empty list of rooms, if there is no room in the system. In Figure 3.10, list of room names, room id and room type will be sent back in the response body to let the user see all available rooms on the application.

Moreover, after the user has clicked on the house on the list of the previous page, the application will ask for permission to the location and Bluetooth to be able to use Estimote with smartphones. The application will start to scan for Estimote in the background. When the user enters the room that has an Estimote place inside or enters

the range of the Estimote, the application will send the notification to the user and begin sending requests to the start timer as in Figure 3.11. The API will send the token of the user, room's id, room's name, room's type and the house's id to the backend to notify the backend service that this user has entered the room. The backend will send back the id to the mobile application to use again in the next task. When the user leaves the range of that Estimote, the application will send API requests including token and id from start timer to stop the timer to collect the time duration in the room. Figure 3.12 shows how the mobile application sends the API request to stop the timer of that id.

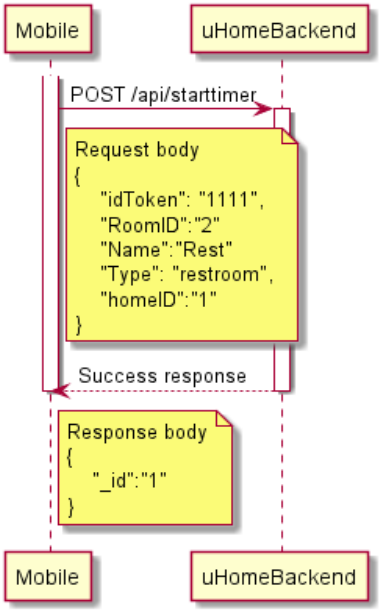

Figure 3.11 Sequence Diagram for starting timer

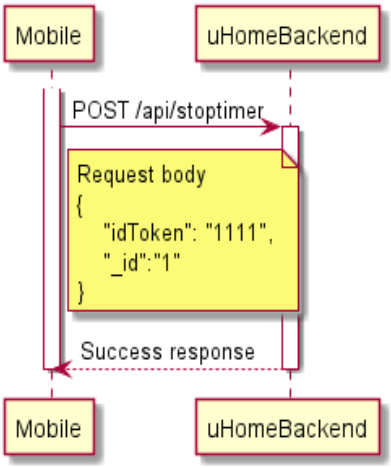

Figure 3.12 Sequence Diagram for stopping timer

The user can select the room from the list and the application will direct the user to the device list page which contains a list of devices that the user can use in the room. This request API will send the room id from the room that the user chooses with the token to receive the list of devices in the room as in Figure 3.13. The response of this API consists of device id, device name and the current status of the devices. If the device is turned on, the toggle button on the mobile application will be “on” or “green”.

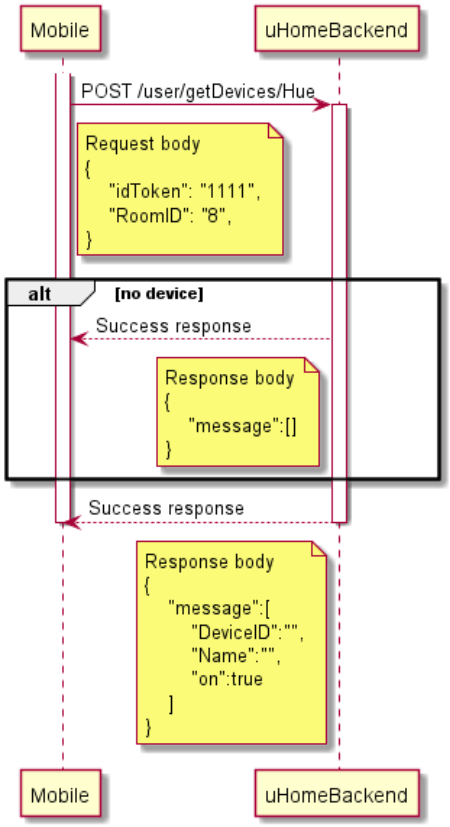

Figure 3.13 Sequence Diagram for getting device list

The user can turn on and turn off the light by pressing on the toggle button of that device on the list. The mobile application will send the API request to the backend to do an action. At this time, the mobile application not only sends the token, but also the device id and home id to let the backend know that the device is in which house in Figure 3.14.

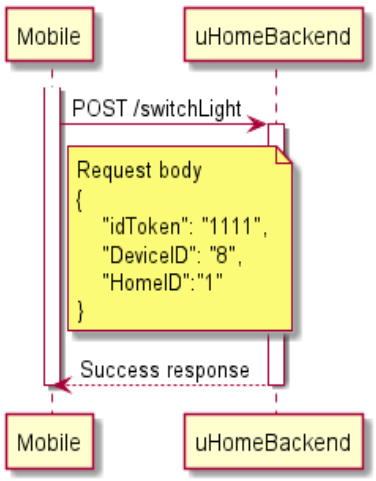

Figure 3.14 Sequence Diagram for toggling light

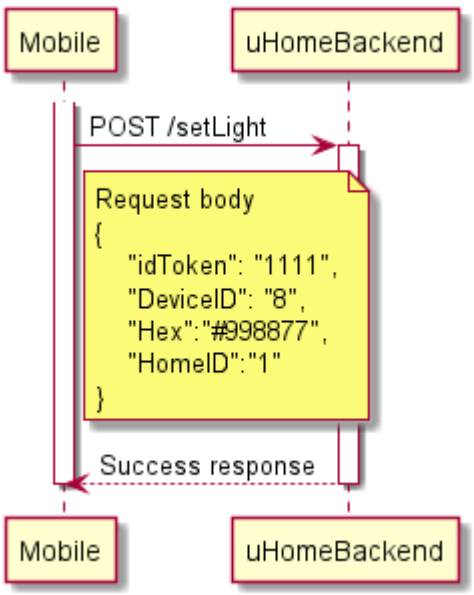

Figure 3.15 Sequence Diagram for changing light color

The user can adjust the color of the light by clicking on the target device. The colour wheel with slide bar to control the brightness will pop-up and allow users to select the light colour. In Figure 3.15, when a user clicks “ok”, the mobile application will send a request to the backend to accomplish a task. The backend will need a request body similar to the API toggle the button, but the addition is the hex value of the light. The light colour will be changed to the target color if the backend returns a “success” response.

Moreover, the user can see the recorded activity of that house by clicking on the top right corner and selecting a record. The application will redirect the user to another activity which shows the bar graph. The graph will display the amount of time the user spent in each room. The user can choose to see the data of any date by clicking on the button to change the date.

The first task of the activity is to get the activity of the current date of using the application as in Figure 3.16. The mobile application will send an API request to the backend to get the room list and the amount of time in each room. In the request body, the token and house's id will be sent with the date of that day in order to let the backend know that the mobile application needs the data of the target house. If the user does not enter any room or the user is in the room, the response data will be empty because there is no time duration to be calculated from start and stop timer. On the other hand, the success response will return the list of room's names and the duration in each room of that user.

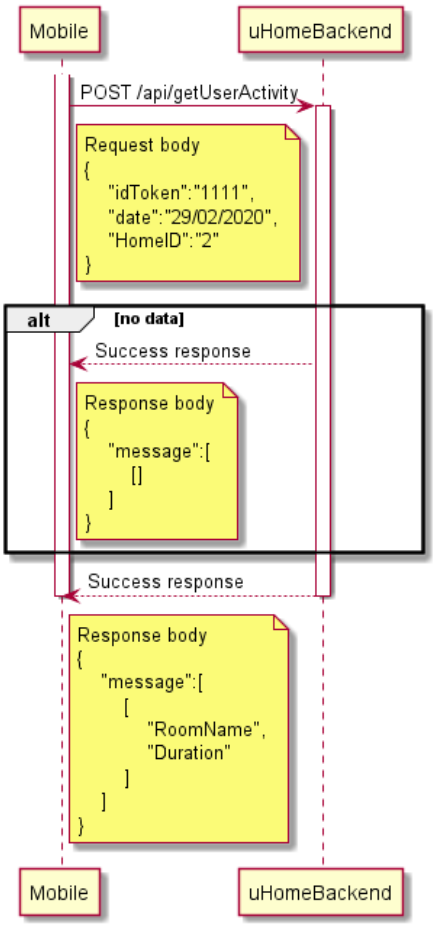

Figure 3.16 Sequence Diagram for getting user record

The last activity is that the user can receive the information of other users in the house by clicking on the top right corner under the 'FindMyFam' menu. This activity will display the information including the latest place and time of the individual family member to track the activity. In Figure 3.17, the mobile application will send an API request to get the current location of the members in the house. By sending the token and house's id, the backend will know that the mobile application wants the information of which house and it might send the list of all members in the house to the mobile application. If the family's member is not in the room, the backend will respond with the unknown room name and unknown time of that user. The response from the backend includes the user's id of the members, user's name, room name where the users are inside, and time when the users enter the room. The information shown on the screen is not real-time data and it requires the user to refresh the application again to receive the new information.

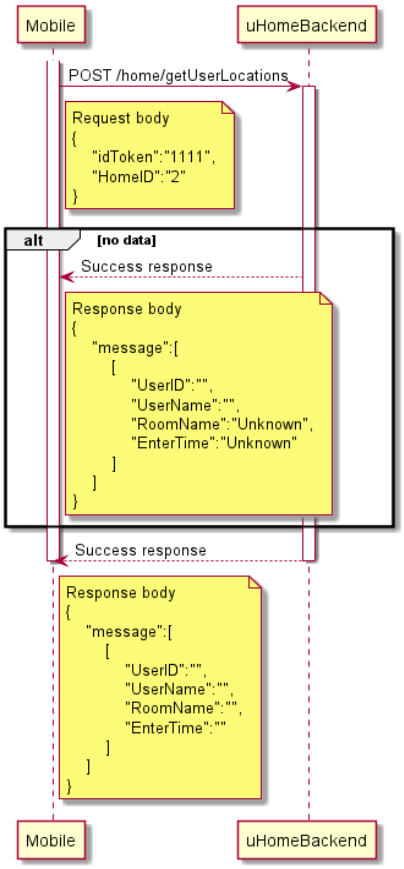

Figure 3.17 Sequence Diagram for getting user location

3.7.2 Web application

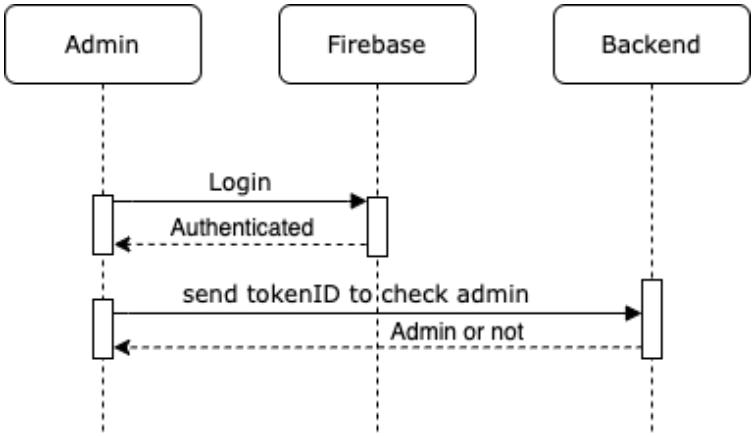

Figure 3.18 Sequence Diagram for Login Page

In Figure 3.18, in the login page, the user is required to sign in with a Google account. Then the web application can get user's information such as email, name, profile picture and token id of the user from Firebase authentication. Token id will be sent to the backend (/checkAdmin) to check whether this user is the existing admin or not. The rest of the information will be displayed on a web application.

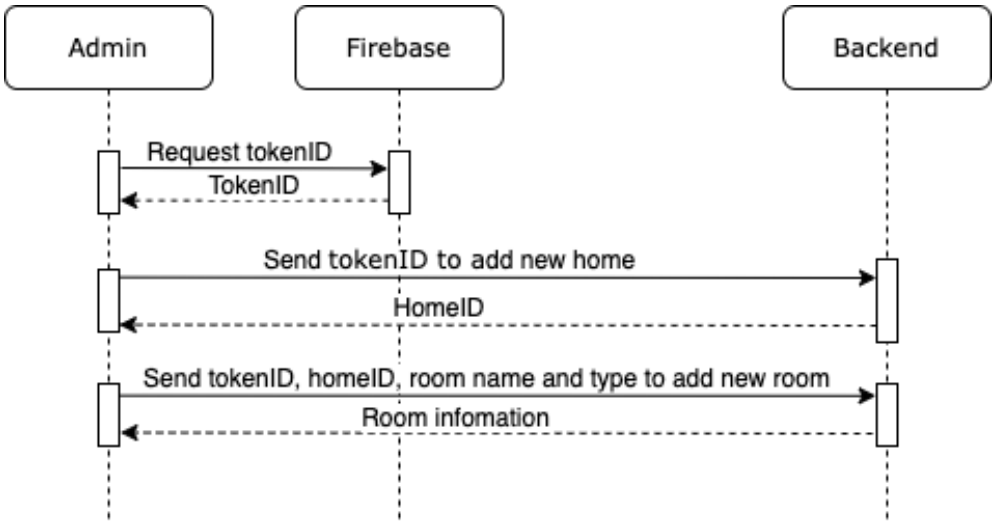

Figure 3.19 Sequence Diagram for Add New Home and Room Page

For the new admin, adding home is the first process that the new admin needs to complete. Web application will request token id from Firebase authentication in

Figure 3.19. After the user fills the name of home, it will be sent to the backend (/admin/addHome) with a token id in order to add a new home. Backend sends home id back as a response. Then the text-field of adding room appears. The user is required to fill room name, size and type. To add a new room, token id, home id, room name, room size and room type are sent to the backend (/admin/addRoom). The response is information about this new room. Moreover, the user needs to add a resident to the new home.

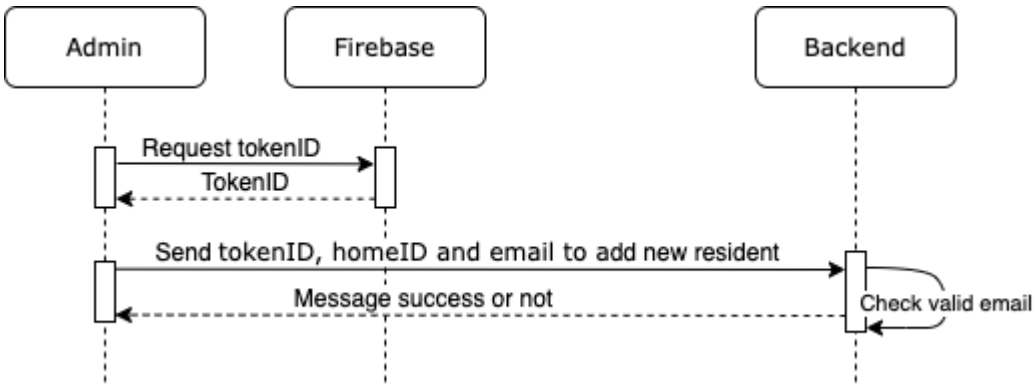

Figure 3.20 Sequence Diagram for Add New Resident Page

In Figure 3.20, in this process, the user is required to enter an email of a resident which needs to be existing users. Web application will send a token id, home id and input email to the backend (/user/addtoHome). If the response is the message “1 record inserted”, adding resident is successful. Otherwise, the user needs to re-enter the resident's email again.

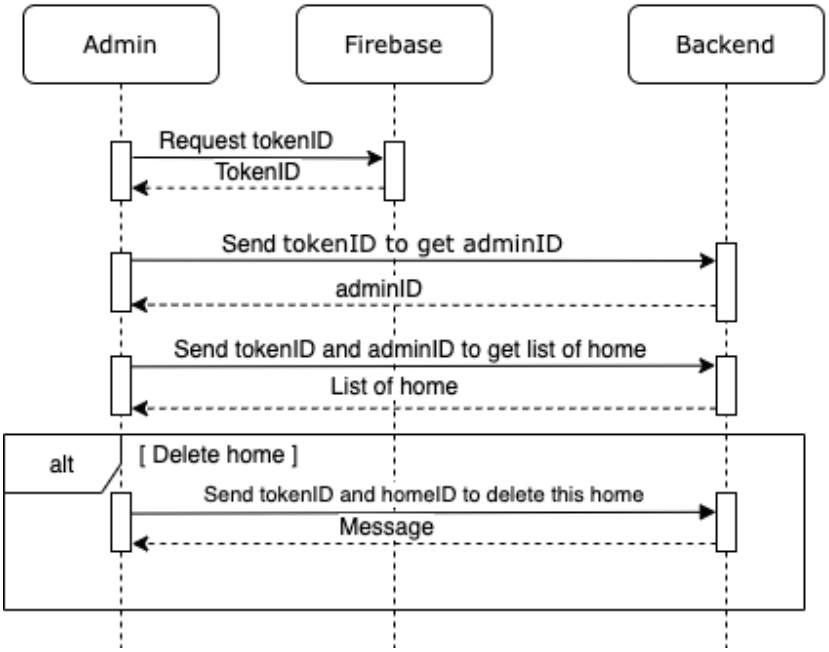

Figure 3.21 Sequence Diagram for Home Page

In Figure 3.21, for existing admin, the first page will show a list of homes under this admin. The token id from Firebase authentication is sent to the backend (/checkAdmin) first in order to get the admin id. Then both of the ids are sent to the backend (/admin/getHome) and get a response as a list of homes. The user is able to get more detail and delete each home, as well as add new home. However, adding a new home and getting more detail are the processes that will be done in other pages. The only process that will be performed on this page is deleting home. In order to delete home, the web application needs to send token id and home id to backend (delete/home).

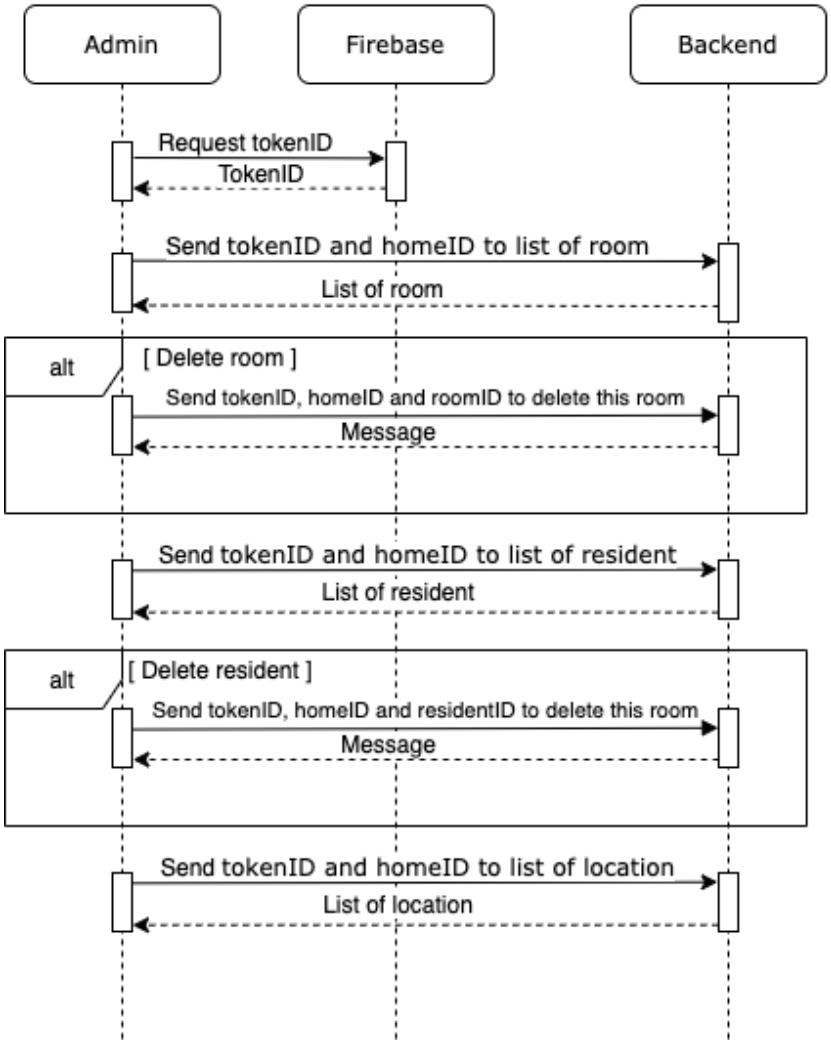

Figure 3.22 Sequence Diagram for Each Home Page

In Figure 3.22, for each home, the web application sends token id and home id that is sent from the previous page to backend (/getRoom, /admin/getUser, and /home/getUserLocations). Response is the list of rooms, residents and location in this home respectively. The same idea as a list of homes that the user can see more detail and add in other pages, but can delete on this page. If the admin chooses to delete a room or a resident, room id or resident id will be sent to backend (delete/room, delete/userfromhome) together with token id and home id.

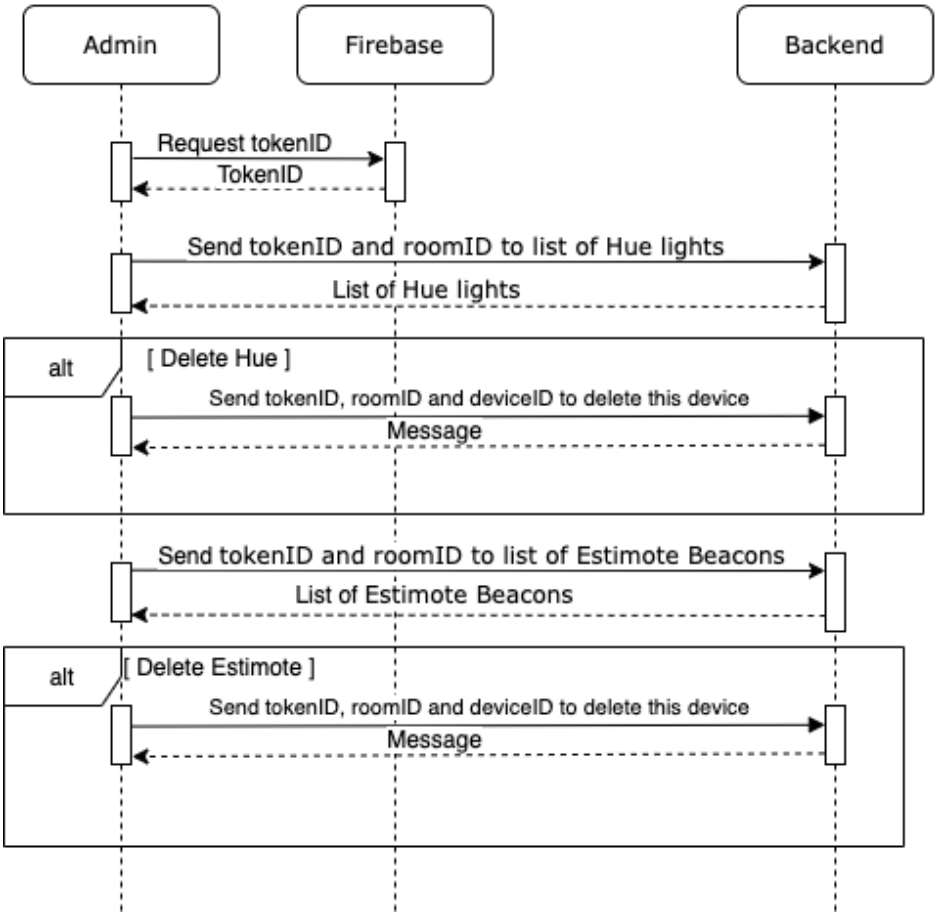

Figure 3.23 Sequence Diagram for Each Room Page

In Figure 3.23, in each room, token id and room id from the previous page are sent to backend (/user/getDevices/Hue and /user/getDevices/Estimote/Beacon/Show) to get a list of both Philip Hue light bulbs and Estimote Location Beacons. These devices can be deleted on this page as well. When the admin clicks the delete button, the id of the selected device will be sent to the backend (api/device/delete) with token id and room id.

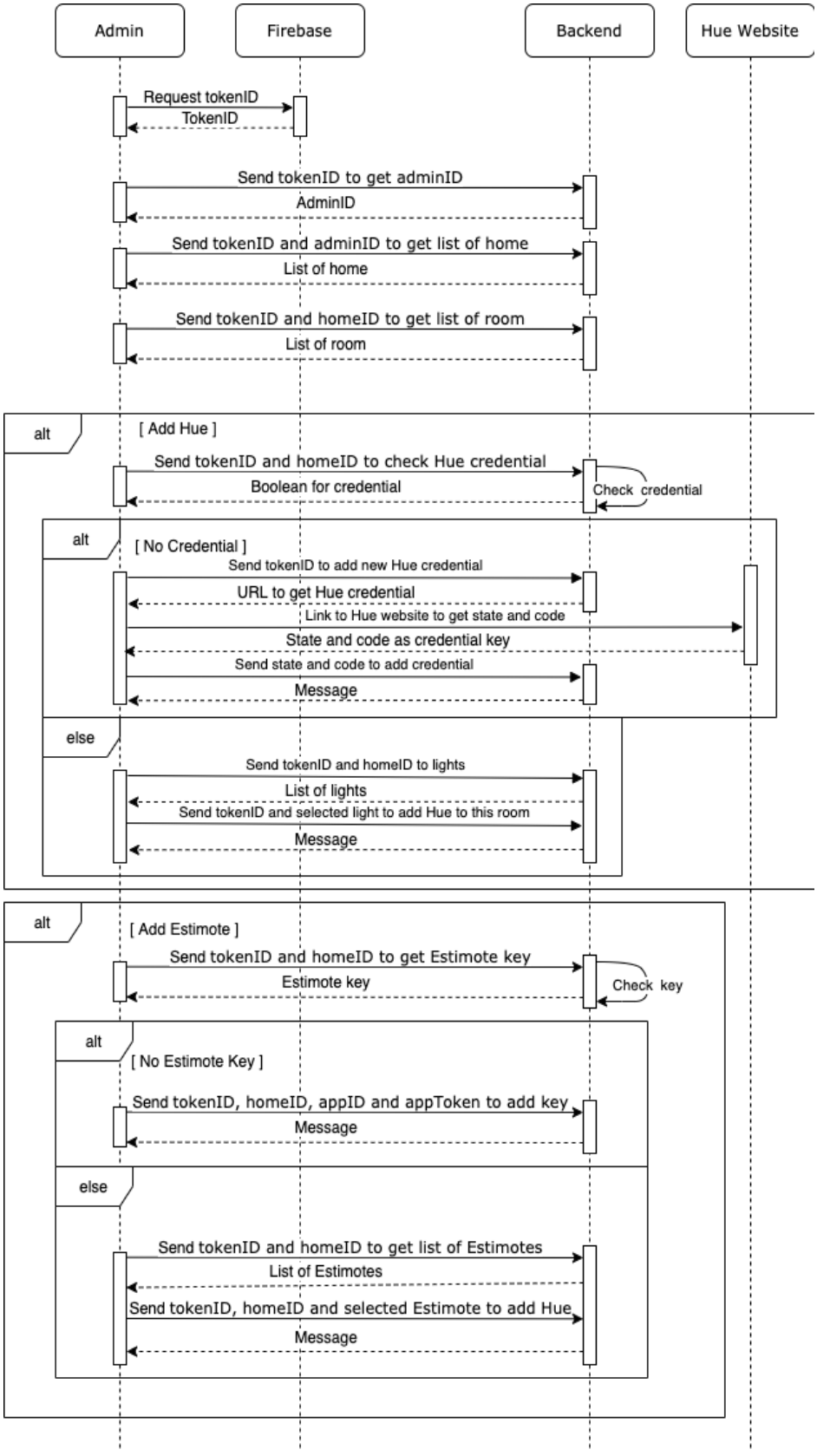

Figure 3.24 Sequence Diagram for Add Device Page

In Figure 3.24, to add a new device, the user needs to choose home, room and type of device. For Philip Hue, web application sends token id and home id to backend (/checkHueCred), and backend responds back boolean. If this home does not have Hue credential, the web application will send token id to backend (/api/addHueUser). The response contains an url that the user needs to visit inorder to get code and state. Once the user visited Hue website, the website will link back to this page with code and state. Then code and state will be sent to backend (/api/addHueUser/callback). After backend gets credential, web application will send token id and home id to get a list of available lights which the user needs to choose. Once the user is done selecting lights, token id, home id, room id, name and light id will be sent to backend (/admin/addDevice/Hue). In case the user chooses to add Estimote, token id and home id are sent to backend (/getEstimoteKey) in order to check that app id and app token exist or not. If there is no Estimote key, the user needs to fill the app id and app token. Then token id, home id, app id and app token will be sent to backend (/admin/add/Estimote/App). Token id and home id also sent to backend (/admin/getDevice/Estimote/Beacon) to get a list of available Estimote. The selected Estimote will be sent to backend (/admin/addDevice/Estimote/Beacon) together with token id, room id, name and home id.

3.8 Database

3.8.1 Table Structure (MySQL)

Table 3.2 Table name: Admin

Column Name	Data Type	Description
AdminID	BIGINT(20)	Primary Key for the table; Used to identify Admins
UserID	VARCHAR(255)	UserID (Retrieved from Firebase Authentication)

Table 3.3 Table name: Device

Column Name	Data Type	Description
DeviceID	BIGINT(20)	Primary Key for the table; Used to identify Devices
Name	VARCHAR(255)	Name of the device
RoomID	BIGINT(20)	Foreign Key; Determines which room is the device in
Type	VARCHAR(255)	Type of device

Table 3.4 Table name: Estimote_Key

Column Name	Data Type	Description
HomeID	BIGINT(20)	Identifier to determine home
AppID	VARCHAR(255)	Authentication value for Estimote
AppToken	VARCHAR(255)	Authentication value for Estimote

Table 3.5 Table name: Home

Column Name	Data Type	Description
HomeID	BIGINT(20)	Primary Key for the table; Used to identify Home
AdminID	BIGINT(20)	Foreign Key; Used to identify Admin of the Home
Name	VARCHAR(255)	Name of the Home

Table 3.6 Table name: Home_User

Column Name	Data Type	Description
HomeID	BIGINT(20)	Used to identify Home
UserID	VARCHAR(255)	Foreign Key; Link UserID to each Home

Table 3.7 Table name: Room

Column Name	Data Type	Description
RoomID	BIGINT(20)	Primary Key for the table; Used to identify Rooms
ID	BIGINT(20)	Foreign Key; Used to link the Room to the Home
Name	VARCHAR(255)	Name of the Room
Type	VARCHAR(255)	Type of Room

Table 3.8 Table name: User_Noti

Column Name	Data Type	Description
UserID	VARCHAR(255)	Used to Identify User
RegisID	VARCHAR(255)	Used to link the Registration Token (retrieved from Firebase Cloud Messaging) to the User

3.8.2 Entity Relationship Diagram

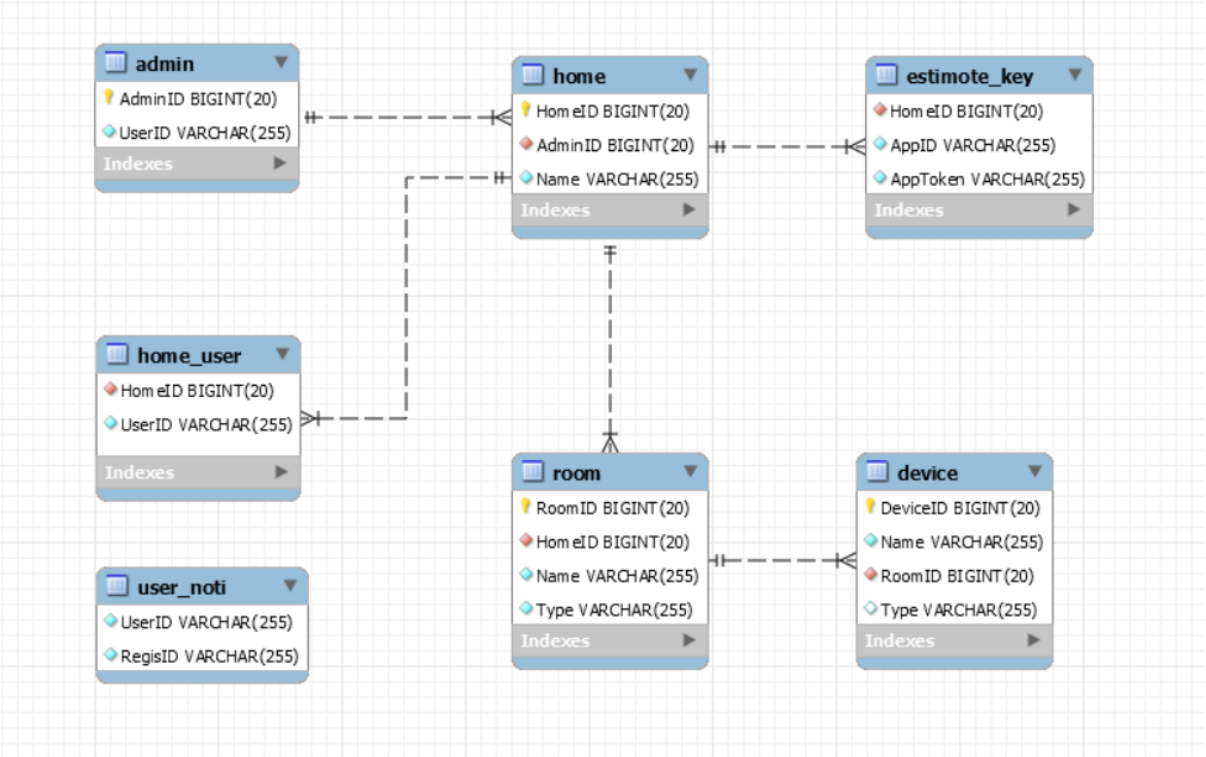

Figure 3.25 Entity Relationship Diagram

The MySQL database has the following relationship between tables (as seen in Figure 3.25). The database was designed so that there is an admin user that can be in charge of a particular home. In a home (house), there can be multiple room(s). Device is then linked in each room. The only table not linked with any table is user_noti as the primary key for this table is UserID which is a string generated from Firebase Authentication.

3.8.3 Collection Description (MongoDB)

3.8.3.1 HueCred

This collection was used to store authentication credentials in order to access Philips Hue API. Each document was also linked with the HomeID (from MySQL table: Home) so that every user under the same house will be able to access and take control of the light. Furthermore, it allows the system to be able to

control the lights without any user interaction in order to enable the system to achieve automation.

3.8.3.2 Devices

This collection was used in order to store information about individual devices. Each document was linked with DeviceID (from MySQL table: Device). For each type of device, there are different JSON objects with different formats and therefore, there is no set schema.

3.8.3.3 Routine

This collection was used in order to store data regarding house automation. Each document was linked with HomeID (from MySQL table: Home) and DeviceID (from MySQL table: Device). This was in order to access Philips Hue authentication credential using the HomeID and also to access the properties of each device. It also stores the time and action that needs to be executed.

3.8.3.4 Setting

This collection was used in order to store the user's preferences regarding the system's execution with respect to the user's house. It stores whether the user wants to turn on Smart Learning and who the target user is. It also stores the time spent in a room before sending a notification to alert other members of the house.

3.8.3.5 Sun

This collection was used to store data if a routine execution was intended to be performed at sunrise or sunset. The data from this collection is used to set the time to perform action and it is refreshed every day.

3.8.3.6 Timer

This collection was used to store data about the user's activity duration in each room.

3.9 Cloud Server

The cloud server was set up on Amazon Web Service (AWS) EC2 Instance. The server hosts the Node JS backend application on port 3000, making it the Cloud Backend server that supports the mobile application and the web application. The EC2 Instance is running on Ubuntu 16.04 server operating system and utilizes a t2.micro tier pricing package which is initially free to use until certain usages are exceeded (i.e. data usage and server uptime).

3.9.1 Backend Application

3.9.1.1 API path hosting

The Node JS application uses a module called "Express JS" to provide ease of use for enabling APIs. These API paths are called by the Mobile and Web Applications and the program will respond to its requests accordingly.

3.9.1.2 Security

The backend application also ensures that the user's data is only viewable by directly related to that particular information. The backend application utilizes Firebase Authentication along with the other parts of the system (i.e. Web and Mobile Applications) in order to identify current users and to ensure that users will only be able to access the system with permission. The backend application uses a library called the "Firebase Admin" which is linked with Firebase Authentication with its purpose being the handler of the tokens sent by clients (Web and Mobile Applications). All API paths are required to have an idToken attached with the request in order for the backend application to get user's information and perform actions as per requested if authorized.

3.9.1.3 Automation (Event Management)

The backend application also manages the actions that need to be performed that have been preset by the user. Every minute, the application reads from the database if there are any actions that need to be performed at that specific time. If it finds any events that are scheduled at that particular time, it will execute them and continue to wait for a minute before checking the database again.

An example of an action would be; turning a light bulb on at 18:00 PM. Once the system's clock reaches 18:00 PM, it will get the information on which bulb to perform the action on and then execute it.

Everyday at 00:00 AM, the system will go through the database to see if any routine actions need to be changed in order to match the day's sunrise and sunset time to perform the action at said time. Then at 00:10 AM, the system will trigger Smart Learning update for users that have activated it.

3.9.1.4 Data Management

The backend application has access to read and write on both MySQL and MongoDB databases. Once it receives a request from a client, it determines if data needs to be read or written and where is the target location of the data that needs to be written or read. With this, the system is able to collect a large amount of data which is then used by the system in order to improve the system.

3.9.1.5 Smart Learning

Smart Learning refers to a function that the system uses the user's behavioural data in order to optimize and improve the system to give a more personalized experience to the user.

The data collected is used to determine the amount of time spent in each room. This allows the system to have the average time spent by users in each room.

The average time spent in the bathroom is summed with the standard deviation of the total time spent in the bathroom in order to get a value, which is then used to update the value of the amount of time spent in the bathroom before triggering notification.

3.9.1.6 Notification

The backend application utilizes Firebase Cloud Messaging in order to send notifications to user's Android based mobile devices. The application implements a library called "node-gcm" which handles sending notifications to mobile devices.

The backend application will send a notification when the user wants to find their mobile device by sending a special type of notification which will make the mobile device ring and vibrate until the notification is cleared on the mobile device.

The application will also automatically send out notification to all the members of a particular household if the preset time threshold is exceeded by the target user (i.e. the elderly person in the house)

3.9.1.7 Device Management

The backend application directly handles the devices that are connected to the system. It uses node-hue-api library as the middleware in order to communicate with the Philips Hue host through their api in order to control the lights.

It also calls Estimote Cloud's api in order to make changes to the Estimote beacons. This includes changing attachment values and changing the broadcasting power (in this case, to match the size of the room).

3.9.2 Database Hosting

The cloud server also hosts the MySQL database and is set up so that it can only be accessed locally. Therefore, only the backend application is able to manipulate the database and hence there is an extra layer of security for the data.

CHAPTER IV
EXPERIMENTAL RESULTS AND EVALUATION

4.1 Evaluation

4.1.1 User Testing

uHome's user interface survey is provided in order to evaluate based on the user's satisfaction both the mobile and web application. In this case, user interface is the main concern since the application cannot be tested by users with the platform. In order to use the fully functional application, devices and the uHome platform are required to be set up.

In order to evaluate the user testing, the survey has been provided for the user's group to use. The sample user interface has also been attached in the survey to let the survey participants understand how the application should work.

The first question asks the user to evaluate the user interface by giving a score from 0 to 5. If the user can clearly see the characters on the screen, the maximum of 5 points will be given. However, if the size of the characters is too small, the users might mark 0 points for this question.

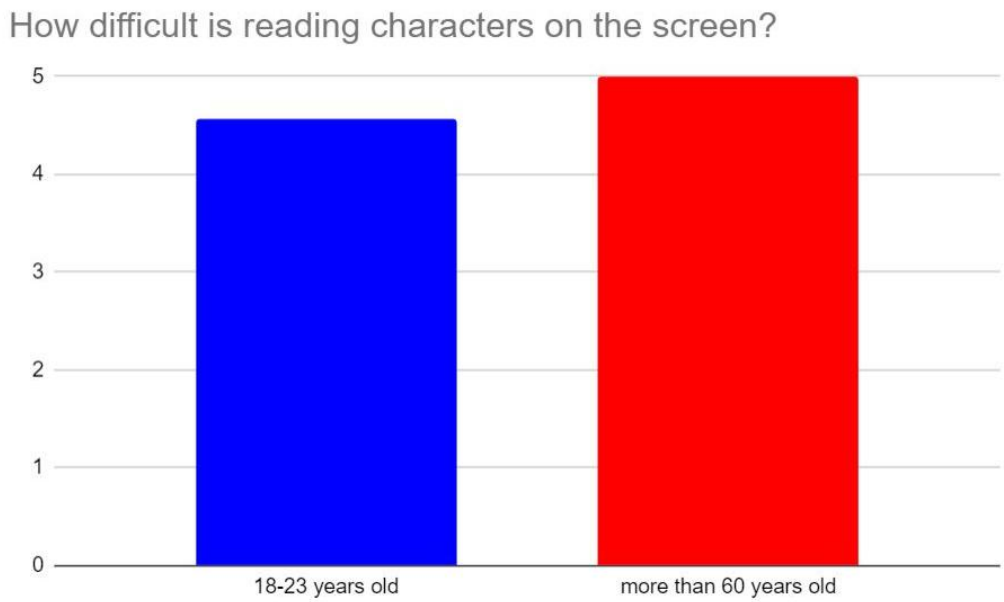

Figure 4.1 Survey of character size

In Figure 4.1, the users have been divided into 4 groups. The first group is the user between 18 to 23 years old and the second group is the user with more than 60 years old. The other two groups which are the users between 23 to 40 and 41 to 60 have no data in this survey. The bat chart shows that the first group of users give the score of 4.5 to the question of easy to read the character on the screen. The second group which is the main target of the uHome system gives the 5 marks to this question. In this case, the elderly agree that the character on the screen is large enough and easy to read.

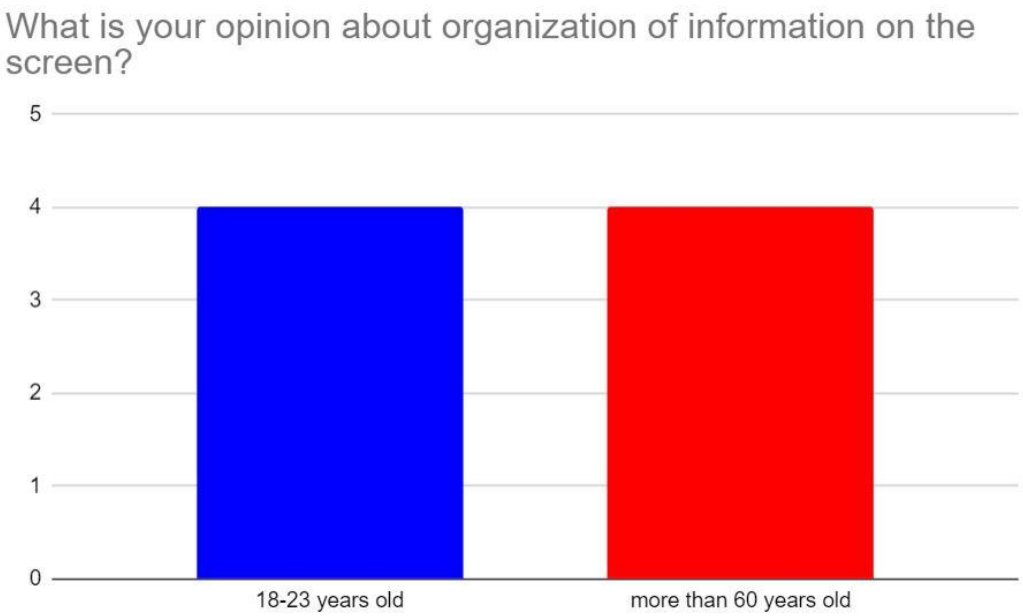

Figure 4.2 Survey of organization on the screen

In Figure 4.2, the survey applicants have been asked to give a score for the organization of the information on the screen. In this question, both two groups give the score of 4 as to prove that some elements and information on the screen are well organized and in the position that is easy to read.

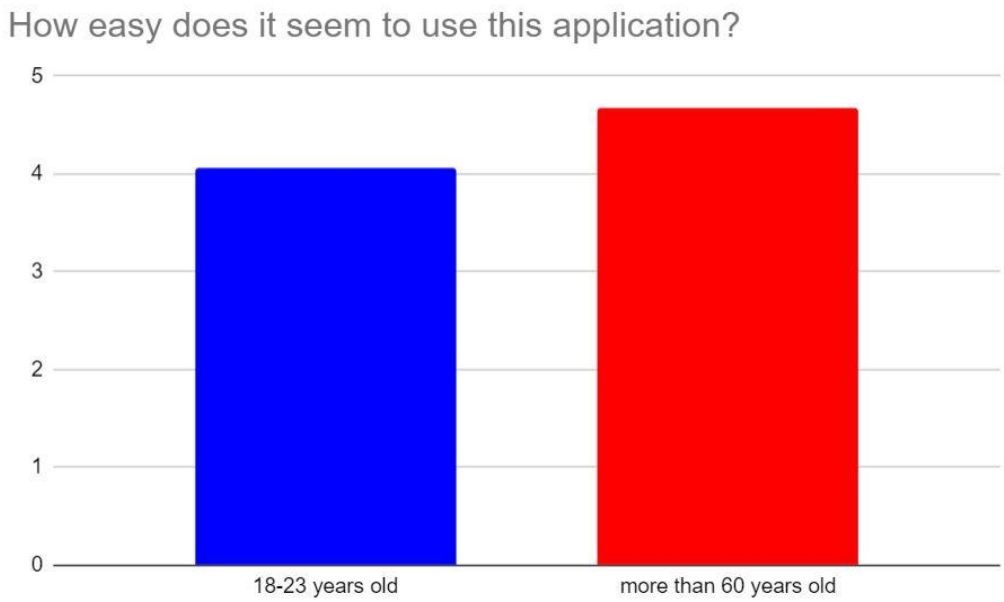

Figure 4.3 Survey of an easy to use application

In Figure 4.3, the question asks the user to rate the applicable use of the application. In the right bar, the elderly have agreed that the uHome application might be easy to use from the provided screenshot of the application. The application probably has the noticeable transition between page and easy to understand the step with the score of 4.6. The first group, teenagers, might want to use the real application to know the process of the application, so the applicants reduce some points in this area.

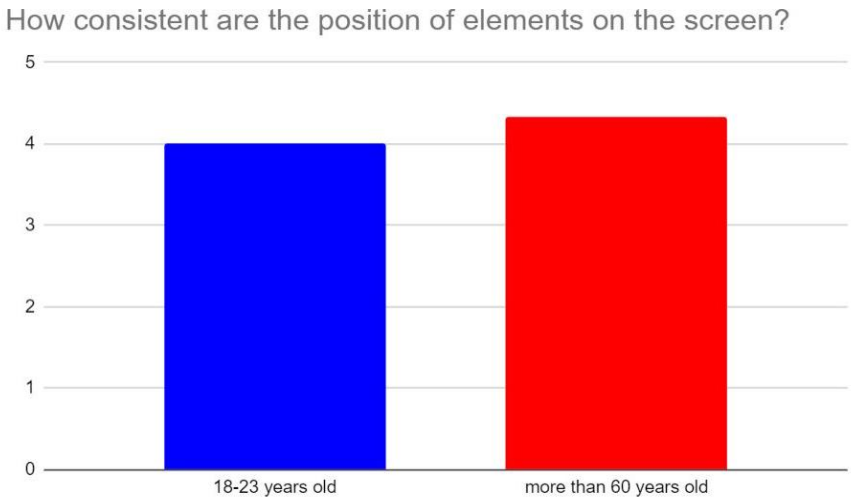

Figure 4.4 Survey of the consistent of the elements

The next question is to let the user evaluate the consistency of all elements on the screen whether the elements appear in the correct position or not. In Figure 4.4, the first group of survey applicants, who age between 18 to 23 years old gives the average score of 4 and the elderly in the second group gives the average of 4.4 which is higher than the teenage for 0.4.

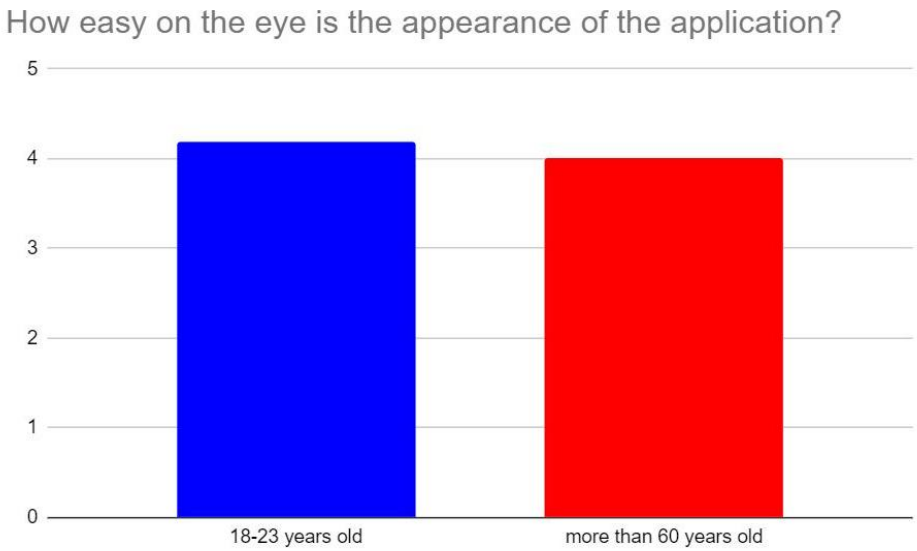

Figure 4.5 Survey of an easy for eye focusing

In Figure 4.5, both two groups of participants agree that the application has a good organization of the elements to be easier for the eye to see. While the first group is giving the average of 4.2, the elderly gives the 4 points to this question. Some applicants say that the character on the screen is on the edge of the screen and it is hard to understand what it is indicating on the page.

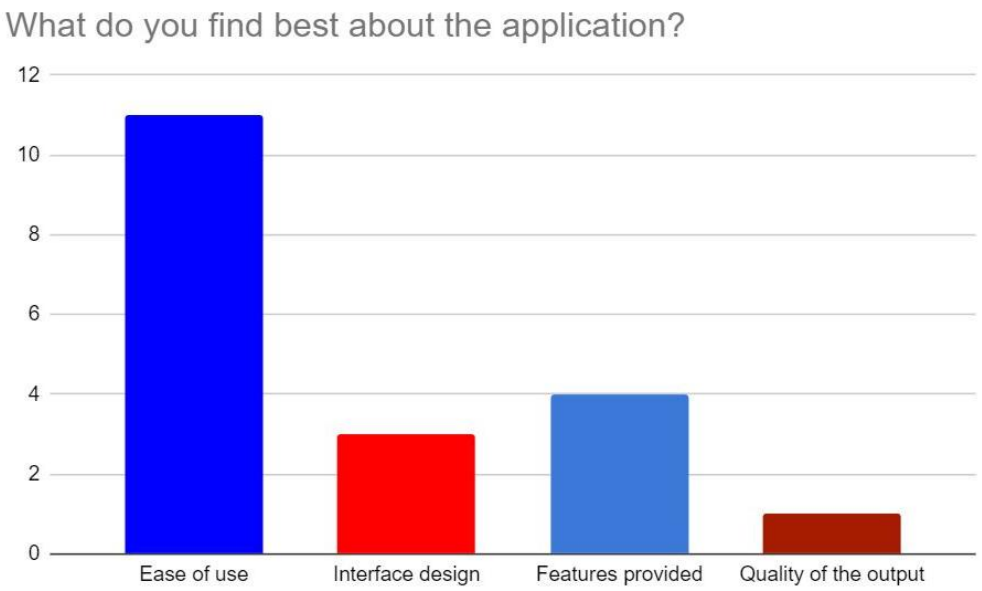

Figure 4.6 Survey of the best part of the application

The last question in the survey asks the applicants to state the best part of the application in the four aspects with the combining in two groups of the applicants. In Figure 4.6, the first aspect is the ease of use which has been ranked as the majority satisfied by the users. The higher ease of use indicates the application is much easier to use. The other aspects have been chosen with a few applicants. As a result, the uHome mobile application provides the user friendly application for the user whether for the teenager or the elderly.

4.1.2 Scenario Testing

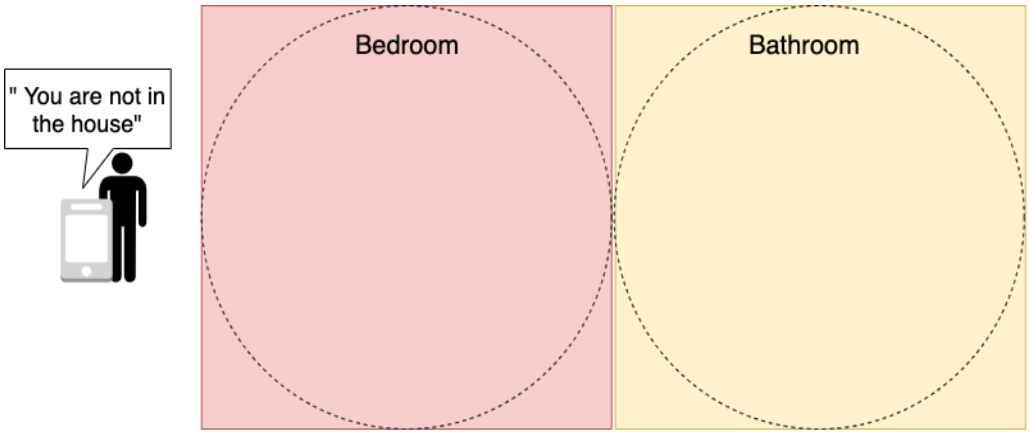

Figure 4.7 Not in the house scenario

In the case that the user is not detected by Estimote Location Beacons in any room, the system, mobile application, will alert with the message of “You are not in the house” as shown in Figure 4.7. Moreover, the current location of this user is also shown in the mobile and web application as unknown.

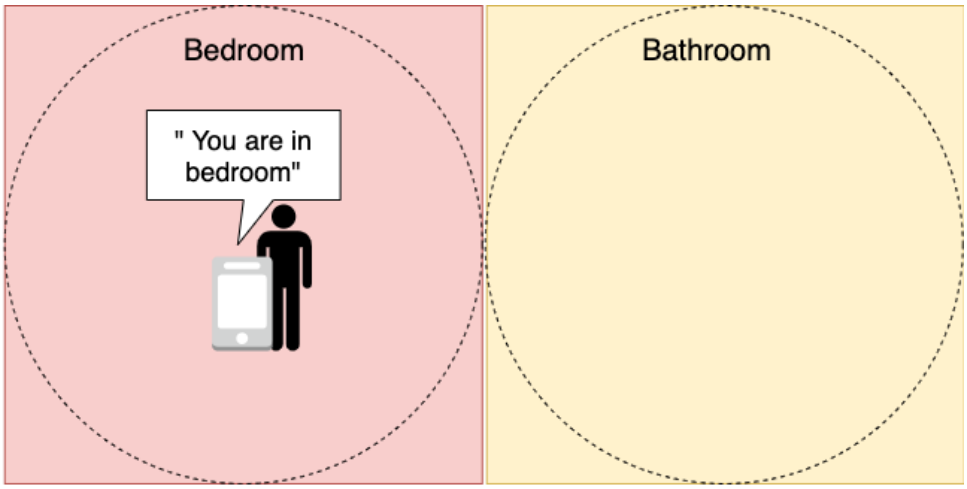

Figure 4.8 In the room scenario

If the user enters a room as shown in Figure 4.8, the Estimote Location Beacon in that room will be detected and the mobile application will alert the user with the message “You are in the bedroom”. Also, the current location shown in the mobile and web application will be changed.

To evaluate, the duration of detected time has been measured to find the average time to perform between the Estimote and the system.

Table 4.1 Record in the opened room on running application

Rounds	Amount of time delay entering the room (second)	Amount of time delay exiting the room (second)
1	13	33
2	34	23
3	8	28
4	70	30
5	25	26
6	34	28
7	30	20
8	5	31
9	4	26
10	29	22
Sum	252	267
Mean	25.2	26.7

From table 4.1, the report has shown that the time to detect the Estimote when the user is using the application is various. It could take from at least 4 seconds to 70 seconds to detect that the user is entering the room. This could be said that the entering event has the error in the signal and has no formal pattern. The entering time has been calculated to find the average time to be 25 seconds. However, the Estimote takes less time which has the same pattern of time to detect the leaving event. The Estimote will alert the user after the user has left the room or the Estimote range for 20 to 33 seconds and the average time is 26 seconds.

Table 4.2 Record in the opened room on application in background

Rounds	Amount of time delay entering the room (second)	Amount of time delay exiting the room (second)
1	14	25
2	16	23
3	10	36
4	14	52
5	19	31
6	47	14
7	50	62
8	13	54
9	12	43
10	49	58
Sum	244	398
Mean	24.4	39.8

From table 4.2, the time duration has been recorded to find the average time and the difference in the performance of the Estimote while not using the application or running in the background. From the first data column, the Estimote can detect the user's activity around 14 seconds and there are 3 rounds that have the record of approximately 48 seconds. The average time of detecting the user is 24 seconds while the user is not using the application. On the other hand, the leaving event tends to have a variety of time. The record ranges from 14 seconds to 62 seconds with the average of 39 seconds to perform.

Table 4.3 Record in the closed room on application in background

Rounds	Amount of time delay entering the room (second)	Amount of time delay exiting the room (second)
1	4	37
2	4	21
3	11	25
4	3	25
5	12	23
6	5	16
7	28	15
8	2	17
9	2	3
10	22	14
Sum	93	196
Mean	9.3	19.6

From table 4.3, the system has been measured to find the time to perform the Estimote in the closed room and outside the room. The mean value of the entering event is 9 second with the least value of 2 seconds. In the same way, the time that Estimote can detect the leaving activity is around 19 seconds which is less than the time to detect in the above cases. The table 4.1 and 4.2 demonstrate the time when the Estimote works in the open room and the result shows that the Estimote can detect more accuracy in the specific area. The Bluetooth signal, in this case, has less space to travel and it can detect another signal easier.

Table 4.4 Record of comparing actual and recorded time

7. Actual and recorded time spent in the room					
Rounds	Entered	Exited	Actual time (minutes)	Recorded Enter Time	Recorded Exit Time
1	11:23	11:28	00:05	11:23	11:30
2	11:31	11:42	00:11	11:31	11:42
3	12:05	12:17	00:12	12:05	12:17
4	12:18	12:35	00:17	12:18	12:35
5	12:37	12:55	00:18	12:37	12:37

Recorded time spent (minutes)	Difference (Recorded - Actual)
00:07	+00:02
00:11	00:00
00:12	00:00
00:17	00:00
00:00	00:00

From table 4.4, the system has been tested to measure the accuracy of the system in order to collect the time when the user stays in the room and compare with the actual time in real events. The test has been conducted in 5 rounds. The system will start to count when the users are detected inside the room by the Estimote beacon. To clarify, the measured time will have a difference in unit of second, but it might produce the same duration of the unit of minutes. The first round starts at 11:23 a.m. and ends at 11:28 a.m. and the duration is 5 minutes in the actual time. However, the system starts at the same time but ends at 11:30 a.m. which exceeds the actual time for 2 minutes. The system has been tested again to find what is the error in the system. As a result, the other 4 rounds produce no error and have functioned properly. From the 4 out of 5 rounds, the uHome system can surely get the actual user average time and use it to calculate for using in the danger alert scenario.

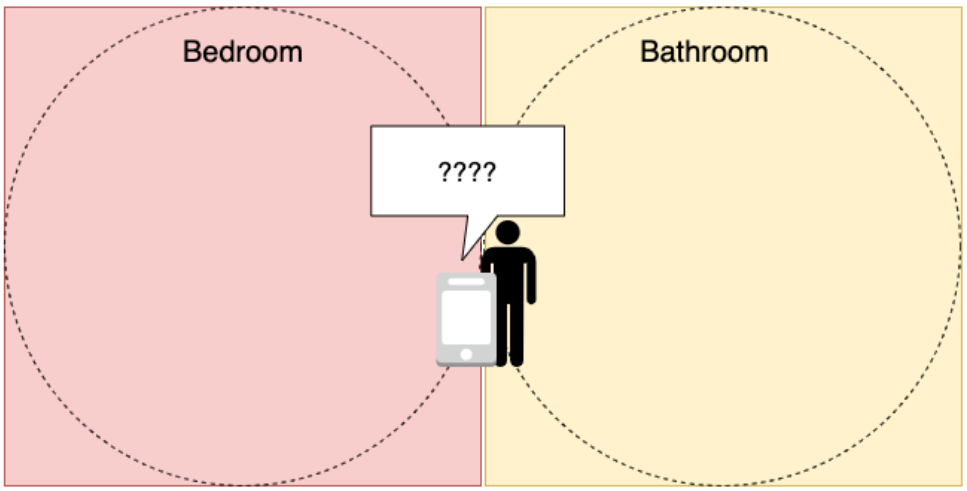

Figure 4.9 Overlap scenario

It will not be a case if the users stand between two rooms. This is because when the admin inserts the room size from the web application, the radius of the Estimote will be calculated to perform accurately. In Figure 4.9, the overlap scenario could not happen. The Estimote signal will be limited by the room size and will not spread into another room.

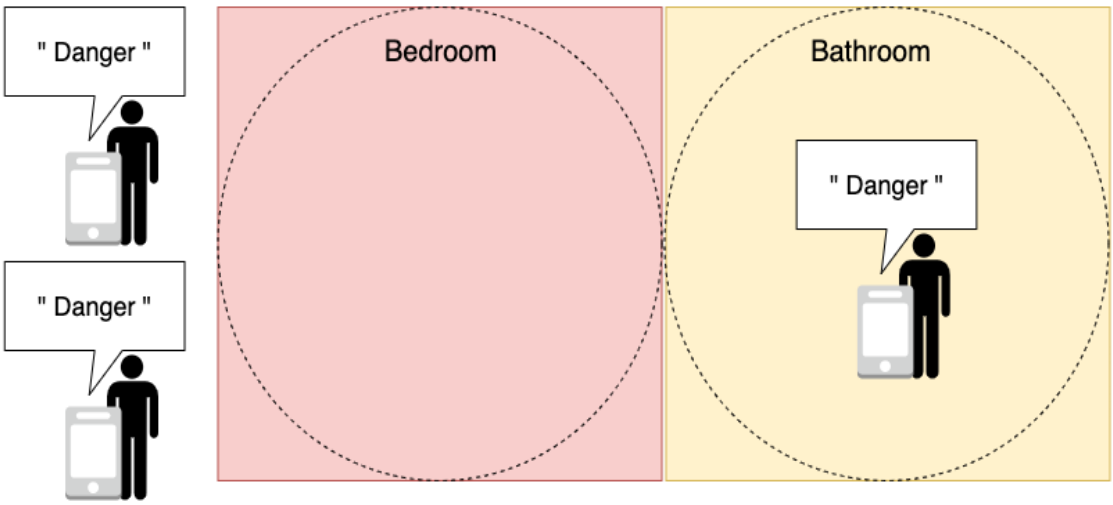

Figure 4.10 Danger alert scenario

In Figure 4.10, since the admin sets a duration that a specific elderly in the house should be in the bathroom, the system should be able to alert everyone under this house with a “Danger” message.

Table 4.5 Record of alert activity

Rounds	Entering time	Alert time	Error
1	02:38	02:43	00:00
2	02:44	02:49	00:00
3	02:51	02:56	00:00
4	03:00	03:05	00:00
5	03:08	03:13	00:00
Sum			00:00
Mean			00:00

The table 4.5 shows the record of the alert activity. When the elderly enter the room and that room has been set the time to not stay more than 5 minutes, the member in that house must receive the alert message from the system that there is someone inside that room for example, bathroom. In the second column, when the user enters the bathroom at 02:38 p.m., at 02:43 p.m., the system will begin sending the alert if the Estimote can detect the mobile application of the entered user in its range. The test has been recorded 5 times and it produces no error in this case.

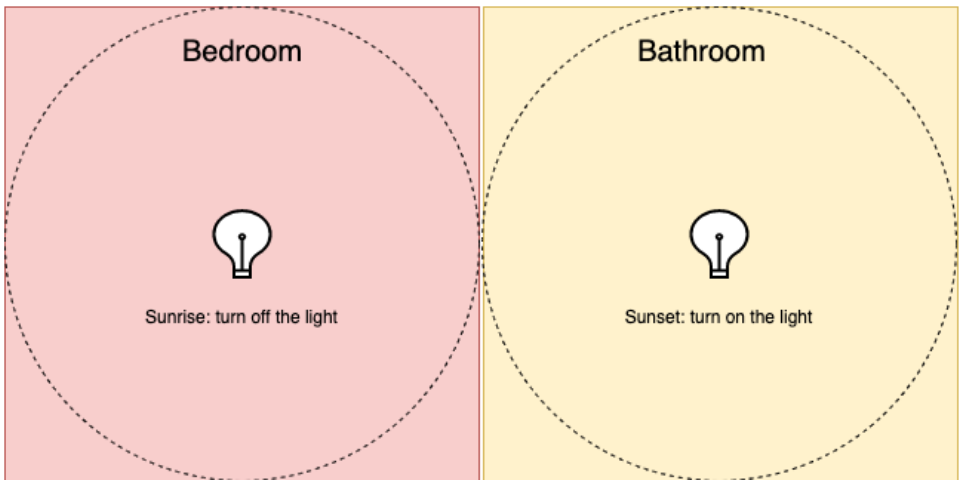

Figure 4.11 Routine scenario

In Figure 4.11, in the web application, admin can add routines which can be according to sunrise, sunset and custom time in order to turn on or off lights in the house. Therefore, the system should be able to automate all the lights according to preset routines.

Table 4.6 Record of routine's accuracy

Rounds	Light	Set time	Action	Achievement
1	Philip Hue 2	13:40	Off	Success
2	Philip Hue 2	13:41	On	Success
3	Philip Hue 2	13:42	Off	Success
4	Philip Hue 2	13:43	On	Success
5	Philip Hue 2	13:44	Off	Fail
6	Philip Hue 2	Sunset	On	Success
7	Philip Hue 2	Sunrise	On	Success

The system has been tested to check and measure the accuracy of the system when the admin has already set the routine of the light. From table 4.6, there are two scenarios to test including fixed time and sunrise and sunset time. The Philip Hue 2 has been set to turn off and turn on alternately periodically every minute. By investigating the light activity, the system works correctly on the first four rounds but it does not turn off the light on the fifth round. It could be explained that the system might need a few times to handle the request and do an activity. However, there is no error on the sunset and sunrise routine

Table 4.7 Record of Philip Hue's performance time

Rounds	On (second)	Off (second)	Change color (second)
1	3	3.5	6
2	3.5	3.69	3.7
3	3.5	4.1	3.6
4	3.5	3.9	3.5
5	3.4	4	3.7
6	3.3	3.9	3.7
7	3.6	3.9	3.5
8	3.5	3.7	4.2
9	3.5	3.7	2.7
10	4.1	3.7	3.4
Sum	34.9	38.09	38
Mean	3.49	3.809	3.8

From table 4.7, the system's performance has been recorded to find the average time to respond to the request from the mobile application to use the Philip Hue light. When the user turns on the light from the mobile application, it requires the average of 3.5 seconds to totally perform the action from turning on to highest brightness. Additionally, the system takes 3.8 seconds to dim down the light and certainly turn off. Changing the light color requires a similar amount of time to perform comparing to turn on and turn off cases.

4.2 Result

From the evaluation method mentioned above, the uHome platform has succeeded in completing the objective of the project. The uHome mobile application has been approved satisfying from the users and it is considered as the user-friendly application for the main target of the elderly. Moreover, the scenarios of the system have been tested to find the average time to respond to the Estimote and the system. The measured time by the system and the actual time have been compared to evaluate the accuracy of the system in order to collect the user data to use in scenario testing.

CHAPTER V
CONCLUSION AND FUTURE WORK

5.1 Discussion

5.1.1 Camera Ario

After observing and discussing the use of the Ario, the purpose of this camera is not clear enough to use in the uHome platform. To track the movement or the location of the user, the camera has a different angle and different distance to measure instead of just sending the current circumstance to the server. The alternative devices and ways have been discovered to make a precise decision to the user's activity. The application of Estimote Location Beacon can be used to detect the current location of users. The use of Estimote has been concerned. In order to use the Estimote in the application, the user must carry the smartphones all the time to detect the beacon. However, the project' scope is that the user has to carry the smartphones all the time and that will cause no problem to the use of Estimote.

5.1.2 Alexa

Alexa is used as another interface that users can command to perform some action, such as to turn on/off the lights, find the missing phone and notify warning messages. The problem is that Amazon allows us to manipulate Alexa only through the Alexa skill console. Moreover, our platform needs account linking with Alexa due to the authentication of the system.

5.1.3 Database

The main database that was initially used in order to store data regarding the system was MongoDB. The reason MongoDB was chosen is due to its ease and flexibility. However, using a NoSQL database made it more difficult to implement the system due to the fact that the system's components are codependent on one another and using NoSQL database means that these relationships had to be dealt with manually and became more workload. As a result, a decision was made to switch to

use MySQL, which is a SQL database and is designed to work as a relational database and works well when dealing with data that are dependent on data from another set. Furthermore, due to the fact that the system will be dealing with devices that have different setups and parameters and MySQL tables are quite rigid and fixed, a mix of MySQL and MongoDB was implemented in order to get the best of both worlds. MySQL was used as the main backbone and relational structure for the system, while MongoDB was used to store other information such as device information. This decision has led to the system being able to store information easily without compromising the relational nature of the data in the system.

5.1.4 Estimote

With the implementation of the Estimote in the application, the first problem to discuss is that the Bluetooth range of the smartphones is different in each model. The Estimote beacon will normally spread the signal with the defined range that the user can configure on the web application. While the application is starting scanning to detect beacon, the smartphone will send the Bluetooth signal out of the device. The process time of detecting the beacon will be different due to the smartphone's Bluetooth signal. The problem of time to process affects the performance of the application to perform the request to the backend and to send the notification. For example, one device must take 10 seconds to detect a beacon and send the start timer request, but the other older Android device can take up to 15 seconds to do an operation. This means the time duration in each room might not be accurate.

5.2 Conclusion

uHome is a system platform where devices are added to the platform in order to transform a house into a smart home. The key difference between uHome and other smart home platforms is the ability to track the user and trigger send notification, which was especially designed to help elderly people that have to live at home alone.

The limitation for uHome platform is when a user is in the middle of 2 rooms, Estimote is unable to detect the user. Moreover, the range of signal emitted by

Estimote is setted within the room, so corners of the room are out of range for detection.

5.3 Future Works

5.3.1 Alexa can be fully implemented into the uHome platform as another interface.

5.3.2 Responsive web application that can work on mobile

REFERENCES

[1] Shah A. (2018). Smart Home Growth Rates Indicate 2019 Will Be a Significant Year. [online]. Available from: https://www.valuewalk.com/2018/12/smart-home-industry/

[2] Stroud F. (n.d.). What is a Smart Home? Webopedia Definition. [online]. Available from: https://www.webopedia.com/TERM/S/smart-home.html

[3] Arlo (a.d.). Arlo HD Security Camera System User Manual [online]. Available from: https://www.arlo.com/en-us/images/Documents/Arlo_Wirefree/arlo_wirefree_um.pdf

[4] Fisher, T. (2019). What is Firmware. Lifewire [online]. Available from: https://www.lifewire.com/what-is-firmware-2625881

[5] Martin, J and Finnegan, M (2019). What is IFTTT How to Use IF This Then That Services. Computerworld [online]. Available from: https://www.computerworld.com/article/3239304/what-is-ifttt-how-to-use-if-this-then-that-services.html

[6] Signify Holding (n.d.). How it works [online]. Available from: https://www2.meethue.com/en-au/how-it-works

[7] IP Location (2018). How does ZigBee Work? [online]. Available from: https://www.iplocation.net/zigbee

[8] Kastrenakes, J. (2019). Philips Hue bulbs now come with Bluetooth, so you don't need a hub. Vox Media [online]. Available from: https://www.theverge.com/2019/6/27/18715370/philips-hue-Bluetooth-smart-bulbs-no-hub

[9] Amazon (n.d.). Request and Response JSON Reference. Amazon. [online]. Available from: https://developer.amazon.com/docs/custom-skills/request-and-response-json-reference.html

[10] Ahmad S, Rouyu PL, Hussain MJ. Never Lose! Smart Phone based Personal Tracking via Bluetooth. International Journal of Academic Research in Business and Social Sciences [Internet]. 2014;4(3). Available from:

https://pdfs.semanticscholar.org/73da/ddb1e43f985d318aef5ec89e88a3f9fe5374.pdf

[11] ScaleGrid (2019). 2019 Database Trends-SQL vs. NoSQL, Top Database, Single vs. Multiple Database Use [online]. Available from: https://scalegrid.io/blog/2019-database-trends-sql-vs-nosql-top-databases-single-vs-multiple-database-use/

[12] MongoDB (n.d.). What is MongoDB? [online]. Available from: https://www.mongodb.com/what-is-mongodb

[13] Beal, V. API - application program interface. QuinStreet. [online]. Available from: https://www.webopedia.com/TERM/A/API.html

[14] Google (n.d.). Video AI - Video Content Analysis. [online]. Available from: https://cloud.google.com/video-intelligence/

[15] Simon (2017). Kotlin on the JVM - Bytecode Generation - What is Kotlin? [online]. Available from: https://kotlinexpertise.com/kotlin-byte-code-generation

[16] Kotlin (n.d.). Using Kotlin for Android Development. [online]. Kotlin. Available from: https://kotlinlang.org/docs/reference/android-overview.html

[17] Kotlin (n.d.). Null Safety [online]. Available from: https://kotlinlang.org/docs/reference/null-safety.html

[18] Horgas A L. (2019). Daily Life in Very Old Age: Everyday Activities as Expression of Successful Living. [online]. Available from: https://watermark.silverchair.com/38-5-556.pdf?token=AQECAHi208BE49Ooan9kkhW_Ercy7Dm3ZL_9Cf3qfKAc485ysgAAAlIwggJOBgkqhkiG9w0BBwagggI_MIICOwIBADCCAjQGCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQMpvR04K9zQLuzd0aDAgEQgIICBZwi6HwGBhAWBuXIRwSTLw6XqDWy9PB2cL1nLYUq_HeSET83Qy9QK97X29RERRInUI28B8Df44ivWRVq-LdUm4_H9rmsl9-zFrK6AmjXCE7k6xY2hRZheAYccakYnRFT1p2o4NYDk0KOs5iP4os85lSVwvIlxjPWr_zGOT9qZdtixEHqCDk8vFl4vBCAW2WMnUcw4zuWMMDWrpmJZspTW40_aAQnoGtCQuTuQGlLh6wiAmkY57bS6WRQjniczQah8_548qVii7RN4UNyRUxnLJYa5vE-oxDo94BMDOcP7hdjc_EjaVhTJ4nr7D0MQESxeXAfeKxmvZyzrokgAW7ov

xjynDS6UhiFfChpCU2xNnM7FP9e-rvsyJTXf9Y4YsLcCTsB1NcQc1isBiED1Mwuih4EqhoYH7-LYx_I2EXzN_UYVaxiWmhxYuOWRLzmhI8dRYVHMZ3tThlHxpN0AuSf_2WabmUOZm-ZZx0H-1pxZO-Ek3N4asptXXIxdvGXV8UP5rSsxB7V8poPtgQoCUzC5CJhHpipTxXeVSJyLE9cUqJHe53kxJaQycKm3KaI6ZT1WqvRmBTUgejBa_HEG1SHP1ipjEZ6adjt8bQX8VvknOzzYDpDj2i28F-JcZjTKCmGCghfAbbnwYJJeOyI9AvaEySBnq3n_L8_uQCYrwrsOWf_DHC8flc

[19] Is there a RESTful API for Estimote Cloud?(2020). Estimote Community Portal [online]. Available from: https://community.Estimote.com/hc/en-us/articles/204233183-Is-there-a-RESTful-API-for-Estimote-Cloud-

[20] Comparison with Other Frameworks — Vue.js (2020). Vuejs.org[online]. Available from: https://vuejs.org/v2/guide/comparison.html

[21] Understanding the Virtual DOM (2020). bitsofcode [online]. Available from: https://bitsofco.de/understanding-the-virtual-dom/

[22] About | Node.js (2020). Node.js [online]. Available from: https://nodejs.org/en/about/

[23] Firebase Authentication [Internet]. Firebase. Available from: https://firebase.google.com/docs/auth

[24] Firebase Cloud Messaging [Internet]. Firebase. Available from: https://firebase.google.com/docs/cloud-messaging

[25] What Is Amazon EC2? - Amazon Elastic Compute Cloud [Internet]. Amazon. Available from: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html

APPENDICES

APPENDIX A
PROJECT SOURCE CODE

- Please click this uHome link to login to our admin website. Note that you will need to visit uHome with your computer as this link is not compatible with mobile devices.
- Please click this uHome_frontend link to see the source code for the web application.
- Please click this uHome_mobile_application link to see the source code for the mobile application.
- Please click this uHome_backend link to see the source code for the mobile application.

APPENDIX B

USER MANUAL

B.1 Web Application

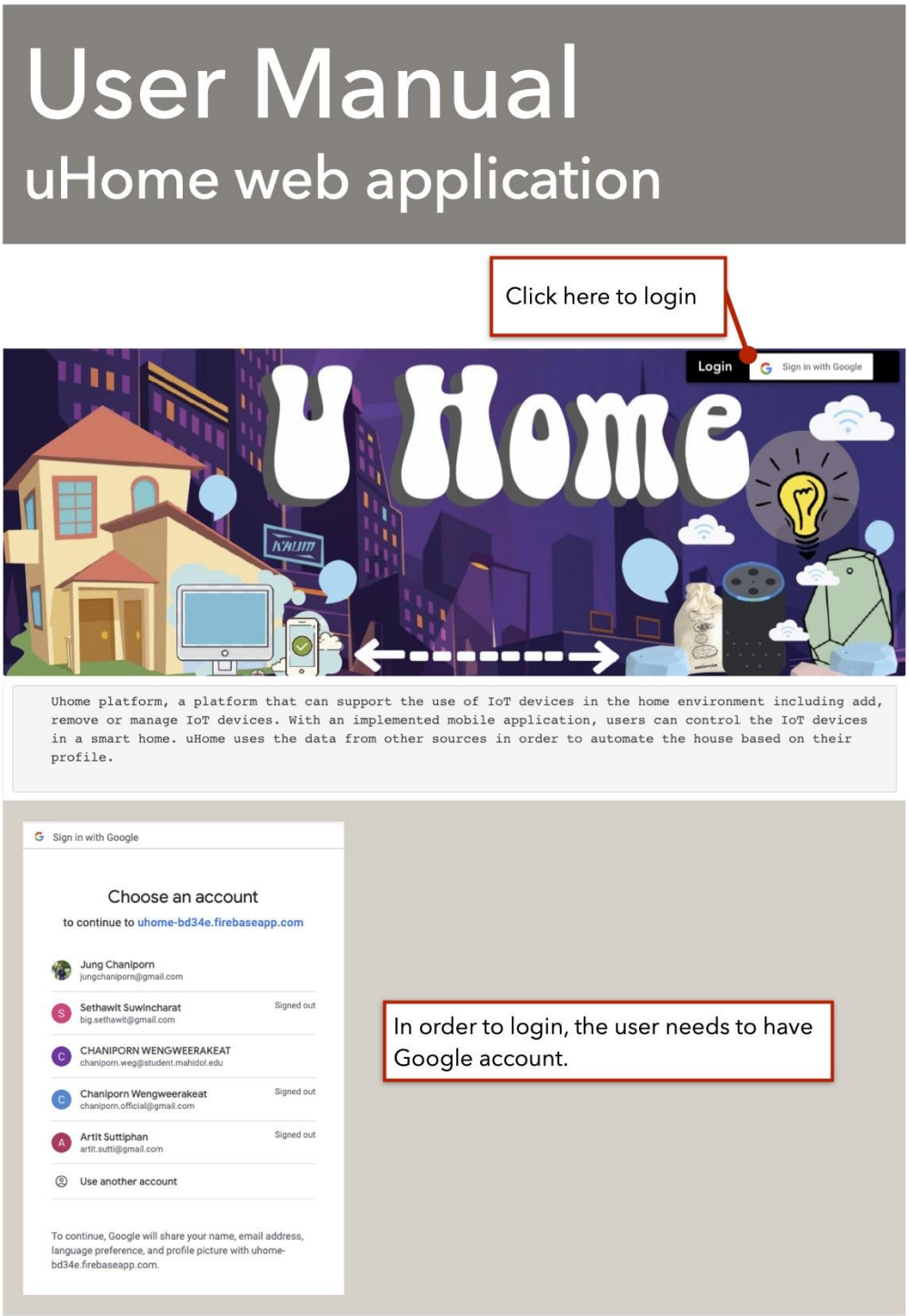

Figure B.1 Web login page

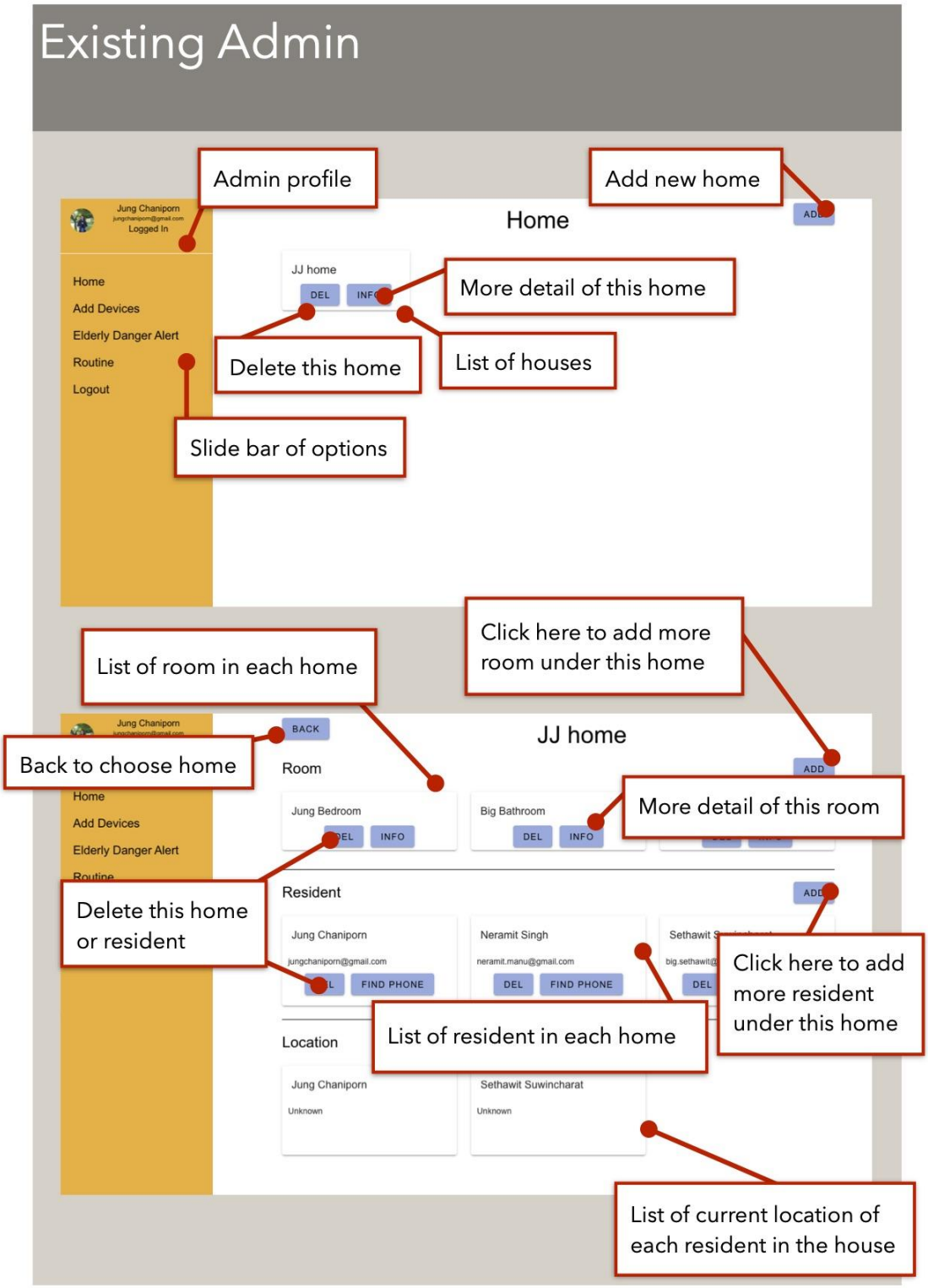

Figure B.2 Default and home page

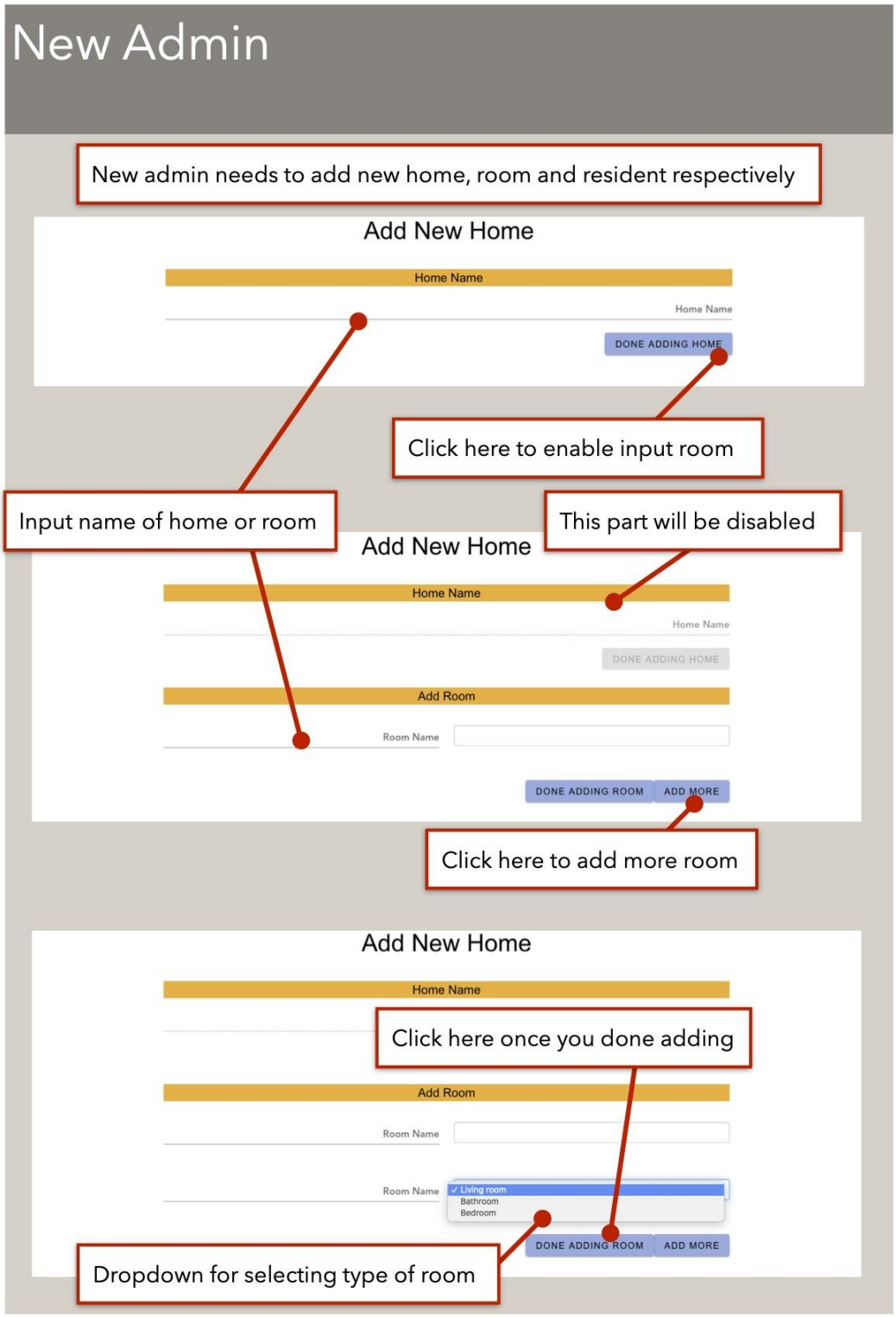

Figure B.3 Add home page

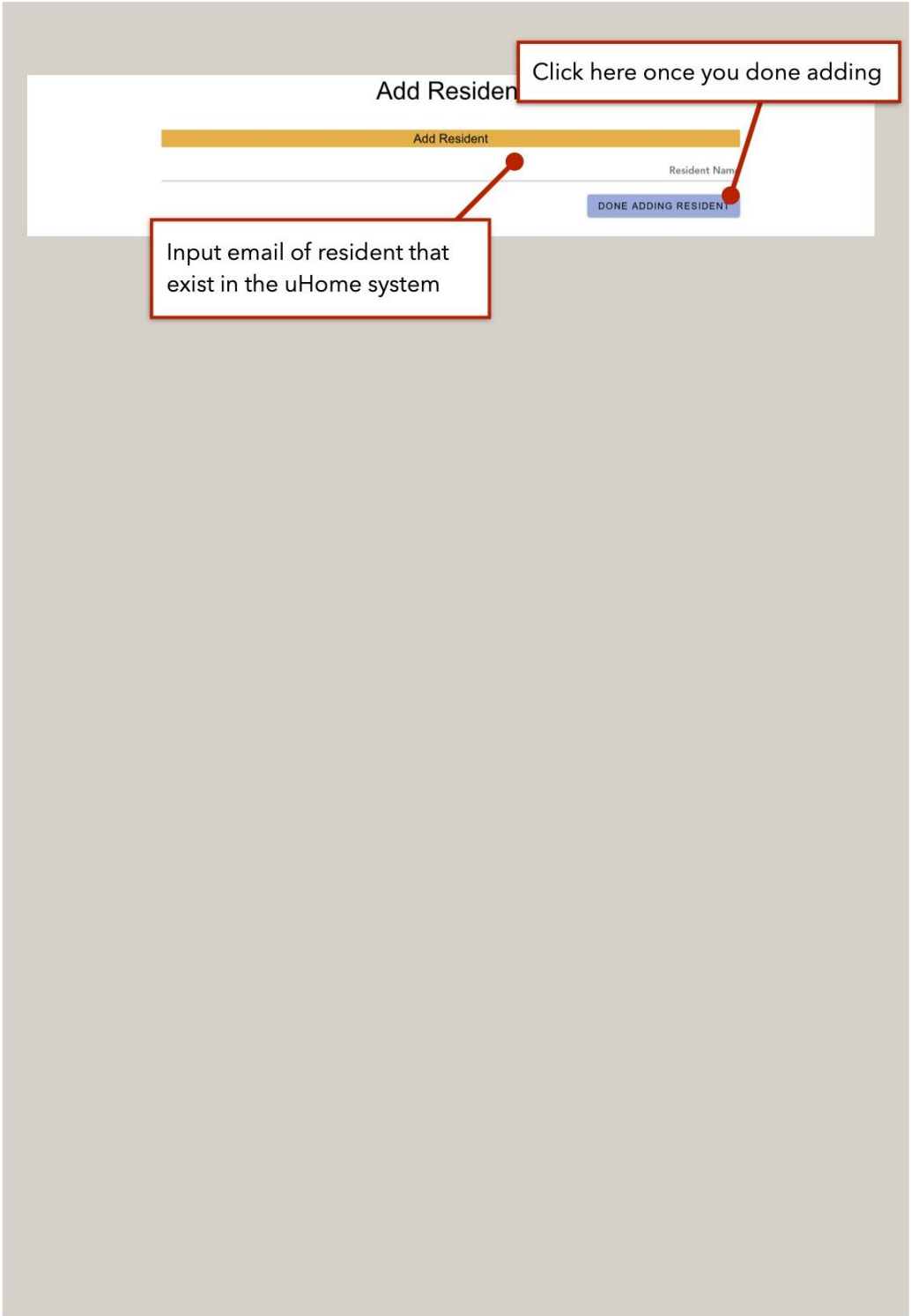

Figure B.4 Add resident page

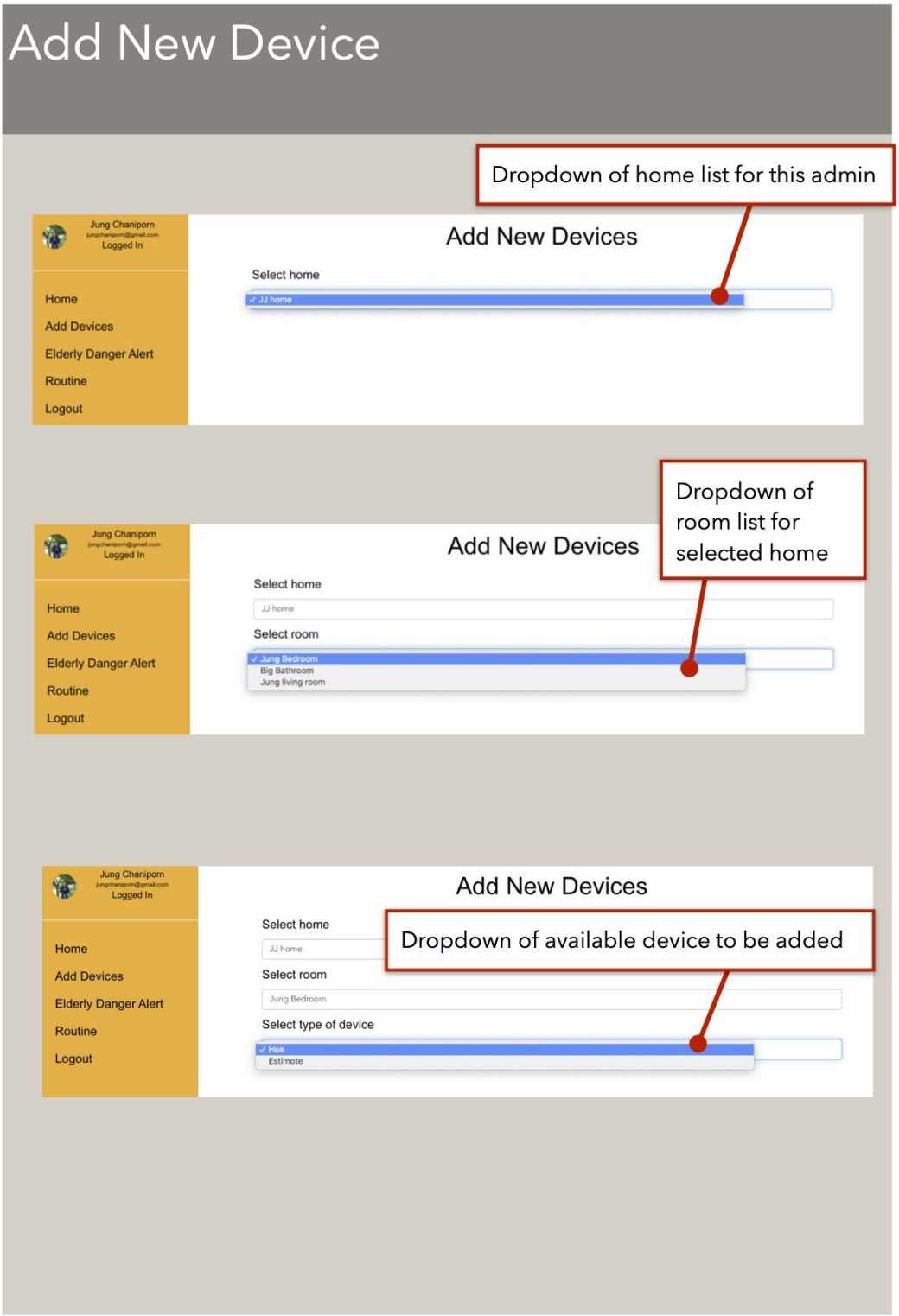

Figure B.5 Add devices page

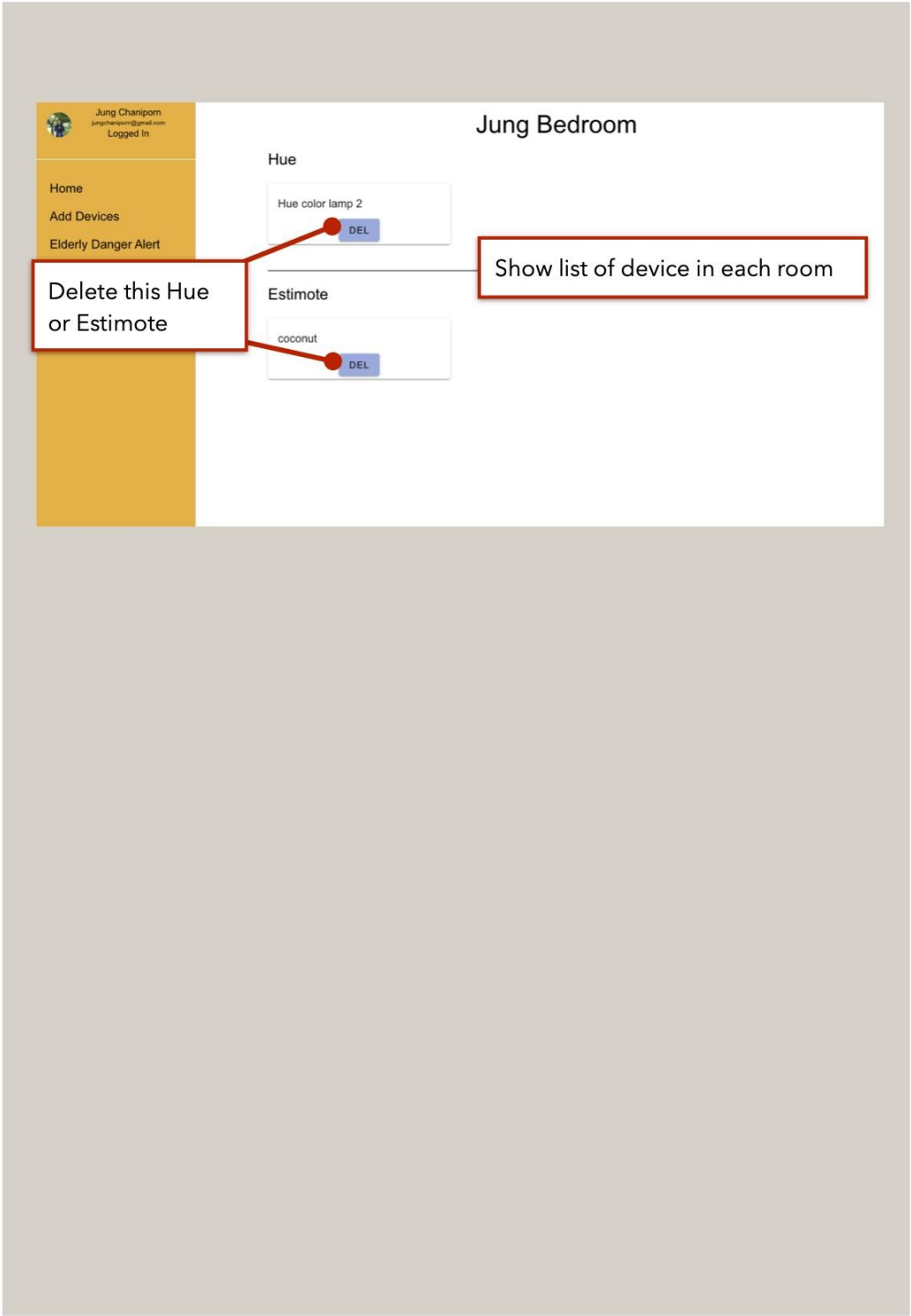

Figure B.6 Device list page

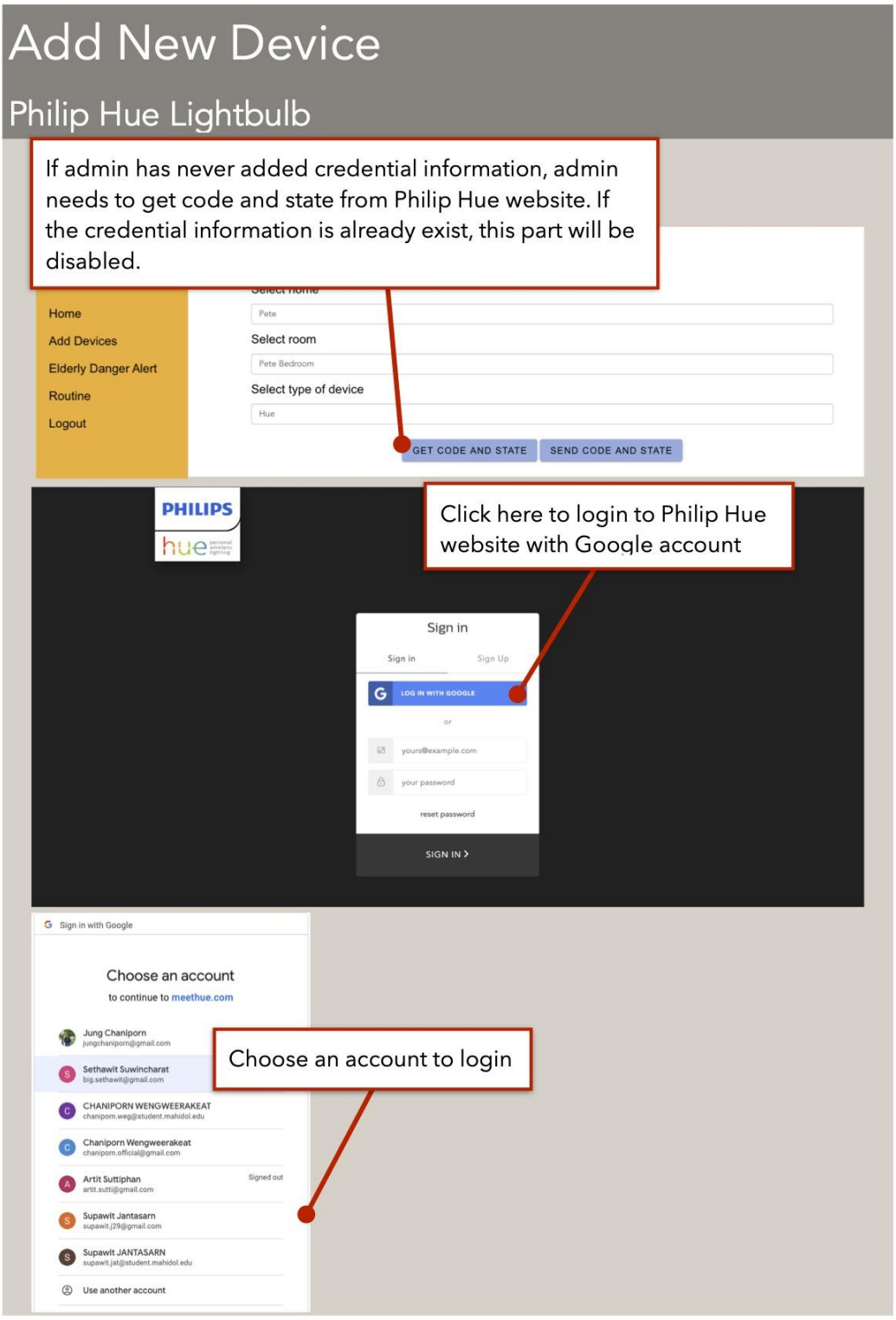

Figure B.7 Add Hue page

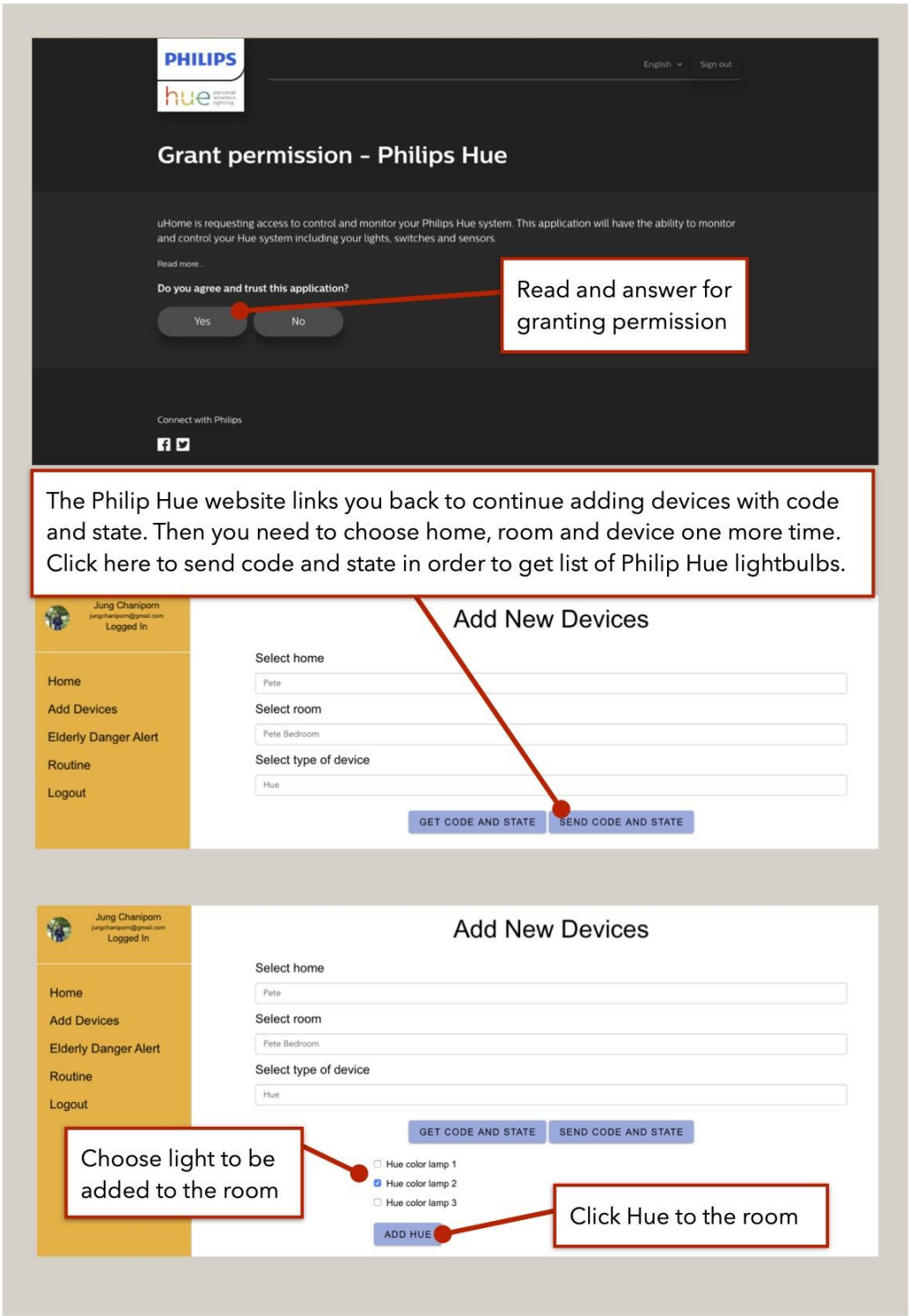

Figure B.8 Grant Hue permission page

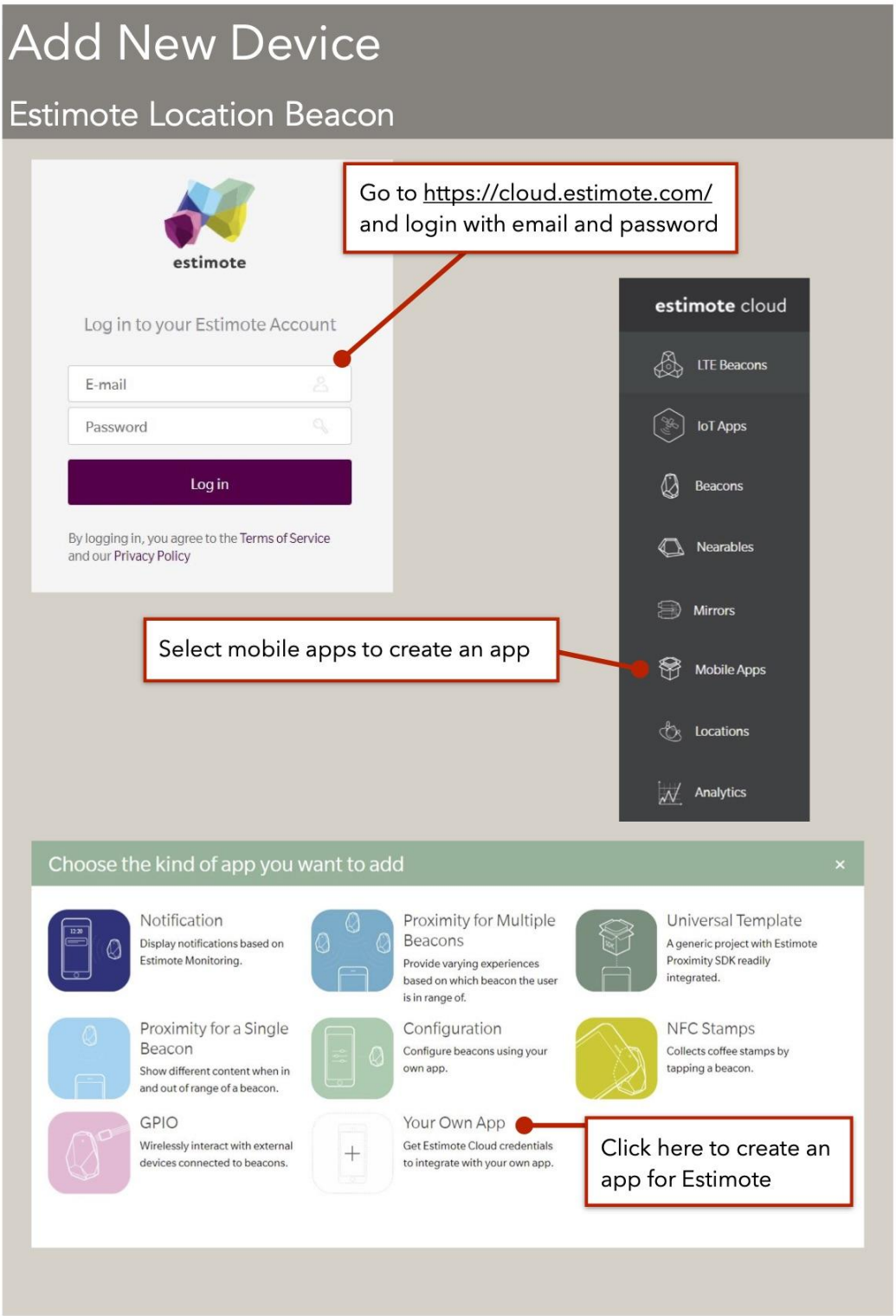

Figure B.9 Login Estimote page

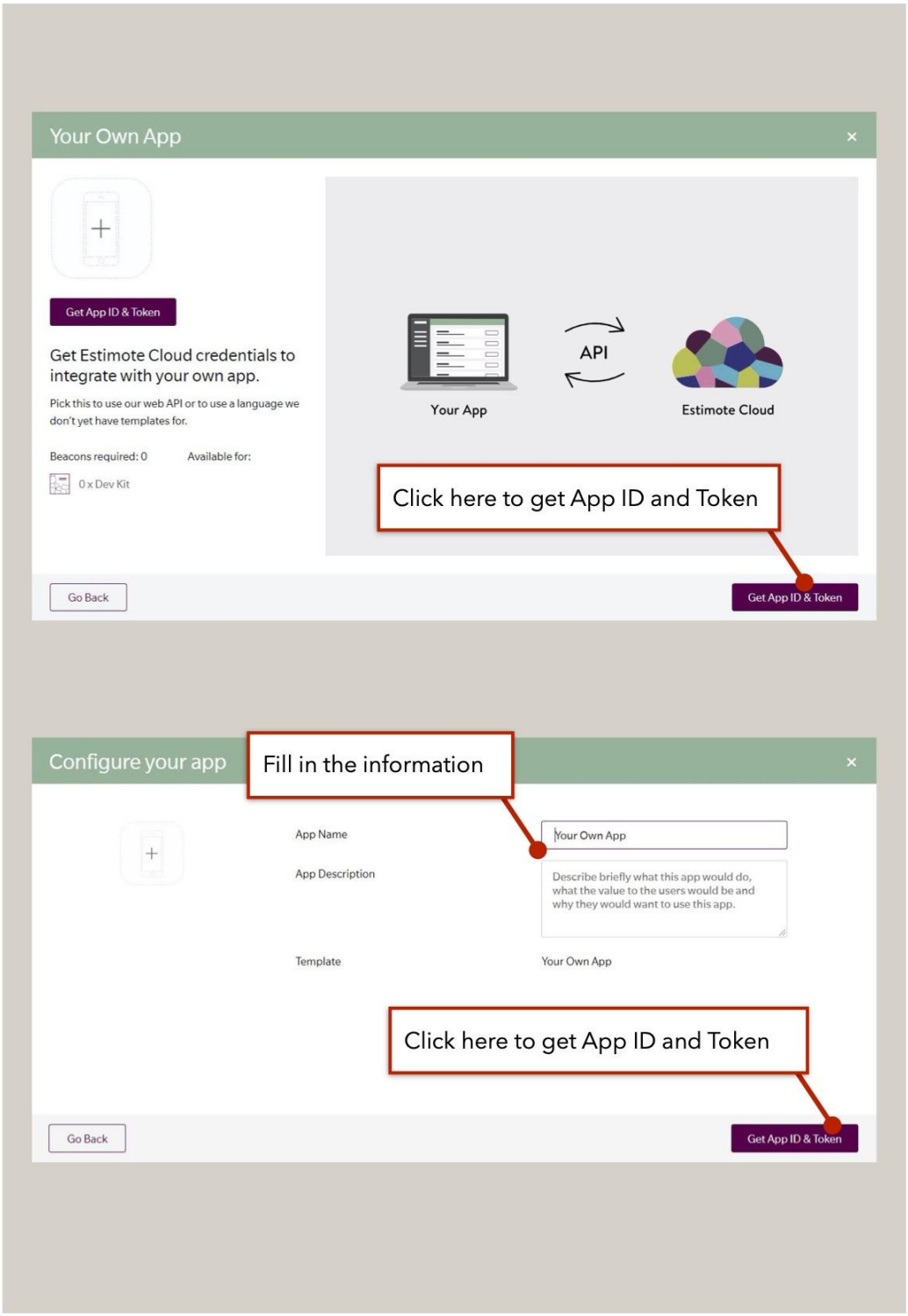

Figure B.10 Create own Estimote application page

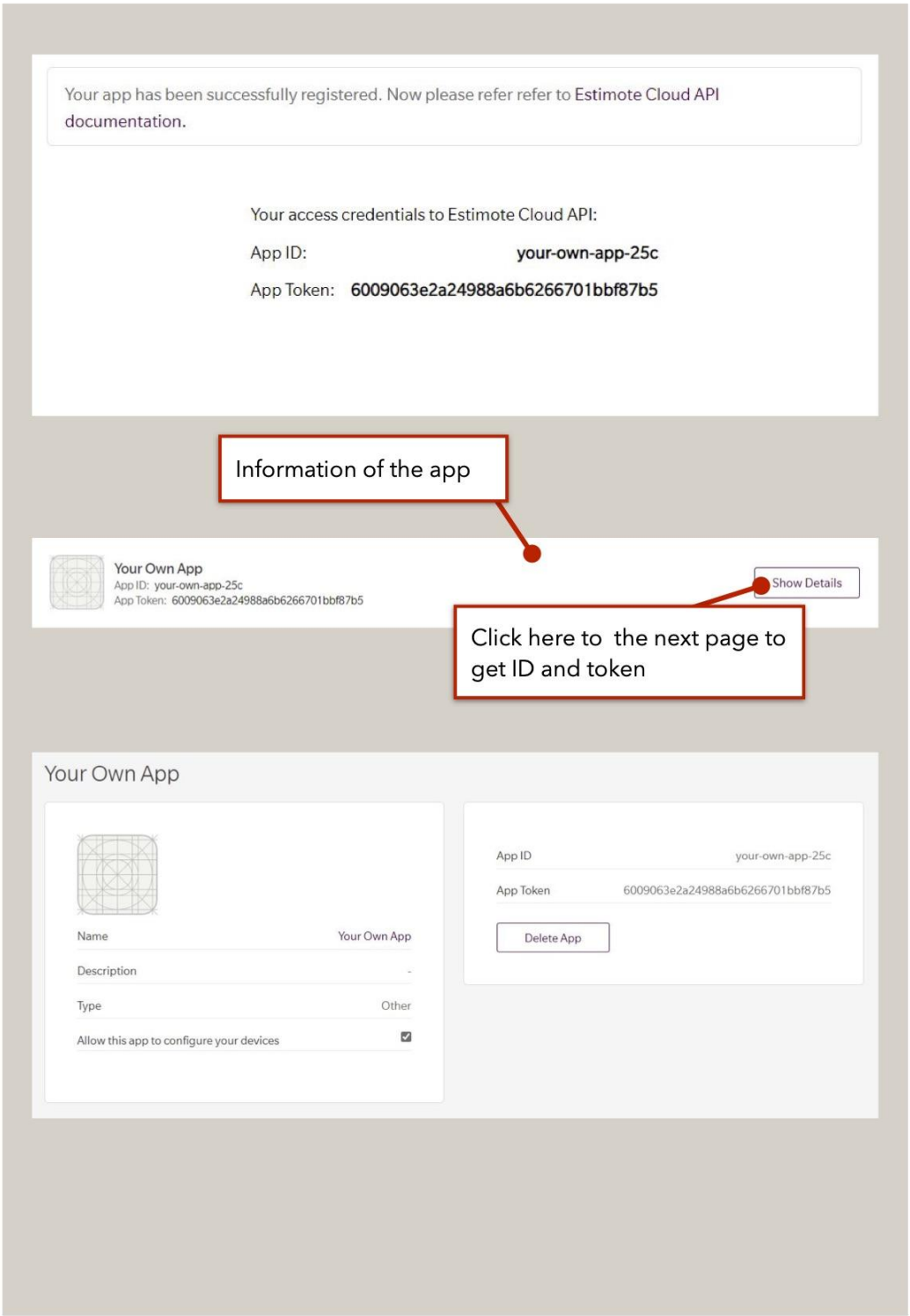

Figure B.11 Success create application page

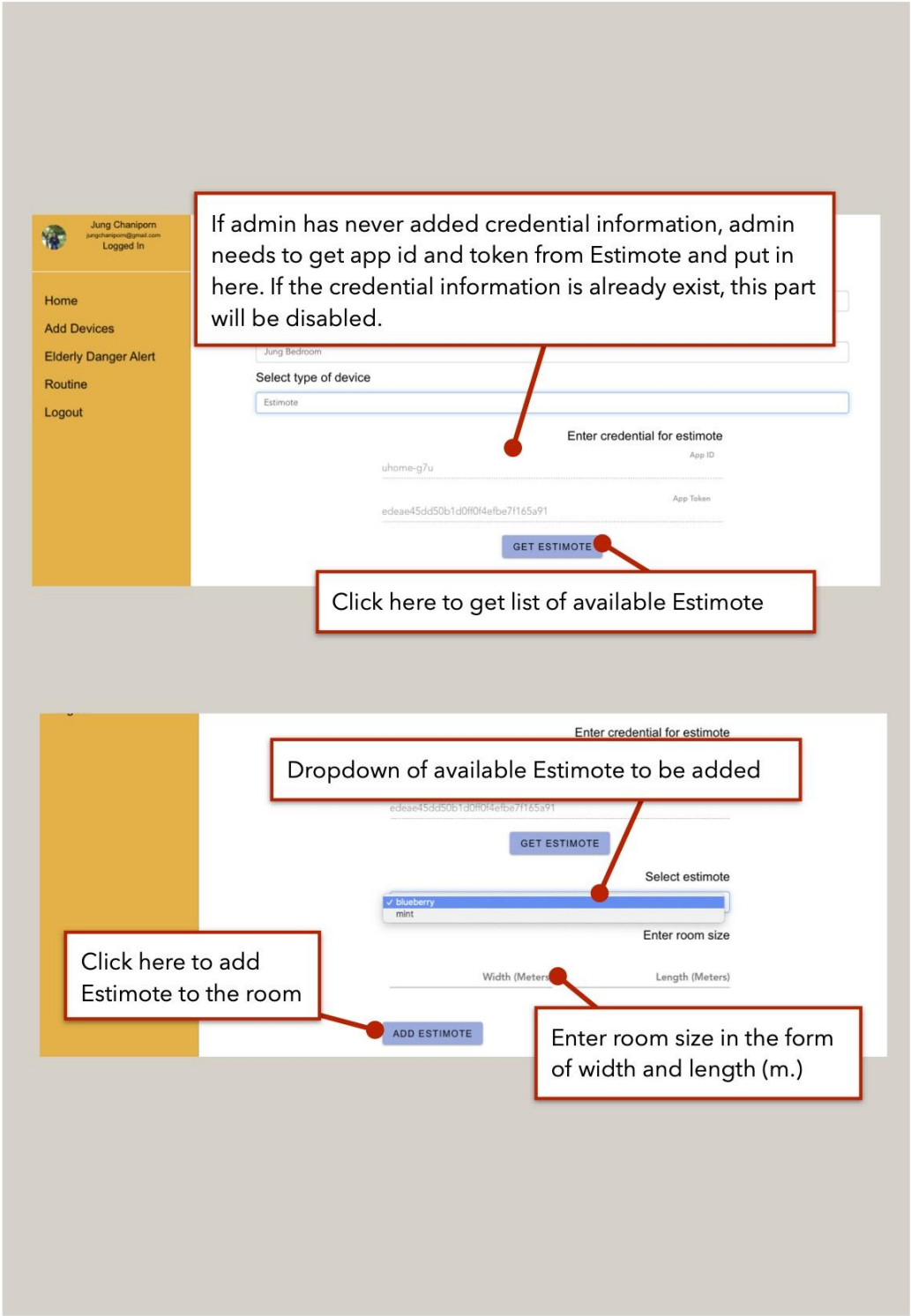

Figure B.12 Get list of Estimote page

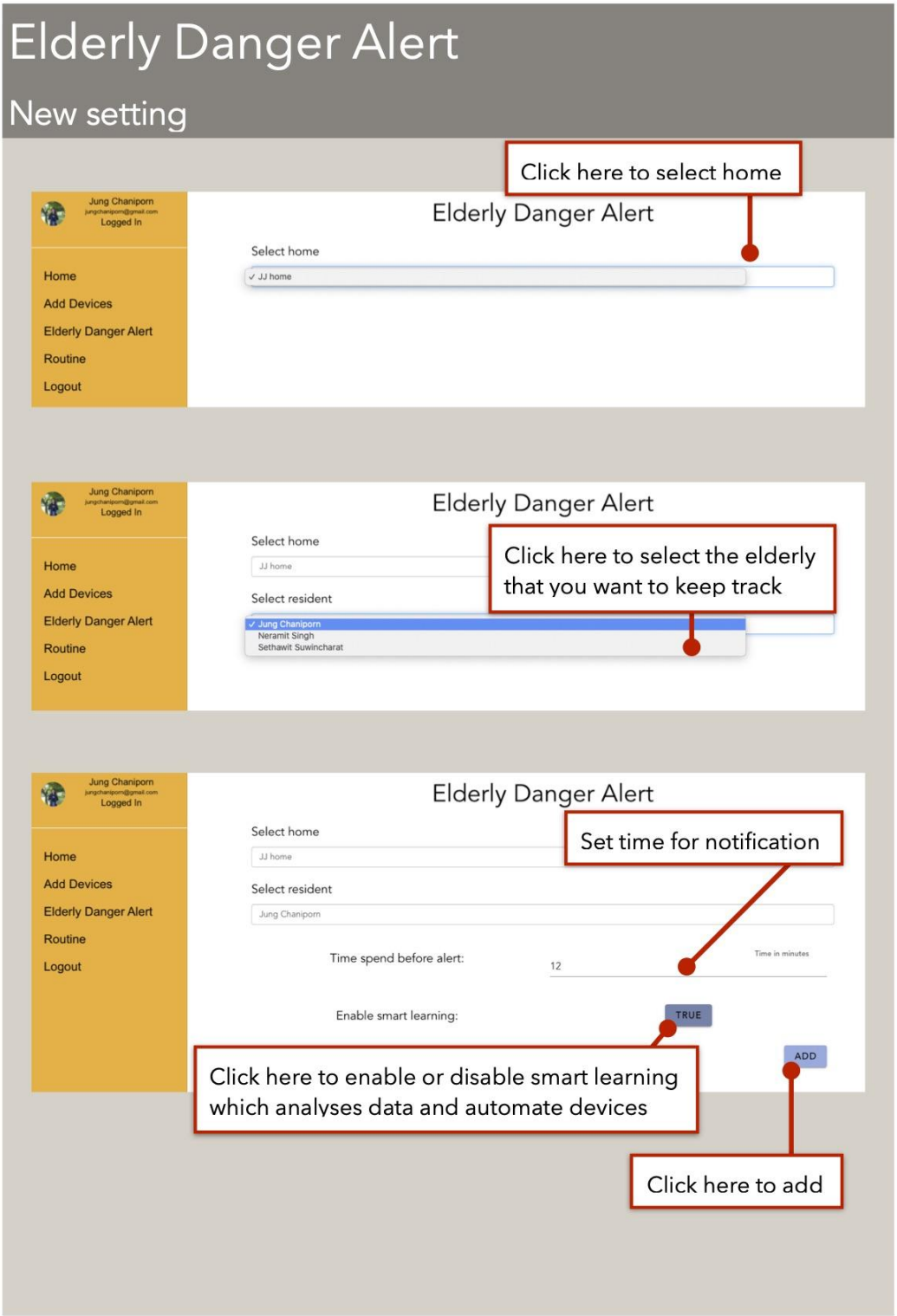

Figure B.13 Danger alert configuration page

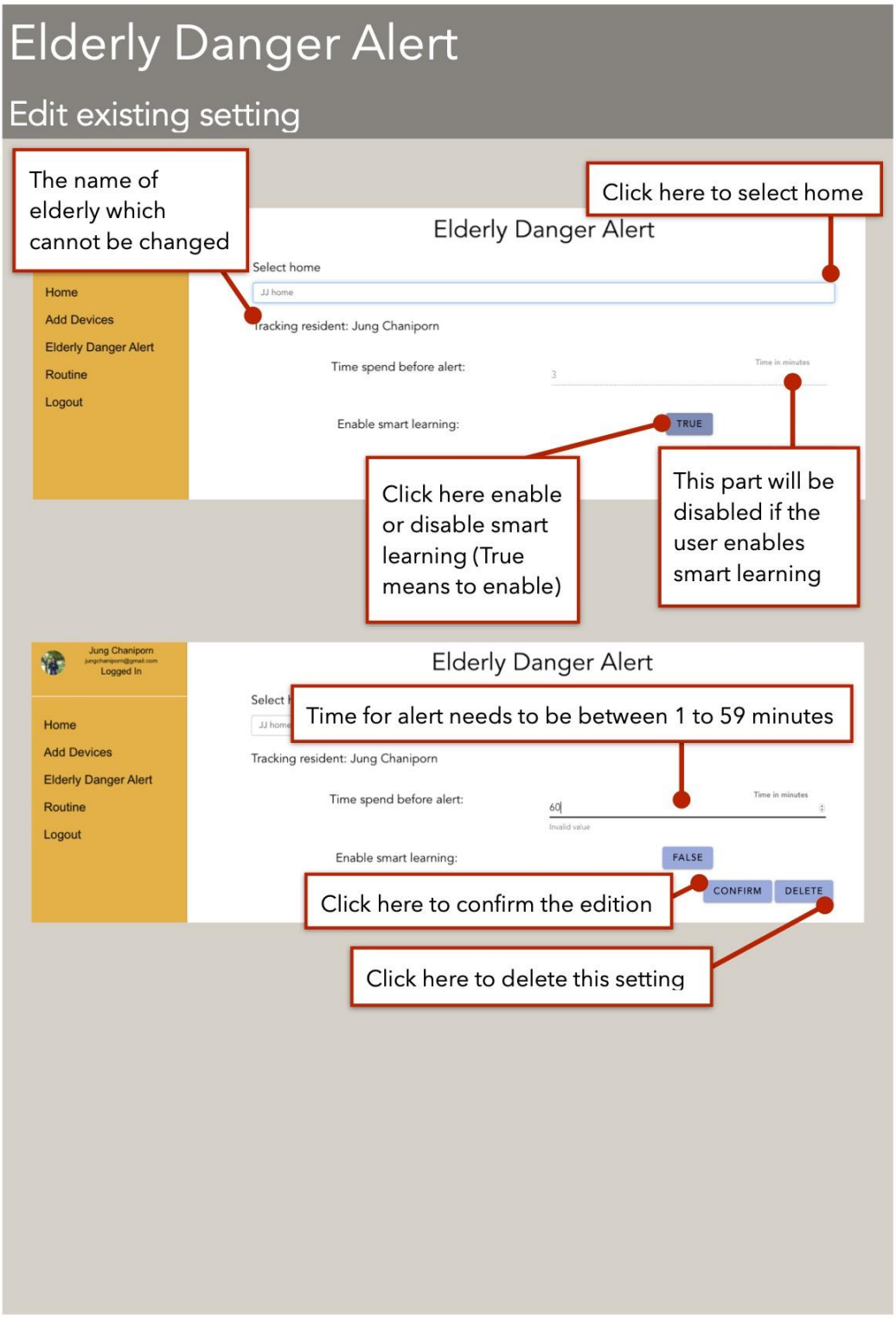

Figure B.14 Edit alert configuration page

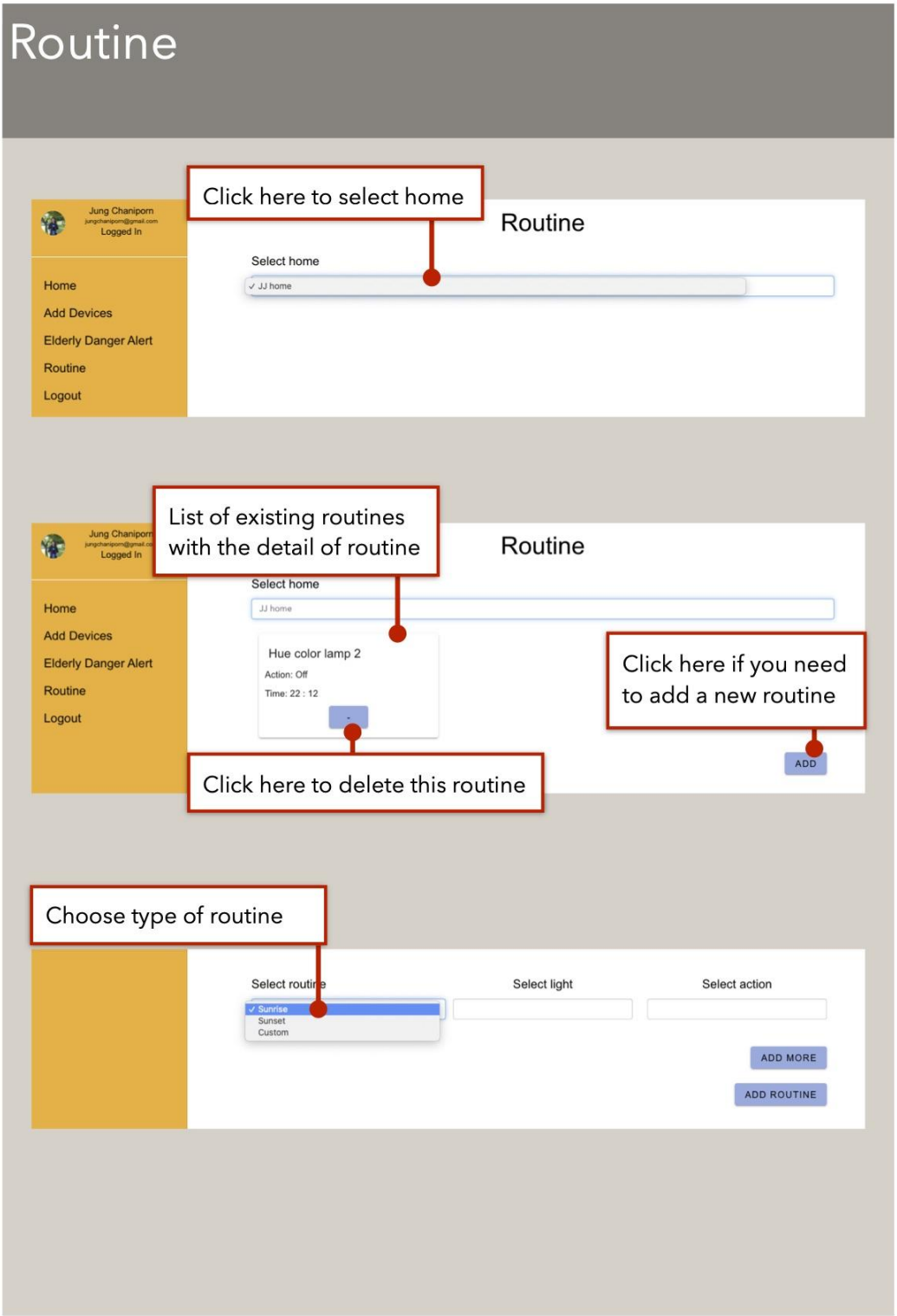

Figure B.15 Routine page

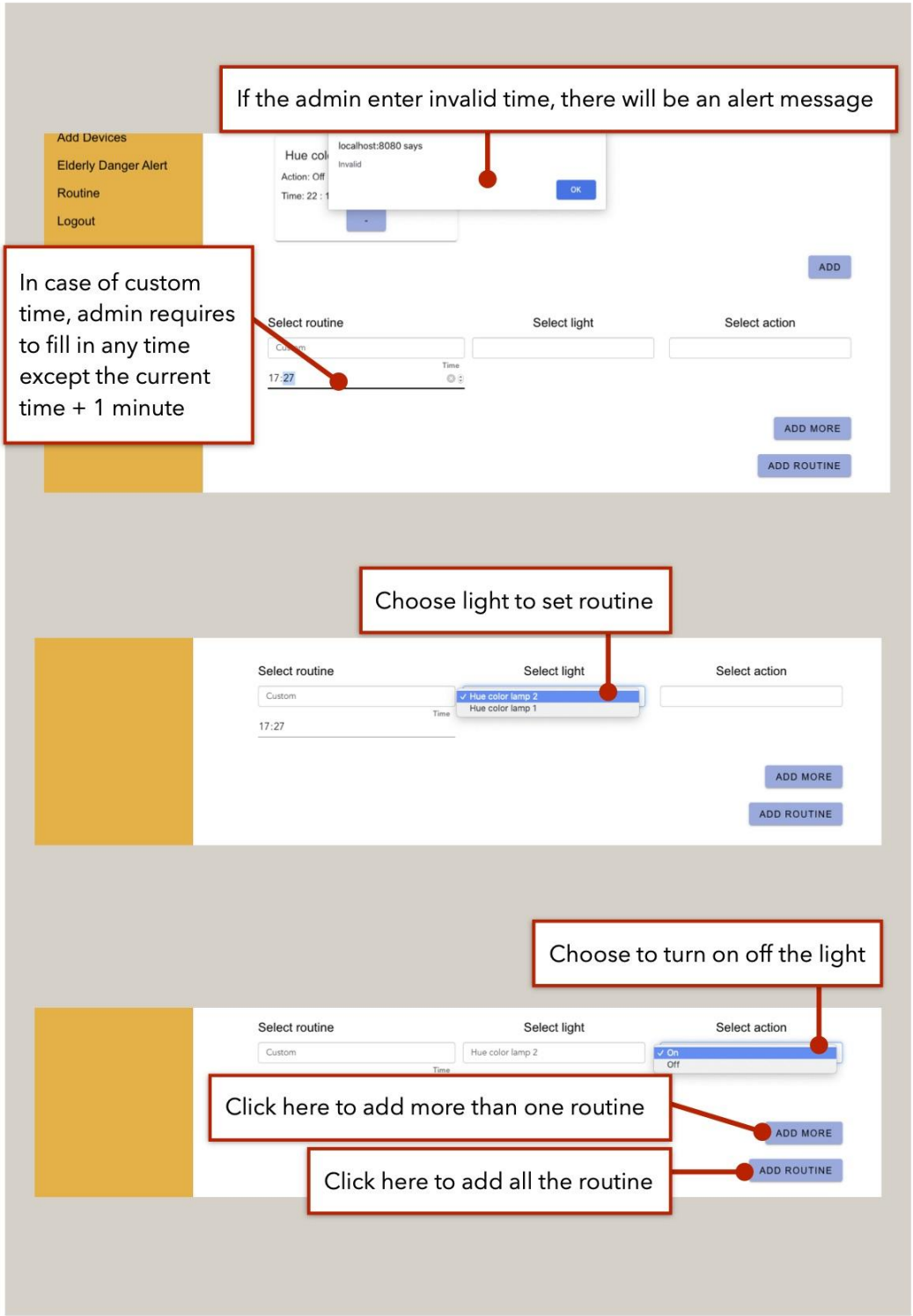

Figure B.16 Edit routine page

B.2 Mobile Application

Figure B.17 Login mobile application page

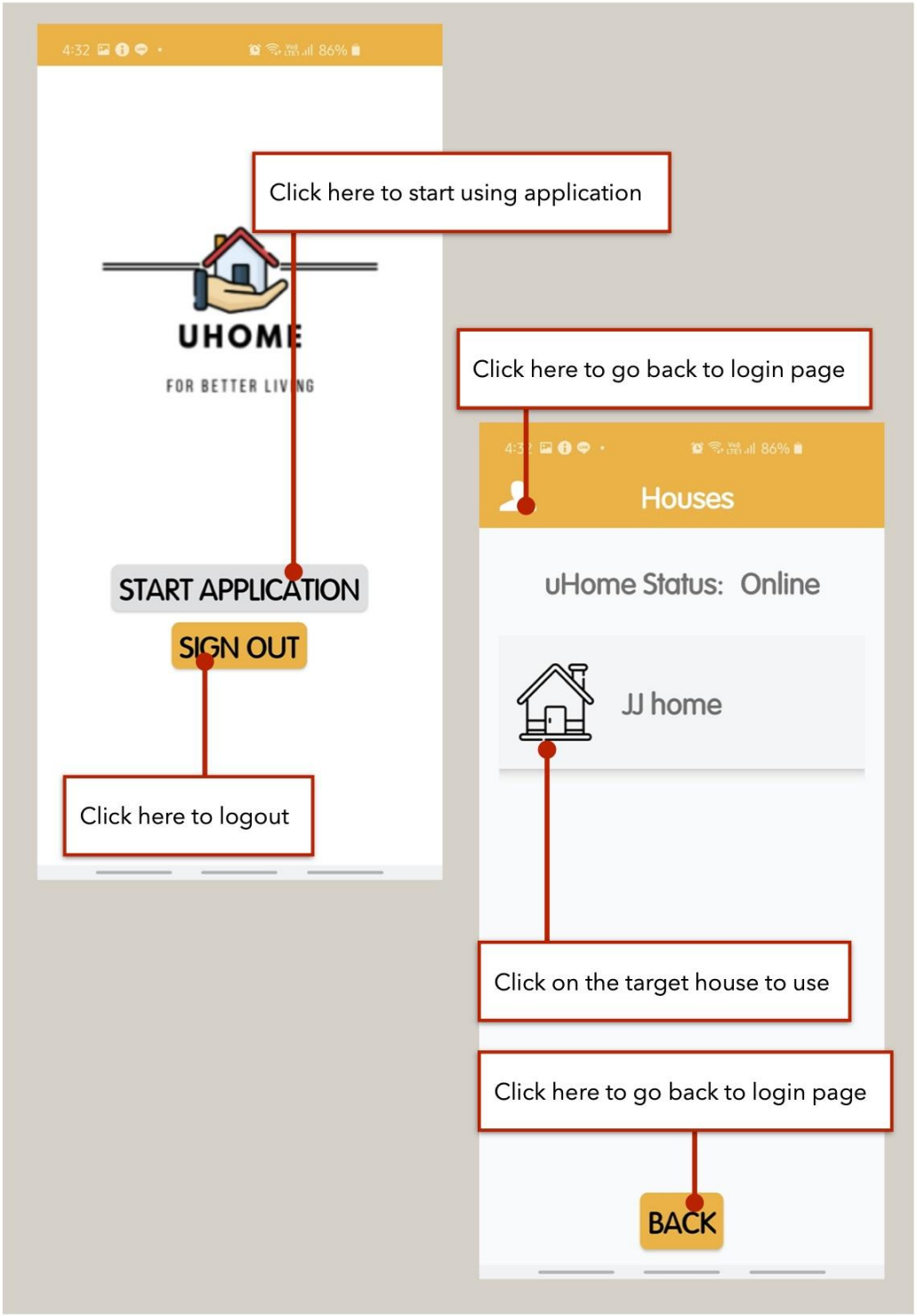

Figure B.18 Home page

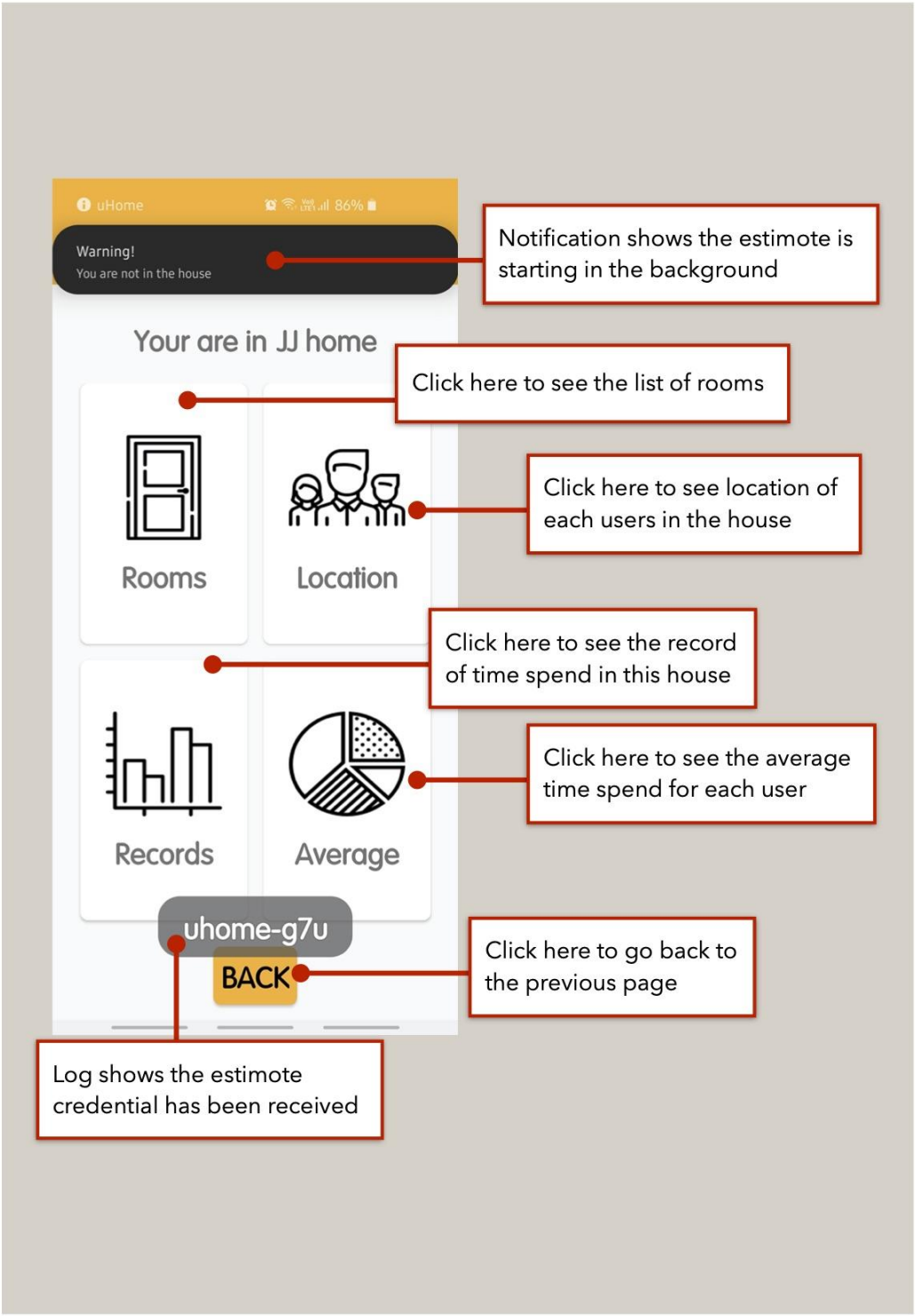

Figure B.19 Mobile application dashboard

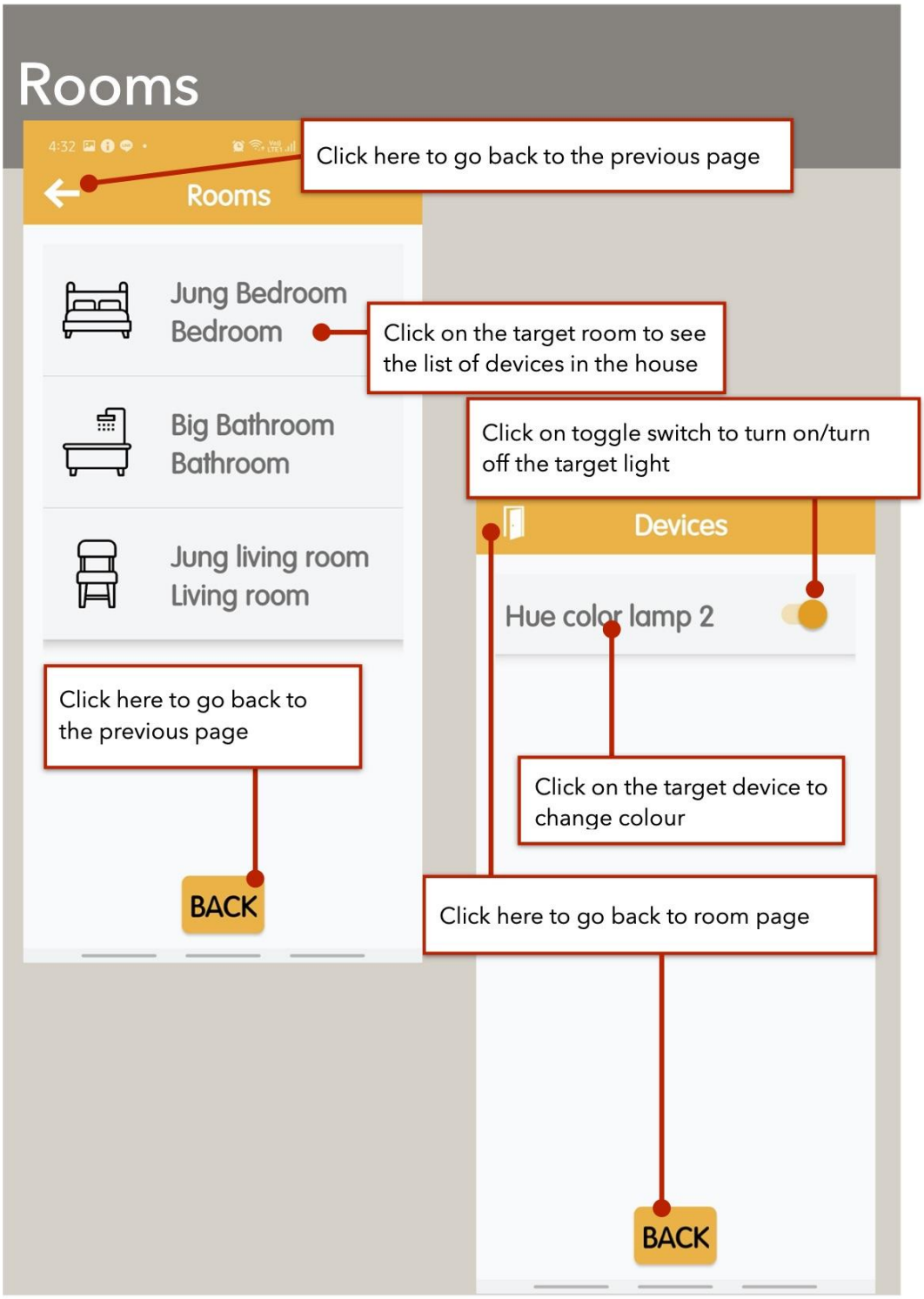

Figure B.20 Room page and Device page

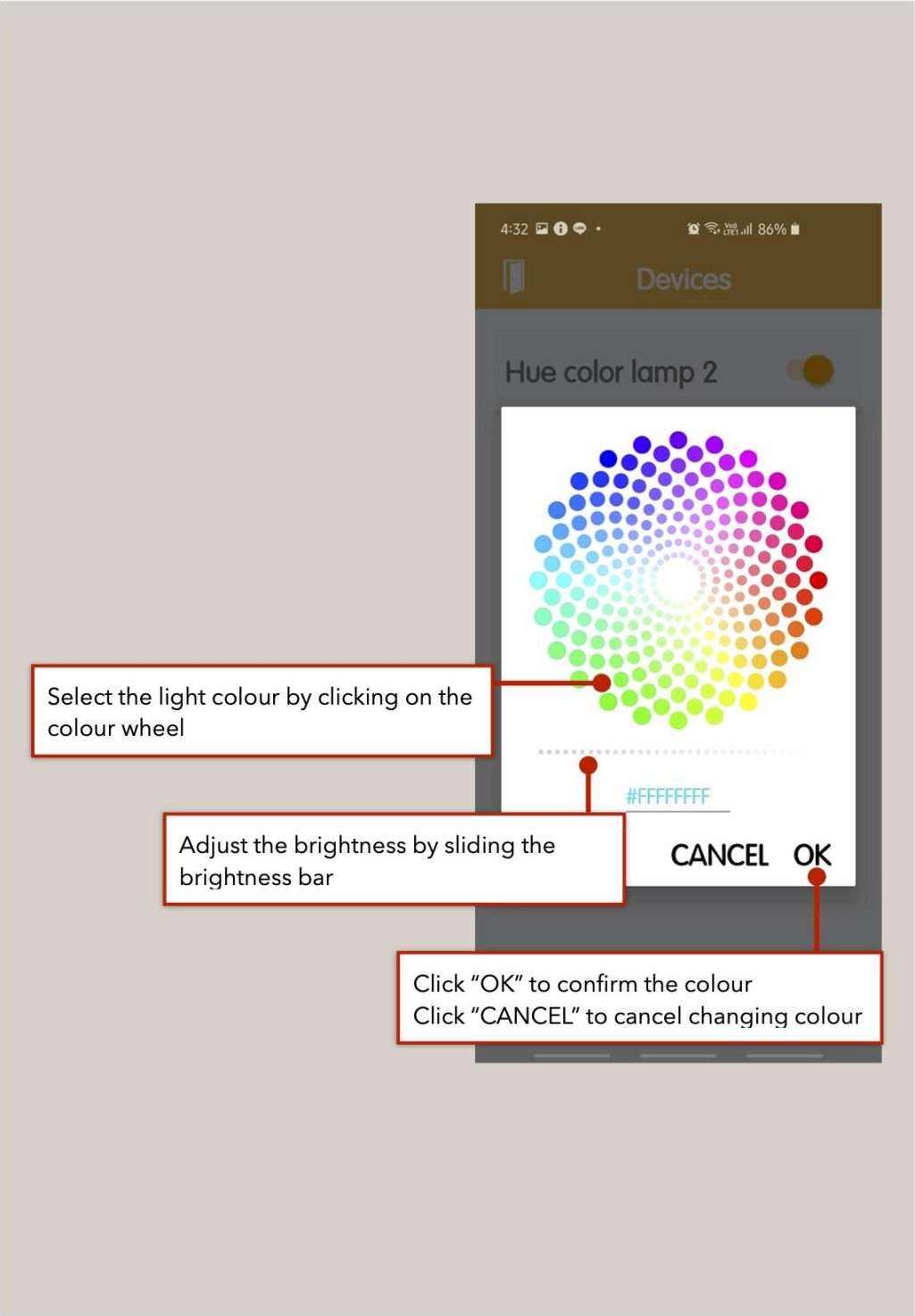

Figure B.21 Colour changing page

Figure B.22 FindMyFam page

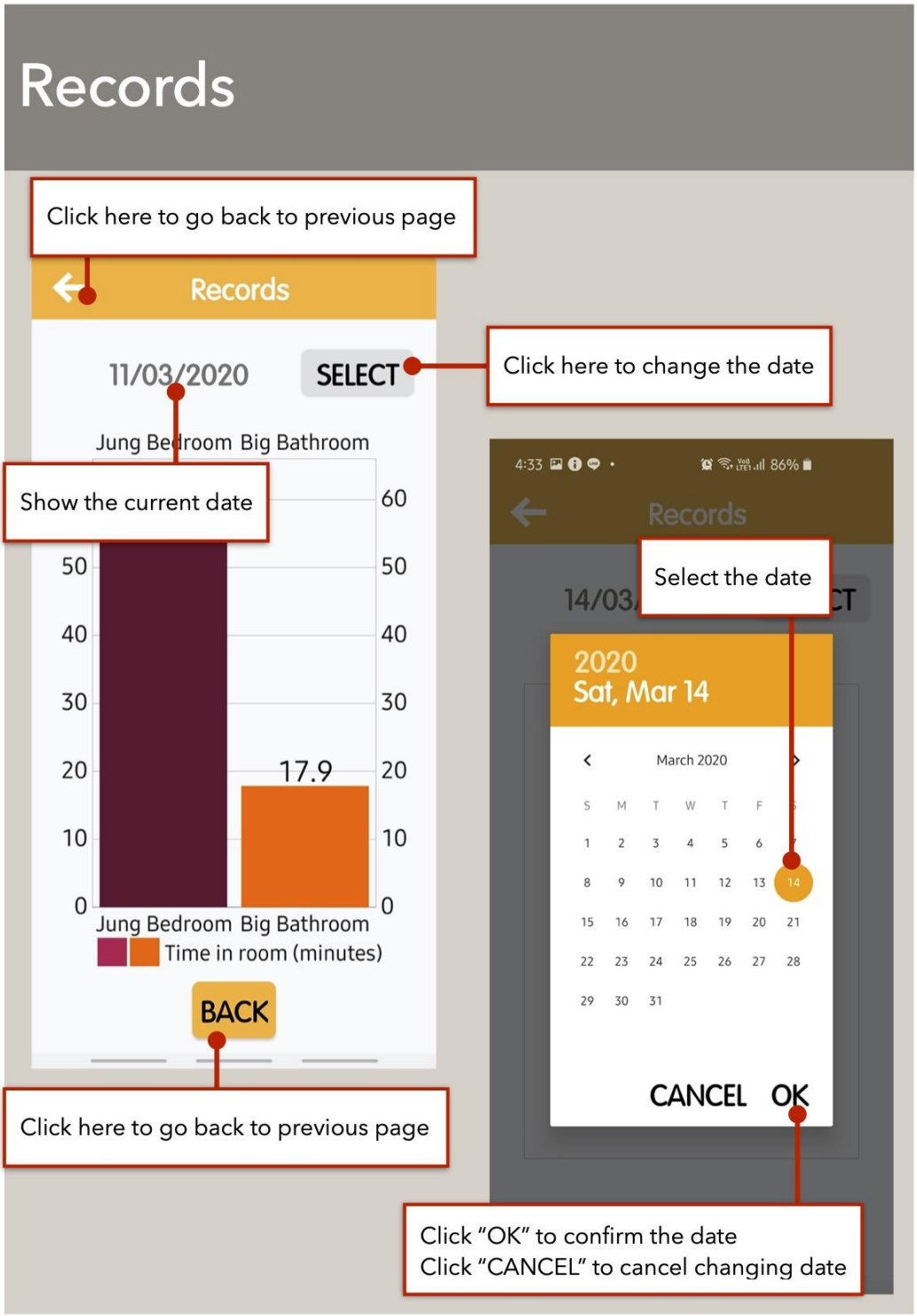

Figure B.23 Record of user page

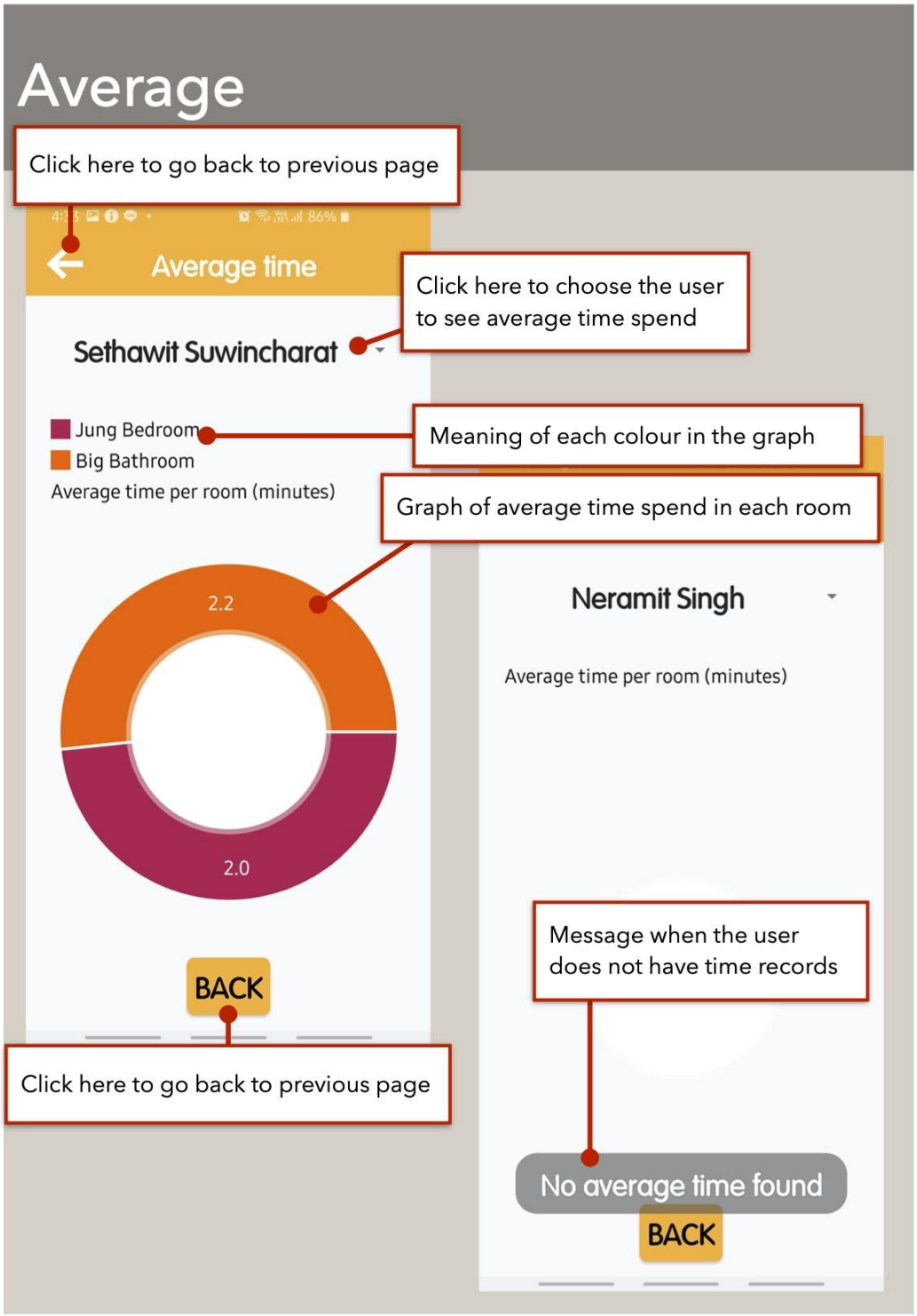

Figure B.24 Average of member page