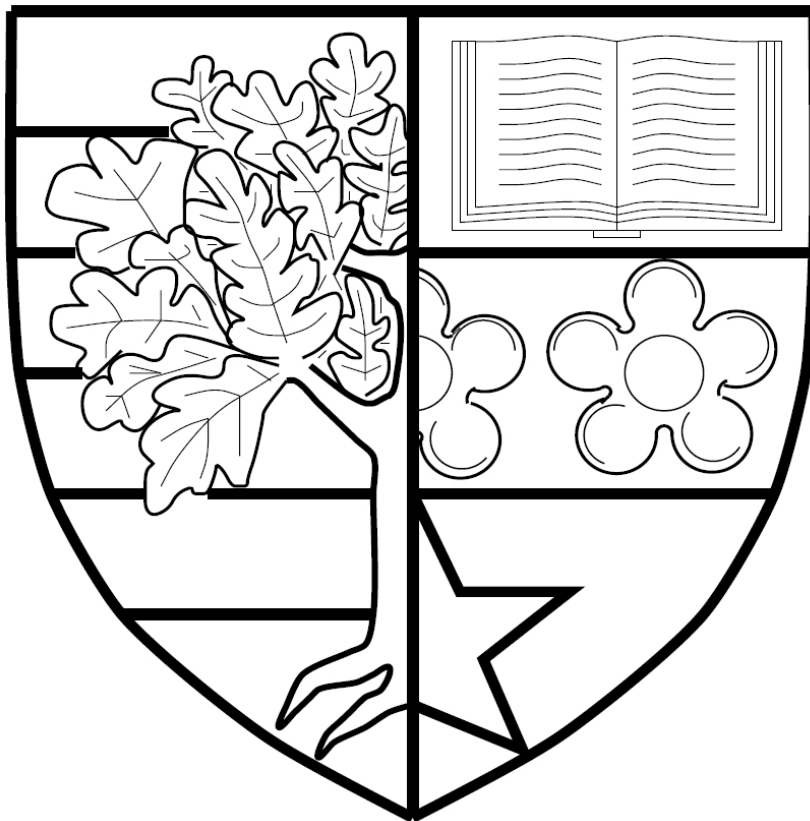


# **Akal – Alzheimer's Calendar**

## **Final Year Dissertation**



AUTHOR: CALLUM M HAYDEN

SUPERVISOR: DR. STEFANO PADILLA

SECOND READER: PROF. DAVID CORNE

DEGREE PROGRAMME: BSC (HONS) COMPUTER SYSTEMS

## Declaration

I, Callum Hayden, confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: Callum Hayden

Date: 23/04/2020

## Abstract

The purpose of this project is to allow the caregivers of dementia and Alzheimer's patients to have a new way of interacting with a patient and all caregivers of a patient. This will be done by creating a system using hardware and software in a cost-efficient method to ensure it is accessible by anyone who believes it could be a benefit. In order to do this, three key components must be created. An API which will allow patients and caregivers to interact, a patient system which will display information to a patient while sending details about their environment to the caregivers, and a system which allows caregivers to interact with a patient's schedule and set limits for when they would like to be alerted about the patient's environment. Research will be carried out to determine the best way to help the caregivers while not adding unnecessary stress to the patient.

This dissertation will show the requirements the final application will met, an explanation of the requirements, an overview of the research carried out in choosing the requirements, an explanation of the evaluation carried out, an explanation of the features of the application, an overview of the testing carried out on the application, and a conclusion expressing if the aims of this dissertation were met.

## Acknowledgements

I would like to thank my supervisor, Dr Stefano Padilla, for the opportunity to work on this idea and for his guidance with not only this project but also with every other issue and worry I had throughout the academic year. I would also like to thank my little brother, “Akil” for calling me “Akal” and giving me a name for this project. A thankyou is also due to my girlfriend, who put up with my long nights of typing and the bright glare from my computer monitors without a single complaint.

A final acknowledgement must be made to all caregivers, those who helped with this dissertation and those who could not. All of your hard work is appreciated. If you work as a caregiver, or even if you are simply caring for a neighbour, a friend, or a family member, all of your hard work is appreciated.

# Contents

Declaration.....	1
Abstract.....	2
Acknowledgements .....	3
Table of Figures.....	7
Table of Tables .....	9
Chapter 1: Introduction.....	10
1.1 Aim .....	10
1.2 Objectives .....	11
Chapter 2: Background reading.....	12
2.1 Alzheimer’s & Dementia.....	12
2.2 Hardware & Software .....	15
2.3 Current Technology .....	17
2.3.1 AARP Caregiving.....	18
2.3.2 Alzheimer’s Caregiver Buddy .....	19
2.3.3 North Lanarkshire Council Caregiving Solution.....	19
2.4 Evaluation Methods.....	21
Chapter 3: Requirements Analysis .....	23
3.1 Functional Requirements.....	23
3.1.1 Must have: .....	23
3.1.2 Should have: .....	24
3.1.3 Could have:.....	25
3.1.4 Won’t have: .....	26
3.2 Non-Functional Requirements.....	26
3.2.1 Must have: .....	26
3.2.2 Should have: .....	27
3.2.3 Could have:.....	28
3.2.4 Won’t have: .....	28
3.3 Requirements Revisited.....	28
Chapter 4: Development.....	29
4.1 Tools & Technologies .....	29
4.2 DevOps .....	31
4.2.1 Server & Domain Setup.....	31
4.2.2 CI/CD.....	32
4.3 Web Application.....	33
4.3.1 User Authentication.....	34
4.3.2 Calendar Feature.....	36

4.3.3 Weather Feature.....	38
4.3.4 Date & Time Feature .....	40
4.3.5 Patient Details Feature.....	40
4.3.6 Alert Feature .....	41
4.3.7 Updating Patient's Temperature Feature .....	42
4.3.8 Image Upload Feature .....	43
4.3.9 Patient's Temperature API Feature .....	43
4.3.10 Multiple Layouts Feature.....	44
4.4 Patient System .....	44
4.4.1 Hardware .....	44
4.4.2 Software.....	45
Chapter 5: Testing .....	47
5.1 Unit Tests.....	47
5.2 Functional Testing .....	49
5.3 Security Testing.....	50
5.3.1 SQL Injection .....	50
5.3.2 Cross Site Scripting Attacks .....	51
Chapter 6: Usability Study .....	52
6.1 Demographic .....	52
6.2 System Usability Study Questions .....	53
6.3 Conclusion.....	56
Chapter 7: Evaluation .....	58
7.1 Objectives Evaluation.....	58
7.2 Requirements Evaluation.....	58
7.2.1 Functional Requirement Evaluation.....	58
7.2.2 Non-Functional Requirement Evaluation.....	60
7.3 Testing Evaluation.....	61
7.3.1 Evaluation of Unit Tests .....	61
7.3.2 Evaluation of Functional Tests .....	62
7.3.3 Evaluation of Security Tests.....	62
7.4 Project Management Evaluation.....	63
Chapter 8: Conclusion .....	65
8.1 Achievements .....	65
8.2 Limitations.....	66
8.3 Future Work.....	67
8.4 Topics Learnt.....	68
8.4.1 Technical .....	68

8.4.2 Domain Knowledge .....	69
Bibliography .....	70
Appendix .....	74
1: Images of System .....	74
2: Code Snippets .....	82
2.1 User Model .....	82
2.2 Login Controller .....	83
2.3 Calendar Vue Component .....	83
2.4 Calendar Controller .....	86
2.5 Notifications Controller .....	88
3: User Evaluation .....	91
3.1 Original User Evaluation Plan .....	91
3.2 User Evaluation Handouts .....	91
3.2.1 Consent Form .....	92
3.2.2 Questionnaire .....	92
4: Considerations .....	95
4.1 Professional Considerations .....	95
4.2 Legal Considerations .....	95
4.3 Ethical Considerations .....	96
4.4 Social Consideration .....	97
5: Project Management .....	98
5.1 Risk Analysis .....	98
5.1.1 Risk table .....	99
5.1.2 Risk matrix .....	99
5.1.3 Action plan .....	100
5.2 Plan .....	101
5.2.1 Sprint plan .....	102
5.2.2 Gantt chart .....	103

## Table of Figures

<b>Figure 1:</b> Image of the DAKboard system. ....	16
<b>Figure 2:</b> Screenshots of the AARP Caregiving application. ....	18
<b>Figure 3:</b> Screenshots of the Alzheimer's Caregiver Buddy application ....	19
<b>Figure 4:</b> Diagram of the system architecture, MVC .....	33
<b>Figure 5:</b> Screenshot of login form.....	35
<b>Figure 6:</b> Screenshot of registering as a patient.....	35
<b>Figure 7:</b> Screenshot of registering as a carer .....	36
<b>Figure 8:</b> Screenshot of the calendar feature .....	38
<b>Figure 9:</b> Screenshot of the weather .....	38
<b>Figure 10:</b> Screenshot of date and time feature .....	40
<b>Figure 11:</b> Screenshot of the HTML email sent when patient's home is too warm.....	41
<b>Figure 12:</b> Screenshot of form to update patient's maximum and minimum temperatures.....	42
<b>Figure 13:</b> Screenshot of the image upload feature .....	43
<b>Figure 14:</b> Screenshot of the form used to change a patient's layout .....	44
<b>Figure 15:</b> Image of the Raspberry Pi wired to the sensor .....	45
<b>Figure 16:</b> Diagram of wiring for the Raspberry Pi and sensor.....	45
<b>Figure 17:</b> Chart showing responses to Q1.....	53
<b>Figure 18:</b> Chart showing responses to Q2.....	53
<b>Figure 19:</b> Chart showing responses to Q3.....	54
<b>Figure 20:</b> Chart showing responses to Q4.....	54
<b>Figure 21:</b> Chart showing responses to Q5.....	54
<b>Figure 22:</b> Chart showing responses to Q6.....	55
<b>Figure 23:</b> Chart showing responses to Q7.....	55
<b>Figure 24:</b> Chart showing responses to Q8.....	55
<b>Figure 25:</b> Chart showing responses to Q9.....	56
<b>Figure 26:</b> Chart showing responses to Q10.....	56



<b>Figure 27:</b> Chart showing an average score from the 24 participants for each question.....	57
<b>Figure 28:</b> Output of running PHPUnit tests .....	61
<b>Figure 29:</b> Screenshot of home page as carer user .....	74
<b>Figure 30:</b> Screenshot of home page as patient user .....	74
<b>Figure 31:</b> Screenshot of patient details feature .....	75
<b>Figure 32:</b> Screenshot of updating or deleting an event .....	75
<b>Figure 33:</b> Screenshot of HTML email for when an event is updated .....	76
<b>Figure 34:</b> Screenshot of HTML welcome email sent when a user registers .....	76
<b>Figure 35:</b> Screenshot of HTML email sent to alert user of door being opened .....	77
<b>Figure 36:</b> Screenshot of HTML email sent if temperature is too cold .....	77
<b>Figure 37:</b> Screenshot of SMS sent if temperature is too cold .....	78
<b>Figure 38:</b> Screenshot of SMS sent if temperature is too hot.....	78
<b>Figure 39:</b> Screenshot of SMS sent if door has been opened .....	79
<b>Figure 40:</b> Screenshot of an SQL injection attempt with ' in email.....	79
<b>Figure 41:</b> Screenshot of an SQL injection attempt with " in email.....	80
<b>Figure 42:</b> Screenshot of an XSS attack attempt with JS in email field.....	80
<b>Figure 43:</b> Screenshot of an XSS Attack attempt with JS in the password field.....	81

## Table of Tables

<b>Table 1:</b> Details of the tools and technologies used in this dissertation. ....	29
<b>Table 2:</b> Details of the DNS records for the akal.app domain.....	31
<b>Table 3:</b> List of unit tests carried out along with their assertions.....	47
<b>Table 4:</b> List of functional tests carried out along with the results.....	49
<b>Table 5:</b> List of the objectives of this dissertation with their achieved status .....	58
<b>Table 6:</b> List of the functional requirements and if they have been achieved .....	59
<b>Table 7:</b> List of non-functional requirements and their achieved status.....	60
<b>Table 8:</b> Table showing risks and informaiton about these risks.....	99
<b>Table 9:</b> Risk Matrix .....	100
<b>Table 10:</b> Sprint plan numbering sprints and stating tasks to be performed in each sprint .....	102

## Chapter 1: Introduction

Dementia is a term for diseases and conditions characterized by a decline in memory, language, problem-solving and other thinking skills. Alzheimer's is the most common cause of dementia (Alzheimer's Association, 2019). With the number of dementia patients growing rapidly, the dementia statistics suggests there will be a 204% rise in dementia patients worldwide from the fifty million in 2018 to one hundred and fifty-two million in 2050 (World Health Organisation, 2019). With this many dementia patients, we can assume there are at least half the amount of dementia caregivers, be they nurses, family members, or friends and neighbours who try to help out.

Currently there is no well-thought-out solution to help these caregivers. With numerous attempts being made with solutions such as using wall mounted calendars to check when a caregiver will be visiting, or solutions which only benefit council nurses in Scotland such as alerts for falls. While these solutions work as a concept, they do not allow for user engagement from all types of users. For a family-member caregiver to find the best time to visit the patient, with the wall mounted calendar approach they have the option of phoning the patient to ask them to check the calendar, or the option of going and checking, if the patient is at home. With the solutions such as the alerts for falls, a message is sent to the council in Scotland, who then send a council caregiver or nurse to visit the patient. The family have no engagement with this solution, when they could check on the patient to save council resources.

### 1.1 Aim

This section will outline the key aims of this dissertation, and what the system should be. This will be used to evaluate if the final dissertation matches the initial expectations placed upon it.

The aim of this project is to develop a cost-effective, hardware and software, tool that can be used to get information about a dementia or Alzheimer's patient's schedule and environment remotely, while attempting to help the patient through reorientation with prompts of the current date and time. This will be achieved by creating an interactive calendar which can be checked and updated remotely, along with a hardware solution to gather data about the patient's environment.

## 1.2 Objectives

This section will discuss the objectives of the project. These objectives will be used to determine if this dissertation has been successful in the achievements section of the conclusion.

The objectives for this project are:

- **Gather requirements for the system.** This will be achieved through research, background reading, and interviews. The interviews will be carried out with caregivers who work for the North Lanarkshire council, along with caregivers met at the “Memory Café” Alzheimer’s group and colleagues from the Ability Net charity.
- **Design a system that is easy to use for multiple user types.** With the use of software such as Adobe XD, prototypes will be made of the user interface of the system prior to beginning the implementation. These UIs will be different depending on the user type. The patient UI will only display information, whereas the caregiver’s UI will display information as well as allowing for adding new information. Designs for different screen sizes and systems will also be created, including a design for mobile, tablet and computers.
- **Develop the system to match the requirements.** The requirements will help solve issues highlighted in the literature review such as patients being too cold by alerting caregivers about the temperature in the patients home, aiding in creating a daily routine for the patient and caregivers by means of a calendar which can be viewed and updated remotely, and repetitive questioning behaviour by displaying information on the system such as the date, time and temperature.
- **Ensure the system is well tested.** This will ensure the end product works as expected and will be done with the use of unit tests, such as Jest for the front-end JavaScript and Phpunit for the backend php.
- **Evaluate the system’s user interface with help from caregivers to ensure it is easy to use.** Using the system usability scale the system will be evaluated by caregivers to find how usable they believe the system is. This will be the overall measure of success in this dissertation.

## Chapter 2: Background reading

This chapter contains information about the research carried out for this dissertation. The research can be split into parts, research about Alzheimer's disease and dementia, research about the hardware and software which will be used to implement this system, and research into how best evaluate this dissertation.

Research has been carried out by both reading articles and by means of interviews and focus groups. Information gathered from literature has been referenced and can be found in the bibliography at the end of this dissertation.

### 2.1 Alzheimer's & Dementia

This section will discuss the research carried out into Alzheimer's disease and dementia. It will include what they are, symptoms, and how to ease the impact of these symptoms on caregivers.

According to the Oxford English dictionary, dementia is "a serious mental disorder caused by brain disease or injury, that affects the ability to think, remember and behave normally" (Murray, James; Minor, William Chester, 2019, p. Dementia). Dementia can be caused by Alzheimer's disease which the Oxford English dictionary defines as "a serious disease, especially affecting older people, that prevents the brain from functioning normally and causes loss of memory, loss of ability to speak clearly, etc" (Murray, James; Minor, William Chester, 2019, p. Alzheimer's Disease) . Other causes of dementia include diseases such as Parkinson's and Huntington's, vascular diseases, strokes, depression, infections such as HIV, and chronic drug use. There are many symptoms to Alzheimer's and dementia, two which are relevant to this dissertation as they can affect the patient's caregiver. These symptoms are repetitive questioning behaviour and patients feeling too cold.

Repetitive questioning behaviour is the tendency dementia patients have to repeatedly ask the same question. This is the most common complaint from caregivers in regard to dementia and Alzheimer's patients. The repetitive questioning can be due to short term memory loss and lack of recall, boredom, or anxiety about the future. The common questions asked are about the time, schedule, current event details, information, and opinions or feedback. Currently, the solution used by caregivers is to provide

reorientation with written reminders, calendars, whiteboards, signs, clocks, and familiar objects such as photos or music (Hawkey, Inkpen, Rockwood, Mcallister, & Slonim, 2005). Interviewees also stated that the use of photographs of family members could help draw the patient's attention towards the system. This could be improved by allowing captions on the photographs such as the person's name and their relationship to the patient, as this would help the patient recognize the people in the photographs.

Repetitive questioning behaviour can be a strain on a caregiver's patience and can lead to angry outbursts and arguments between the caregiver and the patient. To help remedy this, this dissertation proposes displaying information in an easy to understand manner. This will allow caregivers to refer the patient to the system to answer some of their questions, which overtime, will lead to the patient not needing to ask the question as often. This dissertation also proposes to aid in providing reorientation with reminders, a calendar, and photos on the system. The use of captioned photographs could help draw the patient's attention to the system while aiding with reorientation, and the use of a calendar could be helpful with scheduling daily plans for the caregivers and the patients. Further literature which agrees with the idea of a calendar comes from the Alzheimer's association. They state that, daily routines can help Alzheimer's and Dementia patients, as well as helping the caregiver. An example plan could include basic everyday tasks such as eating and getting prepared for bed (Alzheimer's Association, 2019). As these are tasks which happen daily, with a calendar application it would be possible for the caregivers to set certain tasks to repeat daily. This would make writing daily plans a lot easier as caregivers would not have to worry about accidentally forgetting to add a simple task such as washing their plates. This system also allows caregivers who do not live close enough to give the patient a handwritten daily plan, to still be involved in creating their daily plan from afar through the system.

Another issue is patients feeling too cold. According to an article by Tena L. Scallan (Scallan, 2017) elderly people feel cold more due to the decrease in their metabolic rate, low blood pressure, and loss of elasticity of the blood vessels, amongst others. This can lead to conditions such as hypothermia should the patient not be able to raise their body temperature. Patients with dementia or Alzheimer's may forget how to turn their central heating on, but on the other hand, some can forget to turn it off

which can lead to their homes being too warm with risks such as dehydration, hyperthermia, malaise, and renal failure amongst others (van Loenhout, et al., 2016). This is corroborated with interviews with caregivers who explain normally, should the family be able to afford it, the heating is almost always on, and creating an environment which the interviewees said was uncomfortably warm. This was a topic brought up during the interviews, and the interviewees reiterated the fact that patients' homes tend to be either too cold or too warm. Most of the interviewees explained that in many homes, a patient either forgets to turn their heating on, or is too worried about the cost of heating to turn it on. They also explained in some cases, a house will be so warm that the patient will be sitting in a t-shirt and shorts, sweating because of the heat, and simply doesn't remember the heating is on. Interviewee's provided a solution to this, which was to gather the temperature from a patient's home and share it with the caregivers. If the caregiver deems it to be too warm or too cold, they could change the temperature remotely through the system. This is unachievable in this dissertation, however, a new idea stemmed from this. The idea is that the system could gather the temperature inside the user's house and send alerts to the patients' caregivers should the temperature be too high or too low. This would allow the caregivers to call the patient and explain how to turn the heating up or down or go to the patient's home and solve it.

While researching repetitive questioning behaviour, an interesting study about using to do lists to help patients with Alzheimer's and dementia (Alzheimer's Society, 2010) was considered. During the interviews with caregivers, some interviewees stated that while lists are good for early stage dementia patients, in practice, they are more trouble than what it is worth. One scenario proposed was to display a list of how to make a simple meal. The interviewee stated issues with this scenario such as causing unnecessary stress on the patient should they not have the ingredients, the patient potentially forgetting what they are doing while cooking leading to a fire, and the stress of trying to remember instructions while moving from the screen to the kitchen. Due to this feedback, to do lists will not be a part of this system.

Other ideas from the interviews while excellent ideas, are un-achievable in this project, such as creating a system which allows for video calls between caregivers and patients or creating a natural language

processing system which would answer questions for the patient much like Alexa or Siri. Given more time and a different approach to this project however, this would be achievable. However, one idea in particular stood out as something important to keep in mind. This is the use of easy to read, large, and bold fonts. Interviewee's recommended this due to, in general, elderly people having worse vision which could cause them to struggle to read the information on the system causing distress.

## 2.2 Hardware & Software

This section will discuss research carried out in regards with hardware and software decisions, along with research towards similar products on the market. The first main decision was which system would be used as a core for this system. The options include Phidgets, Udoo, Raspberry Pi and Arduino. However, the Phidgets and the Udoo do not suit this project due to their cost. With one of the goals of this project being to develop an affordable solution, making the Raspberry Pi or the Arduino the best options. When comparing the Raspberry Pi and the Arduino, the Raspberry Pi has a lower power consumption (5V compared to Arduino's 7-12V) along with more RAM (256-512MB compared to Arduino's 16-32KB), a built in way to connect to the internet (using both Ethernet and WiFi), and the option to connect multiple peripherals through USB (2 USB ports compared to Arduino's 1) made it a better option (Maksimovic, Vujovic, Davidović, Milosevic, & Perisic, 2014).

After the hardware was decided, the software had to be chosen. The system was mostly created with the Laravel framework. "Laravel is an MVC web-development framework written in PHP. It has been designed to improve the quality of your software. Laravel was designed with the philosophy of using convention over configuration. This means that it makes intelligent assumptions about what you are trying to accomplish so that in most situations you will be able to accomplish your goals with much less code" (McCool, 2012)

MVC stands for model-view-controller. An MVC pattern divides the program's logic into three main elements. The model is the central component of the pattern. It is the data structure. The view is the representation of information, and the controller is the logic which accepts an input and returns a command for the model, or a view. The benefits of this includes:



- Views and controllers can be substituted to provide alternate user interfaces for the same model.
- Multiple simultaneous views of the same model are provided.
- Easy to change the user interface.
- Easy to test the core of the application, due to it being encapsulated by the model.

These benefits apply to this dissertation as there will be different views returned to the user based on which model (user type) is detected by the controller (logged in) (Kılıçdağı & Yilmaz, 2014).

Once the tools for this dissertation were researched, the next goal was to check if there was a system similar to this dissertation currently available. While no system that does the same was found, a system which involved an electronic calendar displayed constantly on a screen was found. This system is called DAKboard. From DAKboard, much can be learnt, but the main difference between DAKboard and this



**Figure 1:** Image of the DAKboard system.

project is DAKboard is aimed towards families or individuals to share in their house. It links with a third-party calendar system of the user's choice and supports Google Calendar, iCloud Calendar, and Facebook amongst others. It includes tools to create a digital photo frame and an office welcome board apart from only a calendar. It costs £285.43 for the hardware which is a 24" 1080p Wi-Fi enabled LED screen. To use the system, it costs £6.19 per month for the use of software for up to three screens. This would allow for the DAKboard in the patient's home, plus two caregivers (dakboard, 2019).

While DAKboard has many benefits and many great ideas, some of the flaws lie in how much it does. The ability to fit many markets works for

DAKboard, but with dementia patients who can be forgetful and may become frustrated if it doesn't work as expected due to touching a wrong button, it can be very distressful. Training would have to be provided to caregivers and family members also, with the issue of the possibility of updating the wrong calendar, or accidentally changing the calendar on the DAKboard to your work calendar.

Overall reviews for DAKboard as a product are very positive. While there are some bad reviews, they tend to be based on the shipping or customer service rather than for the software. The hardware also has some bad reviews based on the hardware being built upon a third-party system. This however would be the same case as this dissertation. Reviews can be found on the DAKboard Facebook page and on Amazon (Reviews, Amazon reviews of DAKboard wall display v2, 2017) (Reviews, dakboard reviews, 2017)

The ideas taken from DAKboard which will be explored more are the options to add photos to the calendar and modify the calendars layout. The use of photos will not follow DAKboard implementation of taking the photos from social media sites such as Instagram or Flickr as the target audience of this project may not use these platforms.

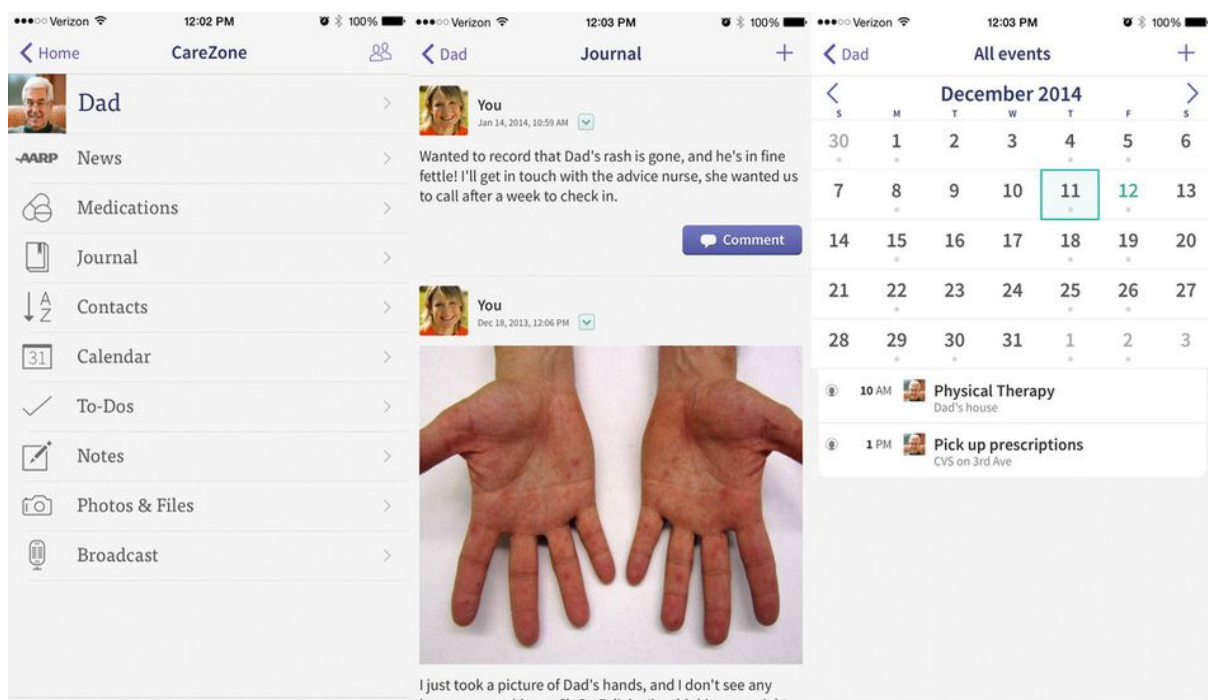
## 2.3 Current Technology

Currently the main systems that are available to help caregivers with patients are AARP caregiving and Alzheimer's Caregiver Buddy. Both of these are free mobile phone applications. Another solution to keep in mind is the solution currently used by the North Lanarkshire council for their caregivers and patients.

While the AARP caregiving and Alzheimer's Caregiver Buddy application are available for use by anyone, the solution used by the North Lanarkshire council is not. It is only available to council staff and is often not referred to as a caregiver application as it is seen as more of a work application as it deals with work rotas, confidential patient information, and the employees details such as holidays and wage slips.

### 2.3.1 AARP Caregiving

AARP Caregiving is a free to use mobile application created by the American Association of Retired Persons. It is an app which allows users to monitor symptoms, coordinate care with family members or caregivers, and keep track of appointments and medications. It also includes a help centre where caregivers can find answers to questions. This application's main focus is on empowering caregivers with information to help them care for an elderly person (Duere, 2016).

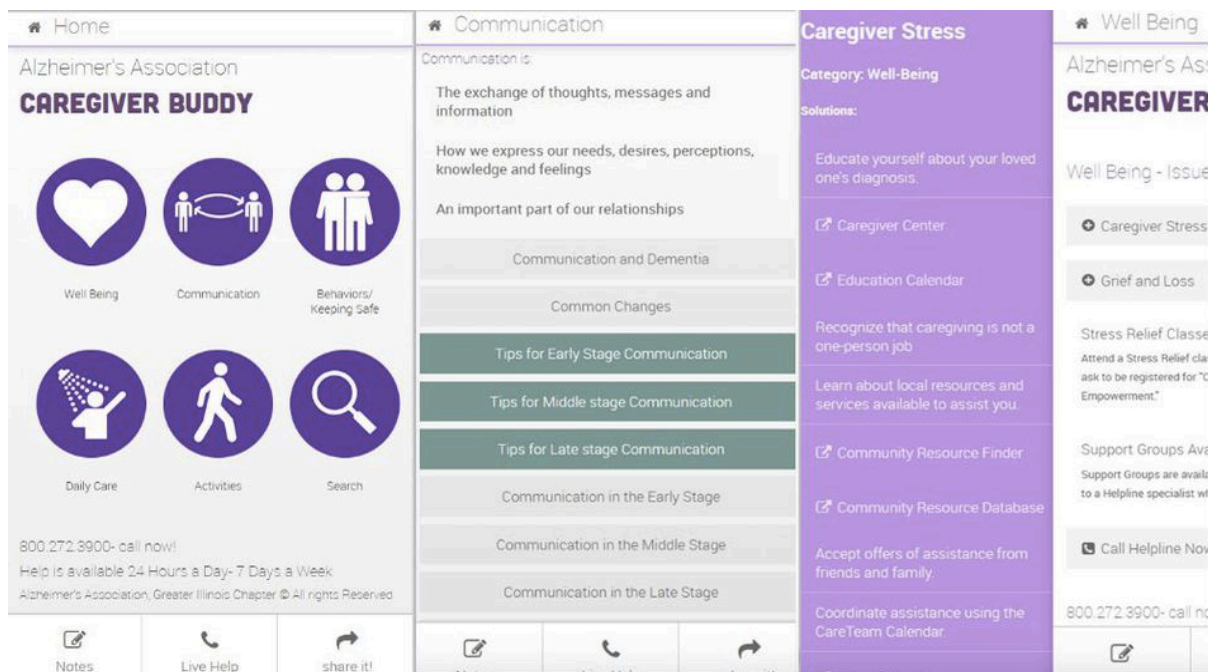


**Figure 2:** Screenshots of the AARP Caregiving application.

This application has a great set of features for allowing caregivers to interact with each other while keeping the focus on the patient. It not only stores vital information about the patient such as their medication, but it also allows for sharing updates, much like a version of Facebook but for sharing details about the patient. It also integrates a calendar feature, much like this dissertation, however, it seems to be aimed more so for the caregivers to arrange appointments for the patient, instead of allowing for the patients to view their plans. The other difference, is that this solution is an application and does not have any hardware available for the patient side, meaning the caregivers do not have access to information about the patient's environment unless they personally visit.

### 2.3.2 Alzheimer's Caregiver Buddy

Alzheimer's caregiver buddy is a free to use application created by the Alzheimer's Association which aims to help caregivers manage their personal stress, navigate family conflict, and reach a 24/7 helpline. Some of its features include tips for how to care for patients relating primarily to mealtimes and personal hygiene, and activities you can do with a patient to help stimulate their body and mind. It also gives caregivers tips on how to deal with some of the behavioural changes in Alzheimer's and dementia patients such as aggressive mood swings or hallucinations (Duere, 2016).



**Figure 3:** Screenshots of the Alzheimer's Caregiver Buddy application

This application shares great resources for caregivers who may not have any experience with caring for patients suffering from memory loss. When compared to this dissertation it is extremely different. Where this application focuses on giving caregivers resources to help them learn how to care for patients more efficiently, Akal focusses more on planning and monitoring a patient with a key focus towards being able to share the plans with other caregivers.

### 2.3.3 North Lanarkshire Council Caregiving Solution

The North Lanarkshire Council offers free care services to all those over the age of 65 as well as anyone with a physical disability, learning disability, ill health, dementia, emotional difficulties, mental health difficulties, a dependency on alcohol or other substances, general frailty, caregivers, and children and

families in crisis (Home Support Services | North Lanarkshire Council, 2017). To provide this service, the council hires care staff, outsources to private companies, and on occasions, must rely on bank nurses. In order for the services to be provided, the staff need information about the patients, such as the reason of the visit, the tasks to be done for the patient, and sensitive information such as the patient's address, and the code for the key safe outside of the patients home. Previously, this data was posted to carers prior to their work hours or collected at one of the central offices. This, however, can lead to the accidental loss of the document, allowing whoever finds it the option to break into a patient's home, with the use of their key, and the knowledge that the patient is unable to stop them in any way. This is where their care giving solution shines.

The North Lanarkshire council has changed the way they provide care through the use of technology. Instead of using paper-based work rotas with sensitive details, they use a combination of hardware and software. Each of the care giving staff are supplied with a mobile phone which has been modified to be unable to download applications, take photographs, or browse the internet. Each phone, however, has an application which allows for a secure login to North Lanarkshire servers. Once logged in, the application not only shows the caregivers work rota, it is also integrated with Google Maps to work as satellite navigation for caregivers to be able to be directed from house to house. The application provides the caregivers all the information the paper based rota provided, along with features such as the previously mentioned navigation, a way to discover what medication is and what it is for through matching the pills to pictures, ensuring patients are not accidentally given the wrong medication; and a way to communicate any issues that arise directly to the right person such as to the police in case of illegal drug use, managers in case of an issue with a patient, or their work colleagues to arrange meeting for jobs which require two people.

This solution, however, is solely focused on professional caregivers, and would be of little to no benefit to the wider public if made available for everyone. It also does not contain a way to interact with a patient, and the caregiver can only report issues, and mark jobs as completed.

This, along with the other examples provided, show great ways of how technology can benefit those in need. From illustrating the uses of technology for interacting between caregivers, to using technology to help a caregiver provide the best care possible.

As technology advances, more and more solutions will be created for the vulnerable who need care and instruction to not only survive, but to be able to have some comforts in their later years.

## 2.4 Evaluation Methods

This section will discuss the research carried out about evaluation methods, what they are, types of evaluation methods, and the chosen evaluation method. “Evaluation methods are the criteria for evaluating the success of a program or project” (Pearce, 2016). An evaluation method will be used to evaluate if this dissertation has been a success or what needs improvement. Evaluation techniques can be either qualitative or quantitative. Qualitative techniques are based on a list of features to be analysed for the system. A qualitative approach is attractive when the decision problem is simple (Dujmović, 1996). Essentially, qualitative techniques are good for understanding the underlying reasons and gathering opinions about a system. Quantitative techniques are used to quantify attitudes, opinions, or other defined variables. It uses measurable data to create facts and find patterns in research (DeFranzo, 2011).

To carry out the final evaluation of the system, the chosen evaluation method is the system usability scale. The system usability scale (SUS) was created by John Brooke and is defined by the creator as “a quick and dirty usability scale” (Jordan, Thomas, Weerdmeester, & McClelland, 1996). SUS is composed of 10 statements which are scored from zero to five based on if the participant agrees with the statement or not. The final score ranges from zero to one hundred where the higher the score the higher the level of usability. The key benefit of SUS is that it provides a single reference score per participants’ view of a product’s usability (Bangor, Kortum, & Miller, 2008).

An alternative considered was the use of first click testing. First click testing is a method for measuring the usability of a website, app, or design by finding out how easy it is to complete a given task (Usability Hub, 2020). This is done by providing participants a list of tasks to complete and testing if their first

click on the application is the correct way to complete this task. The significance of this is due to the fact that if users get the first click right, they have twice as much of a chance of completing their task than if they get it wrong (McGovern, 2011). The aim for this is for 90% or more of participants to get the first click correct.

The final alternative considered is the five second test. Five second tests are a method of user research that help you measure what information users take away and what impression they get within the first five seconds of viewing a design (Five second tests, 2020). The main goal of this type of testing is to gauge users' first impressions.

To carry out a five second test, participants are given a screenshot of a specific page of the application to view for five seconds and are then asked questions about the screenshot. Generally, the questions are about the appearance of the application, participants opinions on branding, or who they believe the intended user of the application would be.

For this dissertation, the five second test would not be appropriate as there is little branding to be tested within the application. This combined with wanting to test the user interface instead of the design and visual hierarchy, makes SUS and first click testing more relevant to this dissertation.

Although SUS or first click testing could be used to test and evaluate this application, they show completely different metrics. A SUS study results in a single point of reference to the overall opinion of the participants regarding the usability of the system, whereas with first click testing, each feature can be tested individually to find what specific UI elements can be improved upon.

## Chapter 3: Requirements Analysis

This chapter will outline the requirements of the system. The requirements are split into two distinct groups which are functional requirements and non-functional requirements. Functional requirements are those which describe the functionality of a system. Non-functional requirements are those which represent a set of standards that can be used to gauge how well the system operates. This chapter will also include a section of revisiting requirements, which will add new requirements to the system which should have been thought about previously, however, were overlooked.

### 3.1 Functional Requirements

This section will list the requirements which describe functionality. The list will be split into sections in accordance to the requirements priority within a MoSCoW analysis. MoSCoW represents must have, should have, could have, and won't have priorities.

#### 3.1.1 Must have:

These are the requirements which are critical to the system and must be completed in order to have a completed system.

**FR 1: The system must store information in a database.** The data stored will include the events of the calendar to be accessed through the internet and user information such as log in details and contact information.

**FR 2: The system must allow users to log in.** The system must allow users to log in to ensure only designated caregiver users can view a patient's information.

**FR 2.1: The system must distinguish between caregiver and patient users.** Upon logging in, the system must know if the user is a caregiver or a patient user and display the corresponding user interface.

**FR 3: The system must have a calendar.** The system must include a calendar which can be viewed and updated. This is to allow for proper scheduling of a patient's needs.



**FR 3.1: All users must be able to view calendar.** Users must be able to view the calendar to allow caregivers to view the patient's schedule, and the patients to view their own schedule.

**FR 3.2: Caregiver users must be able to update the calendar.** Only caregiver users must be able to update the patient's calendar. This is to ensure simplicity for patient users while allowing for adding and modifying a patient's schedule.

**FR 4: Caregiver users must be able to add new caregiver users to patient user.** As caregivers change, it is a necessity to allow caregiver users to add new caregivers to a patient user.

**FR 5: Patient users must be able to view current date and time.** To help with the repetitive questioning behaviour, it is a must that patient users can view the current date and time easily on the system.

### 3.1.2 Should have:

These are the requirements which are important to the system but are not essential to having a completed project.

**FR 6: Patient users should be able to view current weather forecast.** To allow patient users a larger link to the outside, patients should be able to see a weather forecast for their local area.

**FR 7: System should be able to contact caregiver users with updates to the calendar.** When an event is re-scheduled or updated on the calendar, a caregiver user should receive an update message either through email or text message.

**FR 8: System should be able to gather data from patient users' environment.** Using sensors, the system should be able to get data about the patient's home, such as the internal temperature, to ensure their home is not too warm or too cold which, according to the research (pages 13-14), could be dangerous for the patient.

**FR 8.1: System should be able to send alerts to caregiver users about patient users' environment.** The system will allow for setting thresholds which, when the data collected from the sensors, are passed, an alert through email or text message will be sent to the caregiver users.

**FR 9: System should be able to send notifications to caregiver users when calendar is updated, or new caregiver user added to their patient user.** The system will send notifications to the caregiver users to ensure modifications made are correct and ensure all caregivers understand what is happening in regard to the patient's schedule.

**FR 10: System should allow for adding images to patient user's view.** As research has suggested (pages 12-13), images can help to draw attention the system. Due to this, there should be images in the system for the patients view.

**FR 10.1: System should allow for changing the image.** To allow caregiver users to add photos of family members, the system would have to allow for changing the image along with uploading images to the system.

**FR 10.2: System should allow for adding captions to the images.** As research has suggested, it is beneficial for the patients to have an image with a caption to help them remember who or what is in the image.

### 3.1.3 Could have:

These are the requirements which would be nice to have but will only be completed should there be left over time in which to develop them.

**FR 11: System could send notifications to caregiver users should the patient user's door be opened.** To benefit the caregivers, a sensor could be added to the patient's door to act as an alarm which contacts the caregivers if the door is opened during set times.

**FR 12: System could allow for changing layout of patient users view.** To allow for personalization, the system could allow for the layout of the patient's UI to be changed by the caregiver users.

#### 3.1.4 Won't have:

These are the requirements which have been considered but will not form part of the completed system.

**FR 13: System won't be voice activated by any user.** Voice activation would be outside the scope of this dissertation.

**FR 14: System won't be capable of ordering food for patient user.** Ordering food through the system may be beneficial for caregivers and patients, however, it will be outside the scope of this dissertation.

**FR 15: System won't allow for video calling between patient user and caregiver user.** As the proposed system does not have a video camera, it would not be possible to add video calling functionality. The added complexity of this would be outside the scope of this dissertation.

**FR 16: System won't allow for modifying the temperature of the patients central heating.** This would add complexity as it would entail modifying a patient's house and modifying the patients central heating, hence it is outside the scope of this dissertation.

### 3.2 Non-Functional Requirements

This section will list the requirements which describe how the system will work. The list will be split into sections in accordance to the requirements priority within a MoSCoW analysis. MoSCoW represents must have, should have, could have, and won't have priorities.

#### 3.2.1 Must have:

These are the requirements which are critical to the system and must be completed in order to have a completed system.

**NFR 1: The system must be secure.** As the patients are vulnerable, it is extremely important to ensure all user data is secure and to ensure non-authorized users cannot see a patient's schedule.

**NFR 2: The system must allow for all users to access simultaneously.** The system must allow for as many concurrent users as possible without down time should too many users connect simultaneously.

**NFR 3: The system must correctly handle multiple users updating same event simultaneously.** To ensure issues do not arise when updating a patient's schedule, the system must handle concurrency appropriately.

**NFR 4: System must be fully tested.** To ensure the system is a valid solution, everything must be tested. This will be done with unit tests, performance tests and security tests; and by following the test-driven development programming principal along with continuous integration and continuous deployment to run the unit tests.

### 3.2.2 Should have:

These are the requirements which are important to the system but are not essential to having a completed project.

**NFR 5: The system should be optimized to ensure requests are as quick as possible.** The system should respond as quickly as possible to ensure the patients view of the calendar is always relevant.

**NFR 6: The system should be portable, working on a range of systems.** As a multiple of different screen sizes are used in people's daily lives, the caregiver side of the system should work on a range of devices such as tablets, computers, and mobile phones; as well as across multiple operating systems such as iOS, Android, Windows, Linux and MacOS.

### 3.2.3 Could have:

These are the requirements which would be nice to have but will only be completed should there be left over time in which to develop them.

**NFR 7: The system could be translated to multiple languages.** To allow for expansion to different countries, the system could be translated into some other major languages such as Spanish, French or Chinese amongst others.

### 3.2.4 Won't have:

These are the requirements which have been considered but will not form part of the completed system.

**NFR 8: The system won't be connected directly to the council's current system.** While integrating the system with the current system used by a local council would be the end goal, this would not be possible to do as part of this dissertation.

## 3.3 Requirements Revisited

This section will discuss requirements that were added to the system although there was no original plan for these requirements. These requirements will not provide revolutionary changes to the overall application and are more so to improve the quality of the application through small additions which should not have been overlooked in the original requirements analysis.

**FR 17: The system should allow caregivers to modify the temperature at which they are alerted in regard to a patient's environment.** While the default temperatures are set to a minimum of 15° C and a maximum of 28° C, caregiver users should be able to change this if the patient perhaps feels uncomfortable when it is below 20° C.

**FR 18: The system should show caregivers information about their patient.** While it is not a necessity, it would be helpful for caregivers to be able to quickly see which patient's calendar they are updating along with the maximum and minimum temperatures used for sending temperature notifications.

## Chapter 4: Development

This section will show the features created throughout the development process, matching these features to the requirements they satisfy. It will also show the tools and technologies used to help with development, the features these tools were used in and a brief description of the purpose of using these tools. Finally, this section will serve as a way of expressing the work carried out along with issues that were encountered throughout the process.

### 4.1 Tools & Technologies

This section will discuss the tools and technologies used to develop the application. It will show all the tools, libraries, third party services, APIs and frameworks used, along with the purpose of these, the feature they are used in and where more information can be found about them. It will also discuss a couple of these tools more in depth while explaining their advantages.

While developing this application, a number of tools and technologies were used to efficiently develop a system which is of high enough quality to be considered a fully finished and polished application.

**Table 1:** Details of the tools and technologies used in this dissertation.

Name	Feature	Link	Purpose
Laravel Forge	DevOps and CI/CD	<a href="https://forge.laravel.com/">https://forge.laravel.com/</a>	Managing the web server.
AWS EC2	DevOps	<a href="https://aws.amazon.com/ec2/">https://aws.amazon.com/ec2/</a>	Servers for storing the web application.
Nginx	DevOps	<a href="https://www.nginx.com/">https://www.nginx.com/</a>	Web server.
Name Cheap	DevOps and Alerts	<a href="https://www.namecheap.com/">https://www.namecheap.com/</a>	Configure the DNS settings and to buy the domain.
GitHub	CI/CD	<a href="https://github.com/">https://github.com/</a>	Host the projects repositories.
Buddy Works	CI/CD	<a href="https://app.buddy.works/">https://app.buddy.works/</a>	Automate testing and deployment.
Laravel	All features of the web application	<a href="https://laravel.com/">https://laravel.com/</a>	Framework used for the backend of the application.
Vue JS	All frontend features of the web application	<a href="https://vuejs.org/">https://vuejs.org/</a>	Framework used for the front end of the application.

Bootstrap	All frontend features of the web application	<a href="https://getbootstrap.com/">https://getbootstrap.com/</a>	Style the web application.
Guzzle HTTP	Weather feature	<a href="http://docs.guzzlephp.org/">http://docs.guzzlephp.org/</a>	Create and send HTTP requests to external services.
MailGun	Alerts	<a href="https://www.mailgun.com/">https://www.mailgun.com/</a>	Create and send emails.
Twilio	Alerts	<a href="https://www.twilio.com/">https://www.twilio.com/</a>	Create and send text messages.
Sentry	All features of the web application	<a href="https://sentry.io/">https://sentry.io/</a>	Log and store server errors.
FullCalendar	Calendar feature	<a href="https://fullcalendar.io/">https://fullcalendar.io/</a>	UI for a basic calendar.
Moment.JS	Weather feature	<a href="https://momentjs.com/">https://momentjs.com/</a>	Format date and time data in JavaScript.
Axios	Calendar, Image and weather features	<a href="https://github.com/axios/axios">https://github.com/axios/axios</a>	Send HTTP requests in JavaScript.
Laravel Passport	User authentication feature	<a href="https://github.com/laravel/passport">https://github.com/laravel/passport</a> <a href="https://laravel.com/docs/master/passport">https://laravel.com/docs/master/passport</a>	Authenticate users.
HTML5 Geolocation API	Weather feature	<a href="https://developer.mozilla.org/en-US/docs/Web/API/Geolocation_API">https://developer.mozilla.org/en-US/docs/Web/API/Geolocation_API</a>	Get user's longitude and latitude.
RaspberryPi 3 model B	Hardware feature	<a href="https://www.raspberrypi.org/products/raspberry-pi-3-model-b/">https://www.raspberrypi.org/products/raspberry-pi-3-model-b/</a>	Physical system used for the hardware system.

The main decision chosen was using the Laravel framework. Laravel was chosen to make the code high quality, readable, reusable, and secure, as it contains many features which can be implemented to ensure the user's data remains safe from malicious attacks. Another reason Laravel was chosen is due to the MVC pattern. Following concepts from a pattern helps with readability along with adding a level of professionalism.

Moment.JS and FullCalendar were used as a way of speeding up development. The main reason the speed up of development was needed is due to the Covid-19 lockdown. As rumours circulated on a likely lockdown of the UK, I changed the timescale for development of the features needed for user

evaluation to ensure this would be completed before a potential lockdown. This turned out to be one of the best ideas in the development of this system.

## 4.2 DevOps

This section will discuss the use of DevOps in the development of this application. DevOps is a catch-all term for the blending of roles between developers and operations engineers (NGINX, n.d.). Essentially, it is the creation and maintenance of servers, the development process of applications, and the automation of getting the applications onto the servers. DevOps, CI/CD and automation of testing are terms which are gaining traction within software engineering. Due to this, I decided to implement DevOps techniques into the project. The DevOps for this project can be divided into two main parts, the server & domain setup and the CI/CD.

### 4.2.1 Server & Domain Setup

This section will outline the server architecture along with how the domain was set up. It will also express any issues found with this along with the key aspects learnt when setting up the server & domain.

The server used for this application is an AWS EC2 (Elastic Cloud Computing) instance with an Ubuntu server. This instance has been modified to accept HTTP and HTTPS traffic to allow for using it as a web server. Once the server was set up, I used SSH to access the server and install Nginx. Nginx consistently beats Apache and other servers in benchmarks measuring web server performance (Nginx, n.d.). Due to this along with being able to recreate the server environment locally using Valet, led me to choose this route for the server.

As the domain is registered with Name Cheap, all DNS records were configured on their website.

**Table 2:** Details of the DNS records for the *akal.app* domain

Type	Host	Value	Purpose
A Record	@	Server's IP address	Points domain to the server.
CNAME Record	www	Akal.app	Points <a href="http://www.akal.app">www.akal.app</a> to the same server as akal.app
A Record	Demo	Server's IP address	Creates a subdomain "demo" and points it to the server.



TXT Record	@	v=spf1 include:eu.mailgun.org ~all	Allows MailGun to edit all mail servers.
MX Record	Mg	mx.a.eu.mailgun.org.	Sets the mail server for accepting email to MailGun.
MX Record	Mg	mx.b.eu.mailgun.org.	Sets the mail server for accepting email to MailGun.

This was a learning experience as I have never had to manage DNS records before. This project gave me a reason to look into DNS and understand what it does, how it works and what different types of records are.

The main issue encountered was the records seemed to not be updating. I changed it back and forward countless times to get the mail server to work. When I eventually decided to try again the next day it was working as expected. This is due to delays in the propagation of the DNS records throughout the internet.

The last part of the server and domain set up was setting up Laravel Forge to use the Nginx server. After completing this, I realized I was able to set up all of the servers directly through Laravel Forge which I will be doing in future projects. Thanks to this realization, the MySQL database was created very quickly through the wizard provided by Laravel Forge.

#### 4.2.2 CI/CD

CI/CD, continuous integration continuous deployment, is a tool used by DevOps. It allows for the automation of the testing and deployment process, and for ensuring new features do not accidentally break features which were previously developed and working.

In order to create the CI/CD system, I needed to have my git repositories hosted on an online service. The chosen service was GitHub due to being familiar with it. Within my repository, I created a staging branch, from which I created branches for different tests and features. These branches would then be merged to the staging branch for the CI/CD system to access.

I also had to configure Buddy Works to automate my testing. To do this, I had to create a database within buddy, and set scripts to be performed. These scripts include pulling from the staging branch of the GitHub repository, installing the project with composer, and running the unit tests. When this was

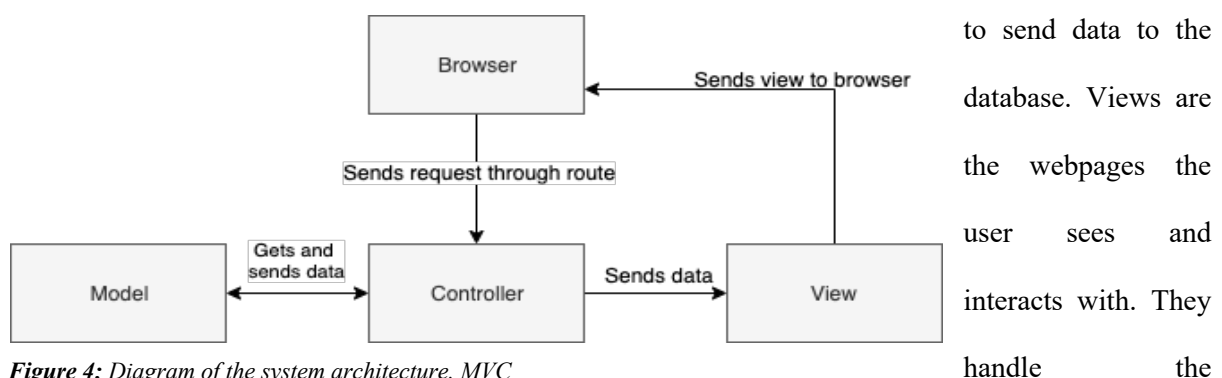
done, the last thing left was to set buddy to run the scripts when new code is pushed to the staging branch.

After the tests run on Buddy, if they pass it pushes the code to the master branch and sends a HTTP request to Laravel Forge to pull the newest version of the web application and install it on the server. If it fails however, it would send me an email to notify me that a test failed, along with information on which test failed.

### 4.3 Web Application

This section will discuss the features of the web application and the API. It will give a brief outline of what the feature does, an explanation of why it was done in that way where appropriate, and which functional requirements the feature satisfies.

The web application is built upon the Laravel framework. Laravel uses routes, models, views and controllers to generate webpages for the users. Routes are the URLs of the application, so everything that comes after `www.akal.app` in this application. Controllers are called by the route and are passed the request details, along with the function to be called in the controller. Controllers deal with the application logic. They get data from Models and return either JSON data for API requests, or Views for a user to interact with through the use of a web browser. Models are a bridge between the application and the database. They are objects which can be interacted with to get the data from the database and



**Figure 4:** Diagram of the system architecture, MVC

presentation logic by using Vue and Blade to dynamically add data to the view. This data can be taken from anywhere, however, in this application it tends to be taken from API requests to the system, hence, from the Controllers and Models.

### 4.3.1 User Authentication

This feature allows users to log in, log out, and register. It satisfies FR 1, FR 2 and FR 2.1.

Laravel passport authentication was used to develop this feature to ensure authentication was secure.

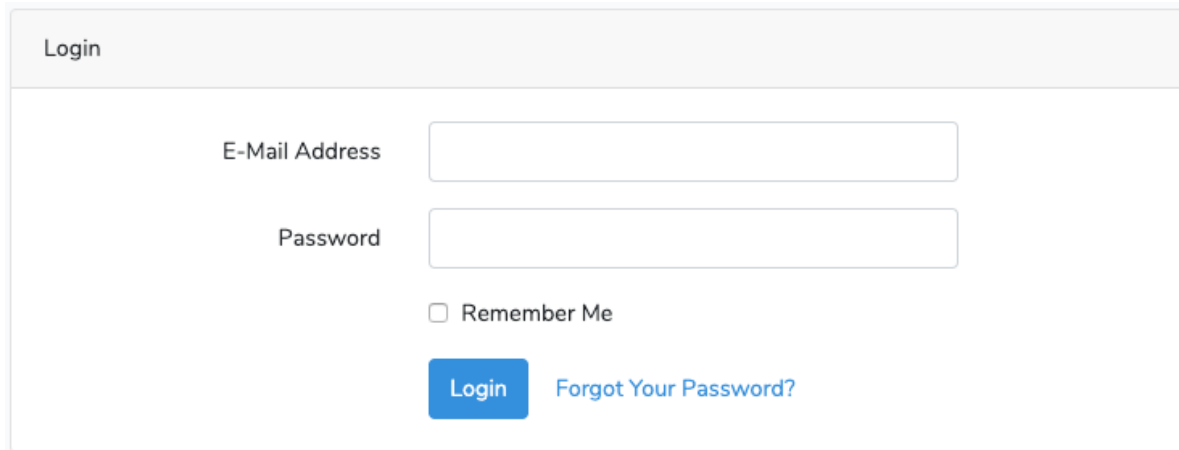
The passport authentication had to be modified however to allow for storing a user type, a patient email, and a phone number. The controller for the registration feature also had to be modified to create a new temperature and layout entries for the database if the user was a patient. The registration features create function works like so:

```
protected function create(array $data)
{
    if($data['userType'] == 'patient') {
        Temperature::create([
            'patientEmail' => $data['email'],
            'minTemp' => 15,
            'maxTemp' => 28
        ]);

        layout::create([
            'email' => $data['email'],
            'layout' => "1"
        ]);
    }
    return User::create([
        'name' => $data['name'],
        'email' => $data['email'],
        'password' => Hash::make($data['password']),
        'userType' => $data['userType'],
        'patientEmail' => $data['patientEmail'],
        'phoneNumber' => $data['phoneNumber']
    ]);
}
```

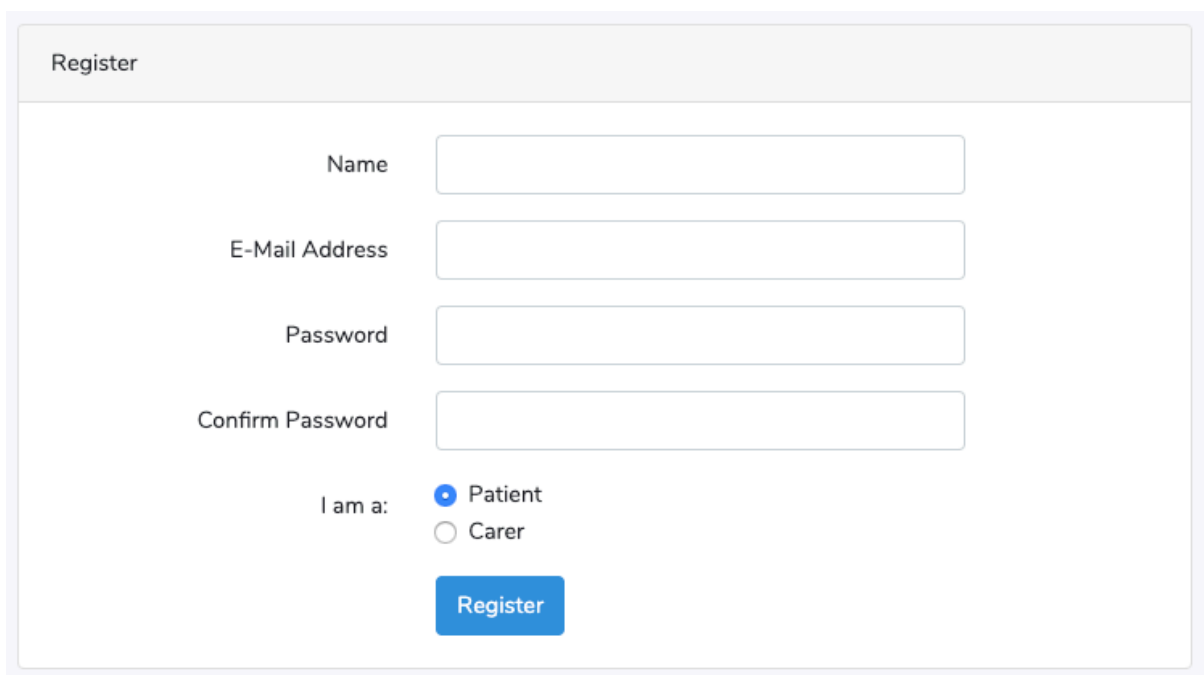
It checks the requests data to see if the user is a patient, if so, it creates a new temperature database entry with default temperature settings, and the patient's email field as the email address passed in the request. It also creates a new layout database entry with the default layout and the user's email address. It then follows on with the logic of creating a new user in the database by storing the data sent in the request to the database, while hashing the password to ensure it is secure.

These screenshots show the views for logging in, registering as a patient, and registering as a caregiver. The user model along with the login controller can be found in the appendix under the 2.1 and 2.2 headings.



The screenshot shows a 'Login' form with a light gray header. Below the header, there are two input fields: 'E-Mail Address' and 'Password'. Below the 'Password' field is a checkbox labeled 'Remember Me'. At the bottom, there is a blue 'Login' button and a blue link labeled 'Forgot Your Password?'.

**Figure 5:** Screenshot of login form



The screenshot shows a 'Register' form with a light gray header. Below the header, there are four input fields: 'Name', 'E-Mail Address', 'Password', and 'Confirm Password'. Below the 'Confirm Password' field is a label 'I am a:' followed by two radio buttons: 'Patient' (which is selected) and 'Carer'. At the bottom, there is a blue 'Register' button.

**Figure 6:** Screenshot of registering as a patient

The screenshot shows a web form titled "Register" with a light gray header. The form contains several input fields and a submit button. The fields are arranged vertically: "Name", "E-Mail Address", "Password", "Confirm Password", "Patient's Email", and "Mobile Number". Each field is a white rectangle with a thin gray border. Below the "Confirm Password" field, there is a section labeled "I am a:" with two radio button options: "Patient" (unselected) and "Carer" (selected, indicated by a blue dot). Below the "Patient's Email" field is the "Mobile Number" field. At the bottom of the form is a blue button with the text "Register" in white.

**Figure 7:** Screenshot of registering as a carer

#### 4.3.2 Calendar Feature

This feature creates a calendar for users to see. It also allows for caregiver users to add new events, delete events, and update events. Each user can only see events related to them, for instance, patients can only see events made by their caregivers and caregivers can only see events created for their patients. This feature satisfies FR 1, FR 3, FR 3.1 and FR 3.2.

This feature is the most complex in the web application as two versions of the calendar had to be created. One for patients and one for caregivers. Although the basic structure of the calendar was created with FullCalendar, the interactivity had to be created for the calendar such as getting and displaying the correct events, adding new events, deleting an event and updating an event. To do this, a calendar controller was created along with a calendar model and a database migration. A snippet of code from the Vue calendar component can be found in the appendix, 2.3, along with the rest of the calendar controller's functions, appendix 2.4.

```

public function up()
{
    Schema::create('calendars', function (Blueprint $table) {
        $table->bigIncrements('id');
        $table->string('event_name');
        $table->string('start_date');
        $table->date('end_date');
        $table->string('patient_email');
        $table->timestamps();
    });
}

```

This is the migration for the calendar feature. All features which require their own database have similar migrations which are used to create tables in the database.

```

/**
 * Store a newly created resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $patientEmail = auth::user()->patientEmail;

    $new_calendar = Calendar::create(array_merge($request->all(),
['patient_email'=>"$patientEmail"]));
    return response()->json([
        'data' => new CalendarResource($new_calendar),
        'message' => 'New event added',
        'status' => Response::HTTP_CREATED
    ]);
}

```

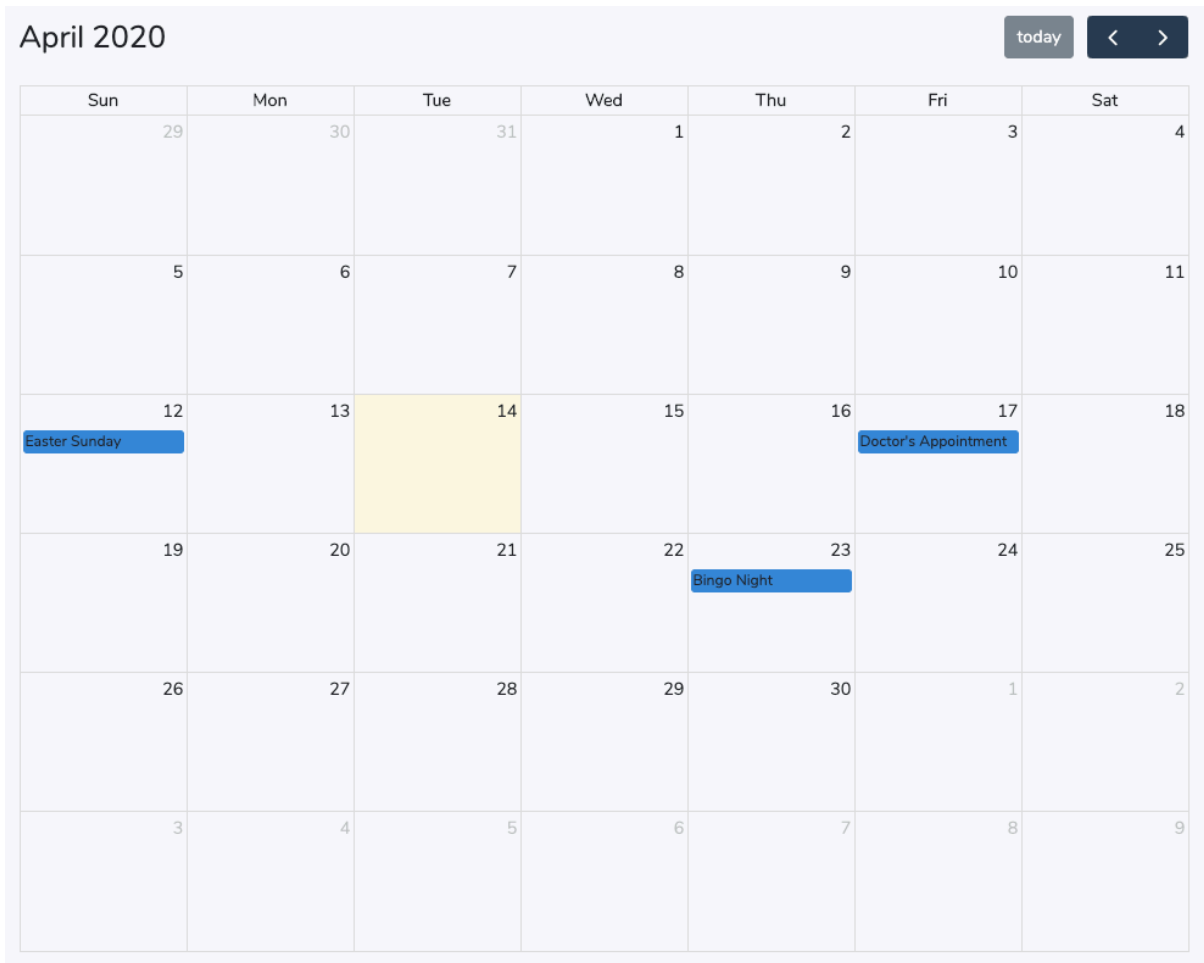
This is the store function of the calendar controller. This function deals with post requests to the calendar route and is used to store the data in the calendars table.

```

class Calendar extends Model
{
    protected $fillable = [
        'event_name',
        'start_date',
        'end_date',
        'patient_email'
    ];
}

```

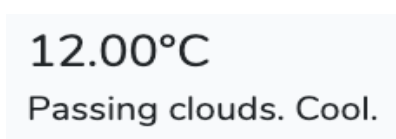
This is the model for the calendar which dictates the data that is fillable for a calendar event.



**Figure 8:** Screenshot of the calendar feature

#### 4.3.3 Weather Feature

This feature allows for getting the users longitude and latitude from the geolocation API. It sends that



**Figure 9:** Screenshot of the weather

information to the applications API which sends it to Here's weather API. Here's weather API returns a lot of information that would confuse an elderly patient; hence, the data is manipulated to only

display the current temperature and a brief description of the weather. This feature satisfies FR 6.

The main issue encountered while developing this feature was a cross-origin resource sharing (CORS) error. CORS is a mechanism that uses additional HTTP headers to tell browsers to give a web application running at one origin access to selected resources from a different origin (MDN contributors, 2020). Due to this and not being able to bypass this error using Axios, I created my own API route which would send a Guzzle request for the data. The following code snippet shows how this was done within the weather controller.

```

class WeatherController extends Controller
{
    public function index($lat, $long) {
        $client = new \GuzzleHttp\Client();
        $request = $client->get(
            "https://weather.ls.hereapi.com/weather/1.0/report.json?product=observation&latitude=$lat&longitude=$long&oneobservation=true&apiKey=VCmEZ0kuIfLP-8jdwh8lV5VTstkD8LQR40XqnWkPsc");

        $response = $request->getBody()->getContents();

        return $response;
    }
}

```

The second issue was having to assure the application could get the user's location. This can only be done through HTTPS leading to the installation of SSL certificates on the server to ensure HTTPS is able to be used. The following code snippet has been edited to only show code relevant to this feature.

```

created() {
    //do we support geolocation
    if(!("geolocation" in navigator)) {
        this.errorStr = 'Geolocation is not available.';
        return;
    }
    navigator.geolocation.getCurrentPosition(pos => {
        this.location = pos;
        let long = pos.coords.longitude;
        let lat = pos.coords.latitude;
        let url = '/api/weather/' + lat + ',' + long;
        axios.get(url)
            .then(data => {
                this.temperature =
                    data.data.observations.location[0].observation[0].temperature;
                this.tempdesc =
                    data.data.observations.location[0].observation[0].description;
                console.log(data.data.observations.location[0].observation[0]);
            })

        }, err => { //display error messages to console
            console.warn(err.message);
        })
}

```

This code snippet shows the created method of the Vue component. It checks if access to the geolocation API has been provided. If it has not, it displays an error message. If it has, it gets the user's location and sends the latitude and longitude to the API. From the response, the temperature and description are saved in state memory to be displayed to the user. If there has been an error at any point, it sends a warning to the console with the error message.

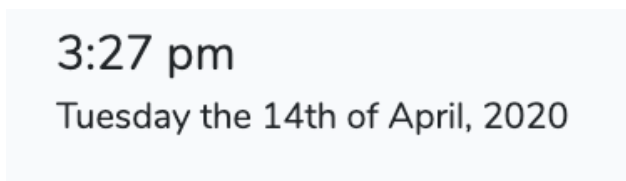


#### 4.3.4 Date & Time Feature

This feature gets the current date and time and displays them to the patient user. It satisfies FR 6.

This feature is set to update every 1000 milliseconds, ensuring the date and time updates as if it were a digital clock. The following code snippet has been modified to only show code relevant to this feature.

```
created() {  
  this.interval = setInterval(this.time, 1000);  
},  
methods: {  
  time: function() {  
    this.datenow = moment().format('dddd [the] Do [of] MMMM, YYYY');  
    this.timenow = moment().format('h:mm a');  
  }  
}
```



**Figure 10:** Screenshot of date and time feature

This shows an interval which begins when the Vue component is created. The interval calls the time function every second. The time function can be found within the methods of this

component and sets the date and time using Moment.JS for formatting.

#### 4.3.5 Patient Details Feature

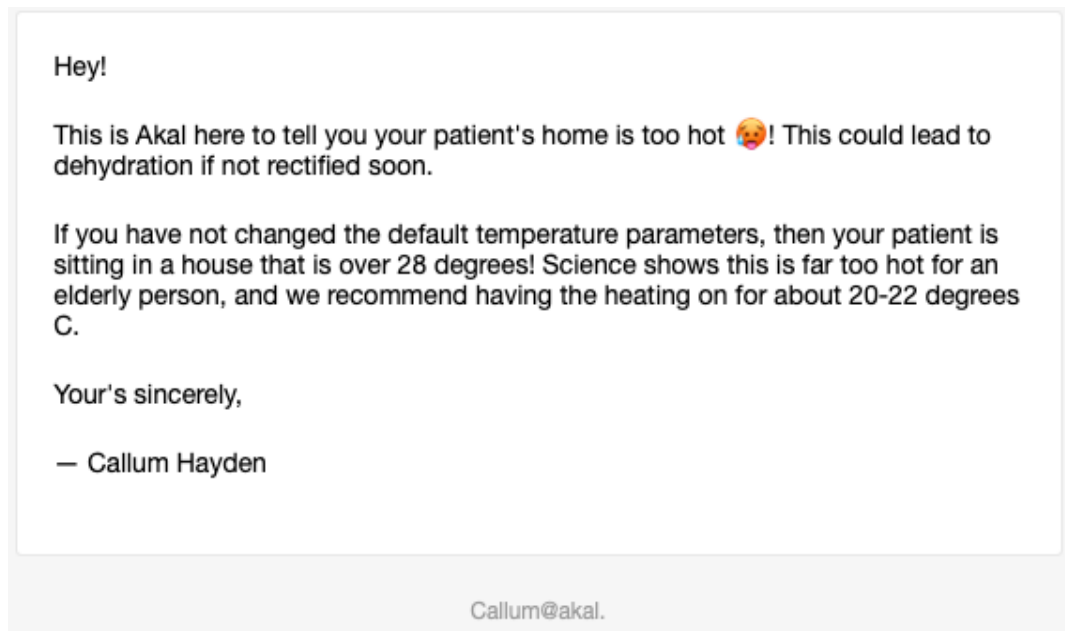
While this feature was not required in the original requirements, it was added as a requirement later on when revisiting the requirements. This feature gets the patient's name, email address, maximum temperature and minimum temperature and displays it to the caregiver. This allows caregivers to quickly see basic information about their patient user. The controller gets the data like so:

```
public function index()  
{  
  $patientEmail = Auth::user()->patientEmail;  
  $patient = User::where('email', "$patientEmail")->get();  
  
  return json_encode($patient);  
}
```

The requirement this satisfies is FR 18. A screenshot of this can be found in the appendix as figure 17.

#### 4.3.6 Alert Feature

This feature is used to alert caregiver users about their patient's environment and any updates to the calendar. It sends text messages and emails which explain what the alert is for and what has changed.



**Figure 11:** Screenshot of the HTML email sent when patient's home is too warm

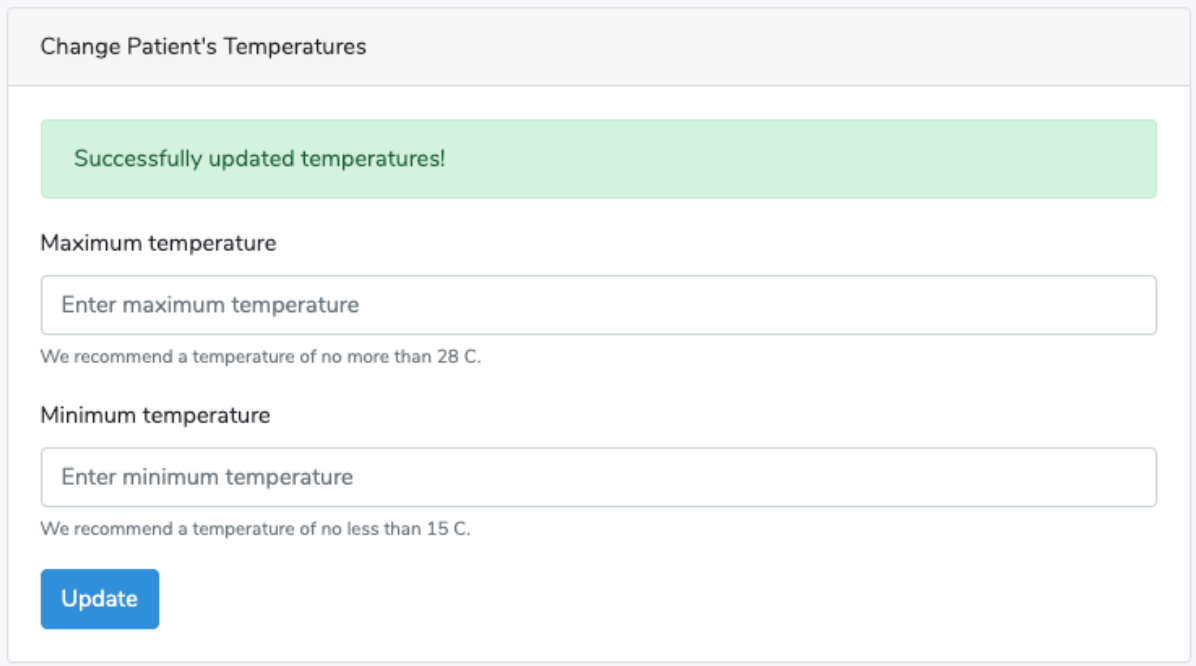
This feature satisfies FR 7, FR 8.1, FR 9 and FR 11. More screenshots showing the rest of the notifications can be found in the appendix as figures 18 – 25. Code snippets for the controller in charge of the alert functionality can be found in the appendix under heading 2.5, with a small modified code snippet below showing the logic of how an email is sent with MailGun configured for the application.

```
public function sendEmail($view, $data)
{
    Mail::send($view, $data, function($message) use ($data)
    {
        $message->to($data['email'], $data['name'])
                ->subject($data['subject']);
    });
}
```

The main challenge with this was having to learn more about DNS settings to ensure emails were sent from an email address associated with the Akal domain. Another element this feature allowed me to learn about was how to create HTML emails. While it was not a requirement to do this, it looks more professional than a standard text only email.

#### 4.3.7 Updating Patient's Temperature Feature

While this feature was not required in the original requirements, it was added as a requirement later on when revisiting the requirements. This feature allows caregiver users to update their patient's maximum and minimum temperature parameters satisfying FR 17.



**Figure 12:** Screenshot of form to update patient's maximum and minimum temperatures

While it is recommended that it stays as default, it allows for caregivers to have more control over the strictness or flexibility of when they will be alerted about their patient's temperature.

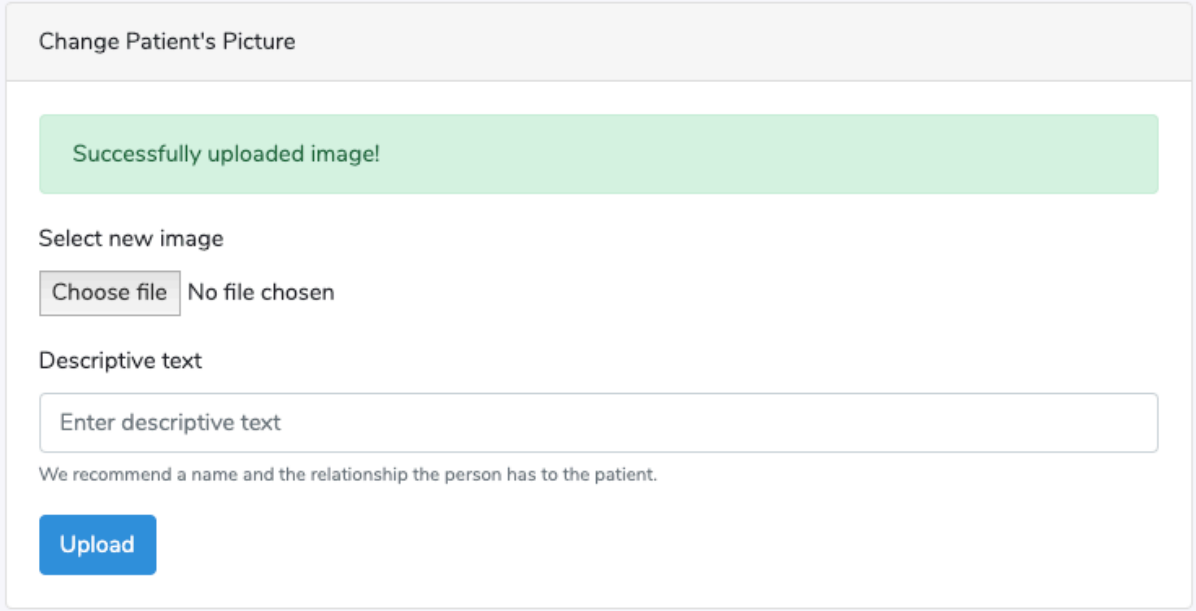
This feature's controller works by finding the temperature in the database and updating the maximum and minimum temperature to those sent from the form. This is the feature which updates a patient's temperature.

```
Public function updateTemperatures(Request $request)
{
    $patientEmail = auth::user()->patientEmail;

    Temperature::where('patientEmail',"{$patientEmail}")->update(array(
        'maxTemp' => $request->input('maxTemp'),
        'minTemp' => $request->input('minTemp')
    ));
    return back()
        ->with('tempStatus', "Successfully updated temperatures!");
}
```

#### 4.3.8 Image Upload Feature

This feature allows for caregiver users to upload images along with a caption to be displayed on the patient's view next to the calendar and below the weather. This satisfies FR 10, FR 10.1 and FR 10.2.



Change Patient's Picture

Successfully uploaded image!

Select new image

Choose file No file chosen

Descriptive text

Enter descriptive text

We recommend a name and the relationship the person has to the patient.

Upload

**Figure 13:** Screenshot of the image upload feature

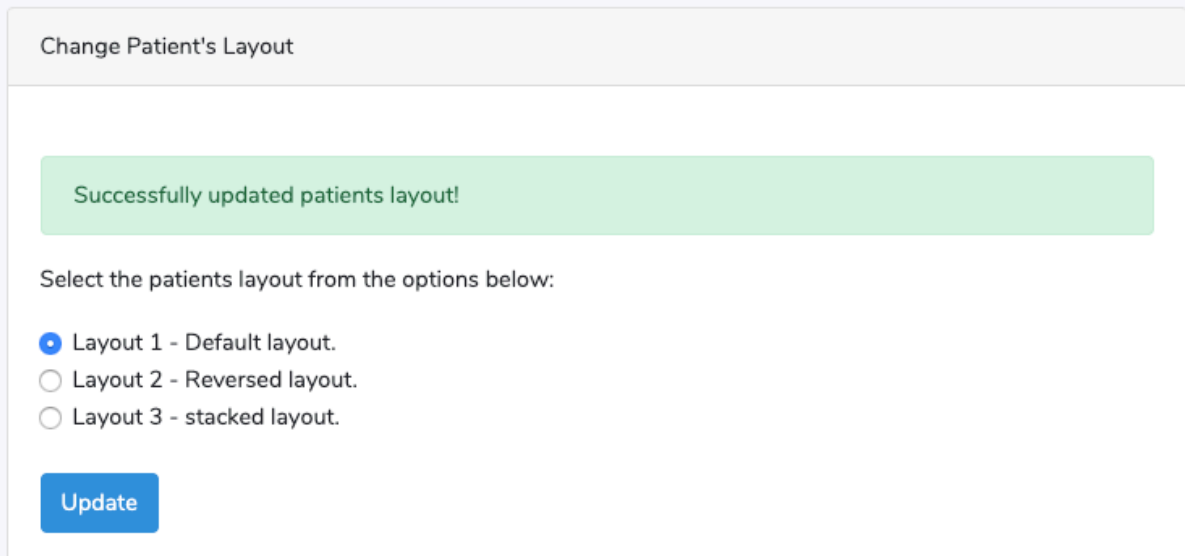
#### 4.3.9 Patient's Temperature API Feature

This feature allows for integrating the hardware system with the web application. It is an API route which is used by the patient's system to send the current temperature. If the temperature is not within a set of parameters it sends a custom alert stating the patient's home is too hot or too cold, along with an explanation of detriment this could have on the patient's health.

This feature uses the alert feature to alert the caregivers. This ensures the code is decoupled by being able to change the alert system to, for example, stop using Twilio to send text messages, without having to also change this feature. This can be seen in the code snippet in the appendix under heading 2.5. The function called when the temperature is sent to the API is the `alertTemperature($temp)` function, where the "temp" variable stores the temperature sent in the request.

#### 4.3.10 Multiple Layouts Feature

This feature allows for a caregiver user to change the desired layout of the patient's application. In order to complete this, two new home pages for patients were created and returned based on the layout stored in the database. This feature satisfies FR 12.



Change Patient's Layout

Successfully updated patients layout!

Select the patients layout from the options below:

☒ Layout 1 - Default layout.

☐ Layout 2 - Reversed layout.

☐ Layout 3 - stacked layout.

Update

**Figure 14:** Screenshot of the form used to change a patient's layout

### 4.4 Patient System

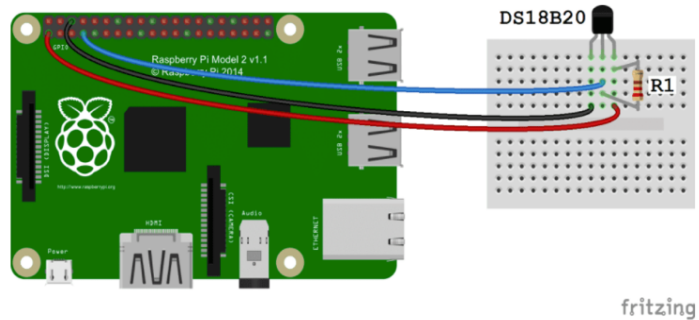
This section will discuss the features of the patient system. It will discuss the hardware and software used in this system, along with the features implemented on this system. The features will be mapped to the requirements to show which functional requirements each feature fulfils.

The main goal of the patient's system is to display the web application in kiosk mode. This means that although it is a web application, the web browser's interface would be hidden showing only the web application. The second goal of this system is to gather temperature data regarding the patient's environment and send this data via internet to the API. Deciding if the temperature is too low or too high is decided by the logic within the API and is not calculated within this system.

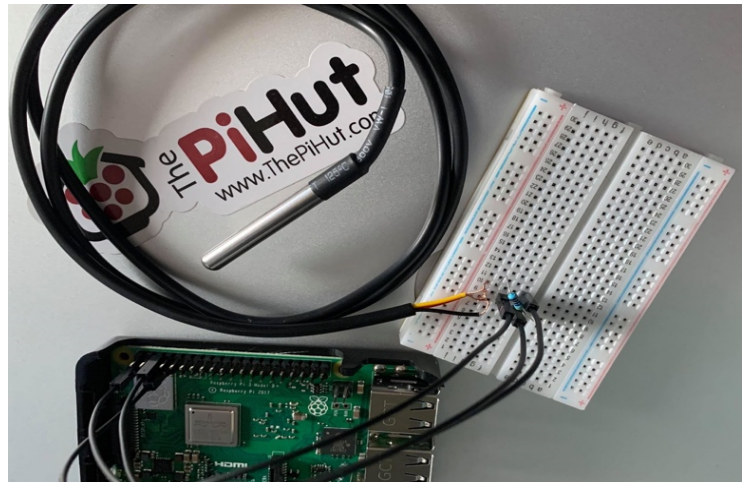
#### 4.4.1 Hardware

This section will discuss the creation of the hardware system, showcasing what was used to create this system and how the sensors are connected to the Raspberry Pi.

In order to build this system, I used a Raspberry Pi 3 Model B, connected to a temperature sensor via a breadboard and cables, with a resistor to limit the current going to the sensor. Before beginning this build, I created a diagram of how the wiring should be done with fritzing. Once the diagram was created, I began wiring the hardware. To improve upon this build, I would have liked to solder the ends of the temperature sensor. This would not only allow for creating connections easier, but it would also ensure the



**Figure 16:** Diagram of wiring for the Raspberry Pi and sensor



**Figure 15:** Image of the Raspberry Pi wired to the sensor

wires do not intertwine causing a short circuit or incorrect signals to be sent. However, I do not own a soldering iron hence this was impossible to do. I would have also liked to mount this system in an enclosed system with a monitor for output.

#### 4.4.2 Software

This section will discuss software which runs on the patient's hardware system. It will discuss how the Raspberry Pi communicates with the temperature sensor, how the temperature data is sent to the API, and how the web application is displayed on the patient's system

There are two different scripts that are used by the Raspberry Pi. The first script is a Python script. It deals with requesting the patient's login details. With these details, it sends a post request to Akal's login route to log the user in.

Once logged in, it gets the temperature from the sensor. This was the most challenging part of this the patient system. This is done with the `read_temp()` function which deals with getting the raw data from the sensor and converting it to a Celsius value. This can be seen here:

```
def read_temp():
    lines = read_temp_raw()
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        temp_c = float(temp_string) / 1000.0
        return temp_c
```

This function is passed as a parameter to the `send_temp()` function which sends the temperature to the API. The `send_temp()` function is called from within an infinite loop with a 900 second sleep statement, to ensure the temperature is sent only once every fifteen minutes. This is due to the fact that if the temperature would trigger an alert, and if the sleep was set to say 5 seconds, then every 5 seconds the caregivers would be alerted about the temperature.

The second script deals with creating the kiosk mode. This is done with a shell script which opens the Akal website in the chromium browser, but removes all browser qualities, hence creating a full screen view of the content on the webpage. This allows the patient to login and view their own calendar. This script also turns off the screen saver to ensure the screen always remains active, along with hiding the mouse. This is to create an illusion of the application being a simple calendar on a screen instead of a complex web application which may intimidate or scare the patient user. The final part of this script is an infinite loop which mimics the key presses of `ctrl + r` which forces the page to refresh. This also contains a sleep statement to sleep for 1800 seconds ensuring the calendar and weather are close to real time. This could be reduced to refresh more frequently, however, it may add unnecessary stress on the server at the current time. This is done like so:

```
while true; do
    xdotool keydown ctrl+r; xdotool keyup ctrl+r;
    sleep 1800
done
```

## Chapter 5: Testing

This chapter will discuss the tests carried out on the finished system from a technical standpoint. It will include an explanation of the unit tests, along with the results of these tests and the importance of them; an explanation of the functional testing, along with what functionality was tested, the expected outcome, and the actual outcome; and an explanation of how security was tested along with the outcome of the security testing.

### 5.1 Unit Tests

This section will focus solely on the unit tests carried out on the application. “Unit testing is testing of individual units or groups of related units” (Runeson, 2006). Although writing unit test code is labour-intensive, it is a practical approach to increasing the correctness and quality of software (Cheon & Leavens, 2002). The use of unit tests is not only important to follow the TDD programming process, but it also helps serve the purpose of testing if an application works as expected.

For the backend of the application PHPUnit tests were created. These tests allow for creating fake data which can then be given to the API. The system then uses the responses from the API to ensure the test conditions (assertions) have passed (return true). All of the PHPUnit tests can be found inside the test folder of the web application.

**Table 3:** List of unit tests carried out along with their assertions

Test	Assertions
User can send current temperature.	Status code 200.
User can view the calendar.	Status code 200.
Guest cannot view the calendar.	Status code 302 (redirect). Redirected to login page.
User can add events to calendar.	Status code 200.
Guest cannot add events to calendar.	Status code 302 (redirect). Redirected to login page.
User can update events.	Status code 200.
Guest cannot update events.	Status code 302 (redirect). Redirected to login page.
User can delete event.	Status code 204 (no content).



User can upload an image.	Status code 302 (redirect). Session contains successful image status.
Guest can view registration page.	Status code 200. View is registration page's view.
User cannot view registration page.	Status code 302 (redirect) Redirected to home page.
Caregiver can register.	Status code 302 (redirect) Redirected to home page. User is authenticated.
Patient can register.	Status code 302 (redirect) Redirected to home page. User is authenticated.
Guest can view login page.	Status code 200. View is login page's view.
User cannot view login page.	Status code 302 (redirect). Redirected to home page.
User can log out.	Status code 302 (redirect). User is not authenticated.
Guest can log in.	Status code 302 (redirect). User is now authenticated.
Invalid email/password cannot log in.	Session contains errors. User is not authenticated.
User can get weather forecast.	Status code 200.
Alert sent for temperature.	Status code 200.
Alert sent for door opening.	Status code 200.
User can change layout.	Status code 302 (redirect). Session contains successful layout status.
User can change temperature parameters.	Status code 302 (redirect). Session contains successful layout status.

These tests can be run by navigating to the applications folder within a console, and using the command:  
“./vendor/bin/phpunit”.

```
public function testUploadingImage()
{
    Passport::actingAs(factory(User::class)->create());

    $file = UploadedFile::fake()->image('avatar.png')->size(100);
```

```

$response = $this->json('POST', '/image-upload', [
    'image' => $file,
    'name' => "Just a test",
]);

$response->assertStatus(302)
    ->assertSessionHas('status', $value = "Successfully uploaded
image!");
}

```

This is a code snippet for testing the image upload which shows the creation of a fake user which uploads a fake image. All of the unit tests involve writing code like this, creating fake users and/or data and checking every API route.

## 5.2 Functional Testing

This section covers the functional testing of the application, explaining what functional tests are, how they were performed and the results from these tests. Functional testing is a type of black box testing that is performed to confirm that the functionality of an application or system is behaving as expected. It is done to verify the functionality of an application (Software Testing Help, 2020).

**Table 4:** List of functional tests carried out along with the results

ID	Functionality to test	Expected Result	Actual Result
1	Logging in as a registered caregiver.	Login successfully.	As expected.
2	Logging in as a registered patient.	Login Successfully.	As expected.
3	Logging in with incorrect details.	Error: credentials do not match our records.	As expected.
4	Registering with valid details.	Register successful and redirect to calendar.	As expected.
5	Registering with email that is already used.	Error: email already taken.	As expected.
6	Registering with invalid email (no @).	Error: email must include @.	As expected.
7	Adding new valid event.	Successfully add event.	As expected.
8	Adding new event, end date before start date.	Successfully add event where end data is the same as start date.	As expected.
9	Adding new event, no title.	Error: must include title.	As expected.
10	Updating event with normal input.	Successfully update event.	As expected.

11	Updating event to end date before start date.	Successfully update event where end data is the same as start date.	As expected.
12	Upload a non-image file.	Error, file not an image.	As expected.
13	Upload an image.	Upload successfully.	As expected.
14	Change patient's max/min temperature to number.	Successfully update patient's temperatures.	As expected.
15	Change patients max/min temperature to string.	Error, number expected.	As expected.
16	Change patient's layout.	Successfully update patient's layout	As expected.

As shown in the table above, all features were extensively tested, and all features acted as expected.

### 5.3 Security Testing

This section will talk about security vulnerabilities the system was tested against. It will include a brief introduction of what these vulnerabilities are, and why the system is safe or vulnerable to these attacks. The attacks tested against were SQL Injections and XSS attacks. A third security vulnerability was considered which is a cross site forgery request attack, however, as CSRF tokens are generated for each active user when a form is submitted, it is impossible for a CSRF attack to take place.

#### 5.3.1 SQL Injection

This section will talk about SQL injections. What they are, if they are possible to perform on this application and how this was tested. It will also discuss why it is not possible to perform SQL injections on this application.

SQL injection is a method of hacking in which malformed SQL queries are produced through un-sanitized user-input (Chandrashekhar, Mardithaya, & Santhi Thilagam, 2012). Essentially, it involves adding closing statements for an SQL query within an input field. This is then followed up with malicious queries and comments to remove the end of the original query.

Following the Netsparker SQL injection cheat sheet, all inputs were tested. I was unable to inject any SQL code into the application. An example of attempting to close an input early with a single quotation mark can be found in the appendix, figure 29 and with a double quotation mark, figure 30.

While the first figure shows an error for attempting to add the double quotation mark, the second figure shows a different result. The use of a single quotation mark was accepted, but due to how the queries are formed, it simply modified the input data without allowing access to the query itself.

This is due to the queries using the framework's query builder, which uses PDO parameter binding to protect the application against SQL injection attacks. This allows for creating forms for user input without having to worry about cleaning strings which are based as bindings (Laravel).

### 5.3.2 Cross Site Scripting Attacks

This section will talk about cross site scripting attacks. What it is, the application's vulnerability to cross site scripting attacks, and how this was tested. Leading to why it is not possible to perform cross site scripting attacks on this application.

Cross site scripting attacks (XSS) are a type of injection, in which malicious scripts are injected into websites. XSS attacks occur when an attacker uses a web application to send malicious code to a different end user (OWASP, n.d.).

XSS attacks were tested by attempting to add JavaScript code into all user input sections. The code used would have sent an alert if the attack was successful, however, there were no alerts displayed on any of the user inputs. Figure 31 and 32 in the appendix show attempting to perform an XSS attack on the login form.

This application is safe from XSS attacks thanks to the use of the Laravel framework. Laravel offers native support that protects the code from XSS attacks, with the feature automatically protecting the application and the database in the process. Any code that contains escape tags is outputted as HTML.

## Chapter 6: Usability Study

This chapter will explain the usability study. It will explain how the study was carried out and the documents that were given to participants, an overview of who the participants were, and an in-depth analysis of the results.

This study followed the original plan which can be found in the appendix (3.1). This uses the system usability scale to determine the usability of the system. There was a total of twenty-four participants, all of whom are caregivers that either work in caregiving or care for a person with Alzheimer's or dementia. Participants were given a brief introduction to the system explaining what the system is for and the purpose of this usability study, they were also given a consent form to fill in (appendix 3.2.1), along with the questionnaire (appendix 3.2.2) which includes a list of tasks to be completed. The following sections are a look at the participants' demography followed by an in-depth look at the participants' level of agreement with each statement from the questionnaire.

### 6.1 Demographics

This section will discuss the demographic profile of the participants of this study. There were three questions asked regarding the participant.

The first was if they work in caregiving as some participants worked in caregiving and some cared for family members. Out of the twenty-four participants, seventeen worked as caregivers (70.8%).

The second question was regarding the frequency they use caregiving applications. Out of the twenty-four participants, twenty-one (87.5%) said they had never used one, none of the participants said they occasionally use one, and three (12.5%) said they use one daily.

The final question about demographics was in regard to the participants' level of computer skills. Out of the twenty-four participants, fourteen (58.33%) said they would consider themselves beginners, eight (33.3%) said they would consider their level of IT skills to be at an intermediate level and two (8.3%) said they would consider their level of IT skills to be advanced.

## 6.2 System Usability Study Questions

This section will discuss the responses to the SUS questions of the questionnaire. It will share what I believe may have resulted in these answers. Each question will also be accompanied by a pie chart showing the responses. In these charts, the responses will be colour coded. Orange will show strongly agree, yellow will show agree, grey will show undecided, light blue will show disagree and dark blue will show strongly disagree.

### Q1: I think that I would use the system frequently.

Over 70% of participants agree or strongly agree that they would use the system frequently. This could be due to the lack of mainstream systems similar to Akal and be a novelty effect, or the participants may have found the system enjoyable to use and may truly believe they would frequently use the system in their day to day lives while taking care of patients.

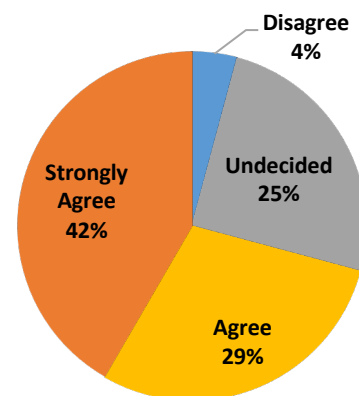


Figure 17: Chart showing responses to Q1

### Q2: I found the system unnecessarily complex.

Over 70% of participants disagree or firmly disagree with this statement. This shows that the system's user interface displays data in a non-complex manner and provides a good layer of abstraction between the inner workings of the system, and the user interface. Overall this shows good usability in terms of the complexity of the system.

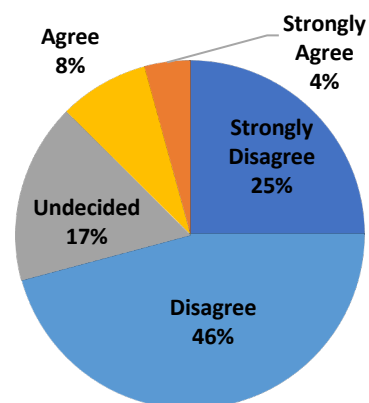
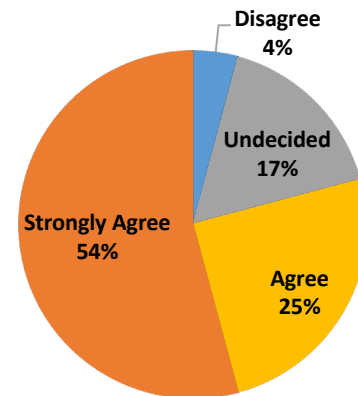


Figure 18: Chart showing responses to Q2

**Q3: I thought the system was easy to use.**

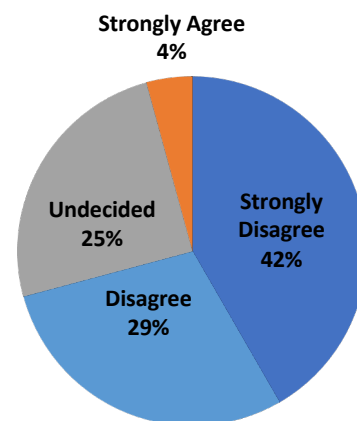
Over half of the participants strongly agreed with this statement. As the participants overall stated they were beginners when it comes to IT, this truly highlights the usability of this application. This could be due to the design of the system following standard design trends such as having login and logout buttons at the top right.



*Figure 19: Chart showing responses to Q3*

**Q4: I think that I would need the support of a technical person to be able to use this system.**

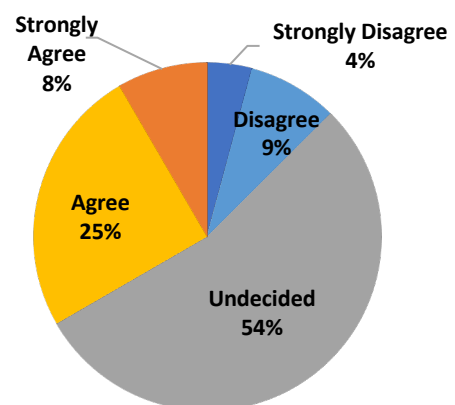
The majority of participants did not believe they would need the support of a technical person to use this system. This is the best-case scenario. On the downside, having 25% of participants being unsure if they would need support is slightly worrisome, and strongly makes me believe this system should be delivered with a short user manual, including a FAQ section regarding the hardware system.



*Figure 20: Chart showing responses to Q4*

**Q5: I found the various functions in this system were well integrated.**

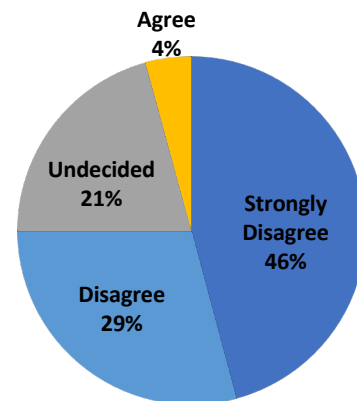
Over a quarter of participants agree, there were many who were undecided which may be due to the question and a lack of understanding of the system as participants only interacted with the caregiver system and did not get to experience the patient system's software nor hardware. In future I would perform this study again but including the patient system.



*Figure 21: Chart showing responses to Q5*

**Q6: I thought there was too much inconsistency in this system.**

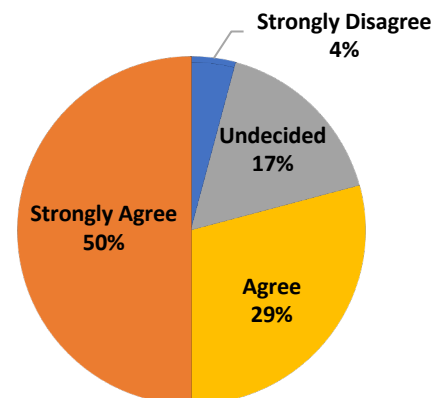
Three quarters of users disagreed or strongly disagreed with this statement, with almost the full last quarter being undecided. Only four percent of participants believed the system was inconsistent. This is likely due to sticking to the same design throughout the application. This was done to be able to reuse key parts of the application and ensure the system is not only useable but also has a professional feel to it.



*Figure 22: Chart showing responses to Q6*

**Q7: I would imagine that most people would learn to use this system very quickly.**

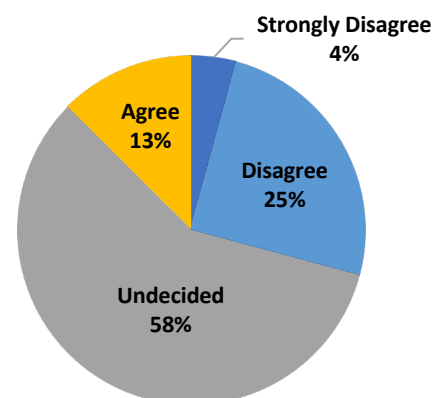
Half of all participants strongly agree with this statement. This shows the simplicity involved in using this system. The tasks performed by the participants included every task a caregiver user can perform with the system. Due to this, I believe this highlights the fantastic design of the caregiver user's features.



*Figure 23: Chart showing responses to Q7*

**Q8: I found the system very cumbersome to use.**

The majority of participants were undecided on this question, with an almost equal split each way for agreeing and disagreeing for the rest of the participants' responses. I believe this could be due to the question being hard to understand and should have reworded the question to "very awkward to use" as I believe this would have given vastly different results.



*Figure 24: Chart showing responses to Q8*



### Q9: I felt very confident using the system.

An overwhelming percentage of users agreed with this statement. This could be due to the system being easy to use hence instilling confidence in the user while using the application, or it could be due to feeling their data will be safe. A major fear of many people is that companies will sell their data, as all participants knew me, this would not be as much of a worry hence this could lead to the confidence felt by the participants.

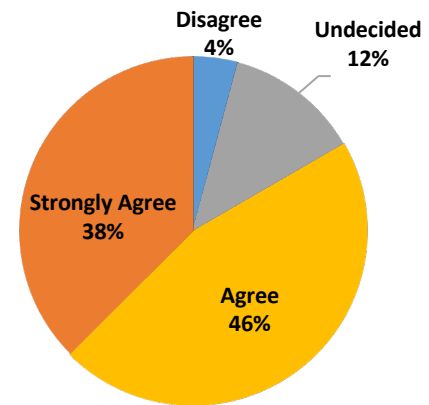


Figure 25: Chart showing responses to Q9

### Q10: I needed to learn a lot of things before I could get going with this system.

The majority of participants disagreed with this statement, meaning they do not feel as if the system has a steep learning curve. Based on some of the written feedback, this is likely due to “everything being where I expected it to be”. This ties in with the third question of if the participants believe the system is easy to use.

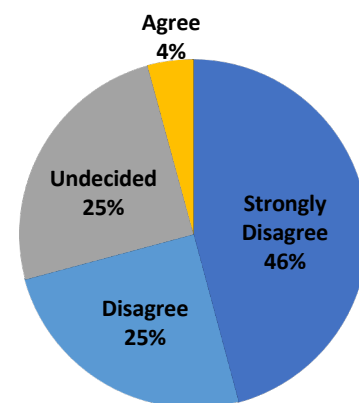


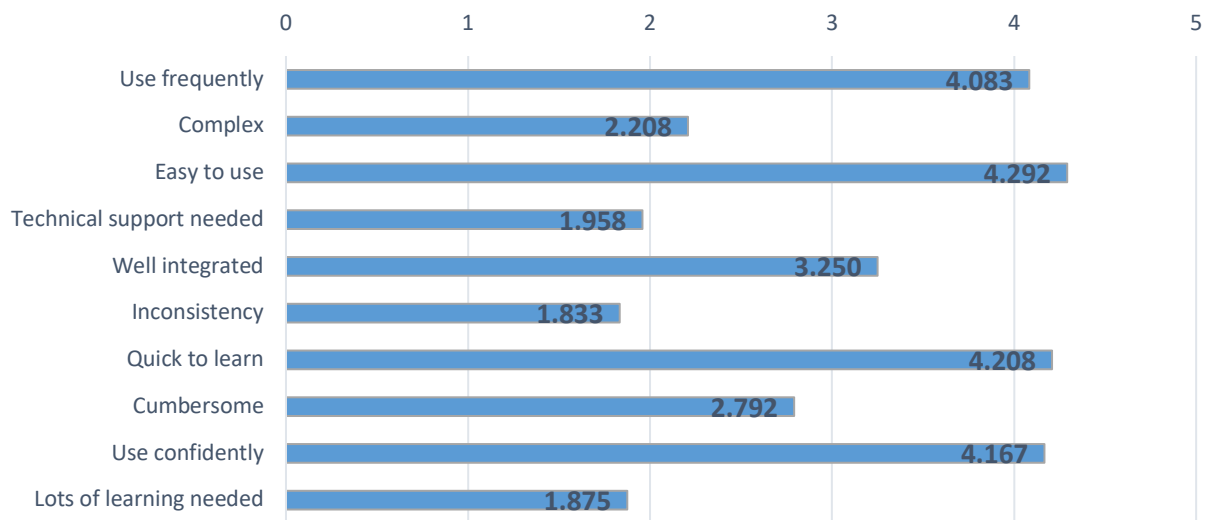
Figure 26: Chart showing responses to Q10

## 6.3 Conclusion

This section will conclude the results of the evaluation study and explain how the participants' responses were translated into a score. This score will be compared with the original aim of scoring above 68 to determine if this system is usable or if it needs more work.

As this study allowed participants to select a word instead of a score, to determine a number value each level of agreement was assigned a number from one to five with one being strongly disagree and five being strongly agree. These scores were then added together and divided by twenty-four (the number

of participants) to get an average score for each statement. These averages were then put into a graph for a visual representation which can be found below.



**Figure 27:** Chart showing an average score from the 24 participants for each question

These numbers however are not an accurate representation of the score. As all odd numbered statements are positive about the system and all even numbered statements are negative, to get the SUS score, each user's score for the even statements are subtracted from five, and for the odd numbered statements, one is subtracted from the score, giving us the accurate representation. These numbers are then added together and multiplied by 2.5, then averaged across all the participants to get a final usability score.

The final usability score is 73.33. The target score of 68 is the average SUS score, scoring higher than this is a success and I am extremely pleased this application scored above average, however, with rewording one of the questions I feel the score would have been even higher.

With SUS however, it does not pinpoint any specific points which can be improved to improve usability. Due to this, comments were requested, but very few were given. In the future I would aim to not only perform a SUS evaluation, but I would also aim to find what parts of the system need improved and how to improve it through using a different evaluation method, or simply adding non-SUS questions to the questionnaire.

## Chapter 7: Evaluation

This chapter will discuss the evaluation of this dissertation. It will evaluate this project based on the objectives of this project, the requirements of the project, how well it performed in testing, and the results of the usability study.

### 7.1 Objectives Evaluation

In this section, this dissertation will be evaluated based on whether it meets the original objectives. This will be outlined in a table and will include a brief conclusion on the performance of this dissertation in this evaluation.

**Table 5:** List of the objectives of this dissertation with their achieved status

Objective	Achieved?
Gather requirements.	✓
Design an easy to use system.	✓
Develop a system which matches requirements.	✓
Ensure system is well tested.	✓
Evaluate the user interface with help from caregivers.	✓

All of the objectives of this dissertation were met, meaning the key goal of this dissertation was met.

### 7.2 Requirements Evaluation

In this section, the system will be evaluated based on how many of the requirements were completed, ignoring the requirements in the “won’t do” category, as these are aimed more for future work and were not intended to be completed as part of this dissertation.

#### 7.2.1 Functional Requirement Evaluation

This section will focus on evaluating which functional requirements were achieved. It will also be used to explain why some requirements were not met. This will be shown in a table format.

**Table 6:** List of the functional requirements and if they have been achieved

<b>Requirement</b>	<b>Achieved?</b>	<b>Evidence</b>
<b>FR 1:</b> System must store information in a database.	✓	N/A
<b>FR 2:</b> System must allow users to log in.	✓	Figure: 5
<b>FR 2.1:</b> System must distinguish between caregiver and patient users.	✓	Figures: 6, 7, 17, 18
<b>FR 3:</b> System must have a calendar.	✓	Figures: 8
<b>FR 3.1:</b> All users must be able to view the calendar.	✓	Figures: 17, 18
<b>FR 3.2:</b> Caregiver users must be able to update the calendar.	✓	Figure: 20
<b>FR 4:</b> Caregiver users must be able to add new caregiver users to patient users.	X	N/A
<b>FR 5:</b> Patient users must be able to view current date and time.	✓	Figure: 10
<b>FR 6:</b> Patient users should be able to view current weather forecast.	✓	Figure: 9
<b>FR 7:</b> System should be able to contact caregiver users with updates to the calendar.	✓	Figure: 21
<b>FR 8:</b> System should be able to gather data from patient users' environment.	✓	Figures: 15, 16
<b>FR 8.1:</b> System should be able to send alerts to caregiver users about patient users' environment.	✓	Figures: 11, 23, 24, 25, 26, 27
<b>FR 9:</b> System should be able to send notifications to caregiver users when calendar is updated.	✓	Figure: 21
<b>FR 10:</b> System should allow for adding images to patient user's view.	✓	Figure: 13
<b>FR 10.1:</b> System should allow for changing the image.	✓	Figure: 13
<b>FR 10.2:</b> System should allow for adding captions to the images.	✓	Figure: 13
<b>FR 11:</b> System could send notifications to caregiver users should the patient's door be opened.	X	Figures: 23, 27
<b>FR 12:</b> System could allow for changing layout of patient users' view	✓	Figure: 14
<b>FR 17:</b> System should allow caregivers to modify the temperature at which they are alerted in regard to a patient's environment.	✓	Figure: 12
<b>FR 18:</b> System should show caregivers information about their patient.	✓	Figure: 19

FR 4 was not completed due to the way caregivers currently create their account. As they create their account, they automatically assign themselves a patient, by using the patient's ID. While this is not the best way of implementing this feature, it allows all of the other features to work.

FR 11 was not completed due to the complexity of this requirement. While initially it seemed plausible, in practice it would require a lot of cables to be flowing around patient's doors to communicate the opening and closing of the door to the hardware system. The alternative to not need cables would be to use expensive wireless hardware which I could not regrettably get with the current Covid-19 situation. The API functionality is there however, with a route created for the door being opened along with the HTML email which will be sent along with the text message. All that is missing for this requirement, is the hardware part of the requirement.

Overall, I am very pleased with the completed requirements. The system is a fully functional system which could be used by the public with little changes. Although there were two uncompleted requirements, one was a low priority "could have" requirements, there was also eighteen requirements completed, meaning 90% of requirements were completed. The completed requirements still highlight the huge volume of work undertaken in the development of this system.

### 7.2.2 Non-Functional Requirement Evaluation

This section will focus on evaluating which functional requirements were achieved. It will also be used to explain why some requirements were not met. This will be shown in a table format.

*Table 7: List of non-functional requirements and their achieved status*

Requirement	Achieved?
<b>NFR 1:</b> The system must be secure.	✓
<b>NFR 2:</b> The system must allow for all users to access simultaneously.	✓
<b>NFR 3:</b> The system must correctly handle multiple users updating same event simultaneously.	✓
<b>NFR 4:</b> The system must be fully tested.	✓
<b>NFR 5:</b> The system should be optimized to ensure requests are as quick as possible.	✓
<b>NFR 6:</b> The system should be portable, working on a range of systems.	✓

<b>NFR 7:</b> The system could be translated to multiple languages.	X
---	---

The non-functional requirements not completed was NFR 7. This is due to a late realization of the work that would be involved in this along with this. It would not only entail translating everything to other languages, it would also involve selecting and displaying the correct language for each user based on browser preferences.

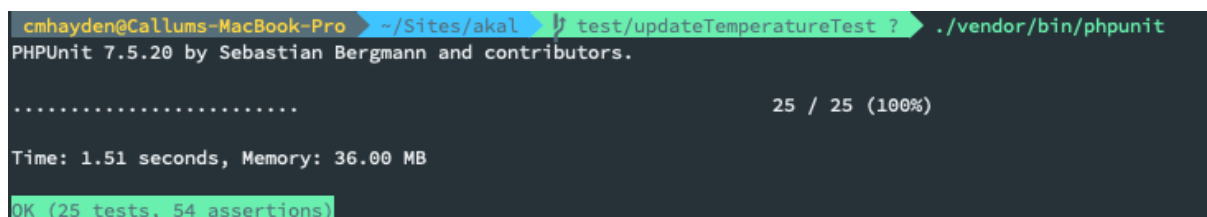
### 7.3 Testing Evaluation

In this section, the results of testing the system will be evaluated. This is to ensure the system is not only well tested, but the results of the tests can show the technical correctness of this dissertation. This section will be divided into three sub sections, unit testing, functional testing and security testing.

#### 7.3.1 Evaluation of Unit Tests

This section will show the output of running the unit tests, along with describing the rough coverage of these tests, and conclude with my opinions of the unit tests.

When the unit tests are run, the test suite goes through all the unit tests, checking all assumptions to ensure they are true. If they are not true, the test suite shows an error instead of “OK” with the number of tests and assertions made. This can be seen in the screenshot below.



```

cmhayden@Callums-MacBook-Pro ~/Sites/akal test/updateTemperatureTest ? ./vendor/bin/phpunit
PHPUnit 7.5.20 by Sebastian Bergmann and contributors.

..... 25 / 25 (100%)

Time: 1.51 seconds, Memory: 36.00 MB

OK (25 tests, 54 assertions)

```

**Figure 28:** Output of running PHPUnit tests

This screenshot not only shows the tests pass, but it also shows the time elapsed in running these tests. A time of 1.51 seconds is excellent and highlights the technical quality of the design of these unit tests. It also shows the memory allocated for use of the unit tests, which is the default 36.00 MB for PHPUnit.

As seen in the testing section (page 47-48), there are tests for every feature performed by the API including features which have not been fully implemented (FR 11 – alerting when a patient’s door

opens), not only testing that something that should work, works; but also testing that something that should not work, does not. This shows that these unit tests have full coverage of the API as everything is tested.

The unit tests are used solely to ensure the API is working as expected and is used to test all code that is committed to the projects repository due to the CI/CD system in place.

In conclusion, although the unit tests were a tedious part of development, it was a necessity for the CI/CD system to work. Based on the output from running the tests I am extremely pleased with the performance of the API and I am certain of its technical correctness.

### 7.3.2 Evaluation of Functional Tests

This section will discuss the results of the functional tests. It will also include a brief conclusion explaining how these results are used to evaluate whether the system's functionality works or if this part of the system has any issues.

The functional testing tested every input a user has with the system. Not only was valid data tested, but extreme data was also tested to ensure the user would understand why an error happened. This was done to make sure that when a user interacts with the system, be it in the intended way or not, the system would interact with the user by giving them feedback on their inputs.

As seen in the functional testing section (pages 49-50), all tests performed had the expected result, which is to give the user correct feedback on their actions. This shows the amount of thought put into the design of the system to guarantee the system would not break due to human error. While designing this meant a lot of testing and a lot of effort, the time employed in this is warranted by the results of the functional tests which highlight the quality of the system's design.

### 7.3.3 Evaluation of Security Tests

This section will discuss the results of the security tests. As a main objective of this project was to ensure user's data is safe, the security testing was an extremely important part of showing the achievement of this objective. The security vulnerabilities tested against were SQL injections and Cross Site Scripting (XSS) attacks.

When trying to get access to the data through SQL Injections, I was unable to. I tried multiple tutorials along with trying every input field available throughout the application, but the results were always the same, giving me no access to the data. This is also true for XSS attacks, which resulted in being unable to force the application to run JavaScript code through the use of input fields.

While I am not a fantastic hacker, I am confident with the security measures in place throughout this application, both implemented by myself such as the hashing of passwords and implemented by the framework such as automatic creation of CSRF tokens for each user. I am sure it is not possible to breach the system through SQL Injections, XSS attacks or CSRF attacks.

However, I understand that through brute force passwords could be obtained, but according to Kaspersky, if the user's password is not very complex, it could take more than two years to brute force (Kaspersky, 2020). The only solution to this would be to enforce users to use complex passwords including special characters instead of the current requirement of having to be at least six characters long.

## 7.4 Project Management Evaluation

In this section, the original project management will be revisited and evaluated based on how accurate the time estimations were, if the sprints were adhered to and if any unforeseen risks appeared. The original plan can be found in the appendix (5: Project Management).

As Winston Churchill once said, “those who plan do better than those who do not plan, even should they rarely stick to their plan”. This quote rings true in this project. While the planning stage was an essential part of the project, it was impossible to strictly stick to the plan. Some small adjustments were made throughout the development process due to both underestimating the amount of work required to develop this application along with overestimating my abilities to complete tasks in such a short manner.

There were also unforeseen risks which impacted my ability to strictly adhere to the original plan and original sprints proposed for this project. These unforeseen risks were the staff strikes and the Covid-19 pandemic, and lead to the following changes in the plan:



- **Staff strikes:** out of respect for my supervisor's right to strike, I decided not to request the biweekly meetings during the strikes.
- **Covid-19 pandemic:** as rumours circulated regarding an impending lockdown of the UK, I changed the time scale for performing evaluation. In order to do this, I left non-essential functionality for the evaluation until last. This includes the development of the patient's hardware system and the sending of text messages to users.

These risks were impossible to have seen coming and were dealt with in the best ways possible to ensure this dissertation would be completed on time without sacrificing on the overall quality of this dissertation.

Due to this I believe the project management for this dissertation was sufficient, it led to me thinking about how risks can affect a project, and how to mitigate them. It has helped me to understand both the importance of having a well-defined plan, and the importance of flexibility throughout a project to adapt to unconsidered risks.

## Chapter 8: Conclusion

This chapter will serve to gather my thoughts about this dissertation. It will express what I have achieved, what has limited my work, what I have learnt in the process and any future work that could be added to this dissertation.

This dissertation is exactly what I wanted to do at the current point in my life. I had an idea which not so long ago was just a dream. Over the course of the first semester with the help of a wide range of people such as caregivers, my supervisor, colleagues and the ethics coordination, this idea has been transformed into a plan, and from a plan into a fully functional application. While the amount of work involved in this dissertation was a lot, with long days of writing, late nights of research, sudden realisations while debugging and feeling that it will never be finished, it has benefitted me in a huge way. It has given me confidence in my ability as a programmer, it has helped me learn about the technical aspects of this dissertation as well as the overall domain of this dissertation, and it has allowed me to gain experience working on larger project, following a widely used programming process in TDD with a DevOps system in place to ensure this programming process is followed.

### 8.1 Achievements

This section will discuss what was achieved in this dissertation, highlighting not only what was achieved, but also how it was achieved.

The main achievement of this dissertation was being able to plan and develop a full system in just seven months. More so a system of this size which contains a web application, a fully functional API, and a hardware system which uses sensors to transform the project into a complex IoT project, something I had not even imagined prior to commencing this dissertation. To achieve this, a lot of work was involved. From late nights in the university labs doing research, to sitting on the train to London typing code. It had many ups and downs and at times felt unachievable, but in the end, it was achieved, to the best of my ability, and in what I believe to be a fantastic way.

This dissertation also achieved an above average system usability score when tested with its intended users, caregivers. While this is not an excellent score, it is a score I am proud of due to the work involved

in finding caregivers to test the system with, traveling to meet them for them to be able to fill it in, and even walking some participants through the study via Skype and having to wait anxiously for their filled in questionnaires to arrive in the post.

In regard to the functional requirements, this dissertation managed to achieve eighteen out of twenty functional requirements along with six out of seven non-functional requirements. This was a very large undertaking of work which I am pleased I did as the level of satisfaction felt now at the end of this dissertation is worth the work put in and more.

At the start of this dissertation an aim was given. The aim was to develop a cost-effective, hardware and software, tool that can be used to get information about a dementia or Alzheimer's patient's schedule and environment remotely, while attempting to help the patient through reorientation with prompts of the current date and time. At a cost of £34.00 for the Raspberry Pi, £4 for a temperature sensor, and £2 for cables used (PiHut, 2020), the aim of a cost-effective solution has been resoundingly met. With the use of sensors and a calendar giving caregivers access to their patient's information this has also met the aim of allowing caregivers to access information remotely. Finally, by showing the patient the date and time, their schedules and the current weather, this also helps with attempting to provide reorientation to the patient user, meaning this part of the aim has also been met. Overall the aims proposed seven months ago have all been met, which is a huge achievement for me.

## 8.2 Limitations

This section will discuss the limitations to this dissertation, along with what would have been done differently should these limitations not have existed.

The first limitation was the scope of this dissertation. Originally, the proposed project was an open hardware and software application. While this gave me the opportunity to set the scope, many of the ideas suggested during interviews were far outside the ideal scope I had set, which was to create an application which uses key concepts of the Internet of Things, which is a development of the internet in which many everyday objects are embedded with microchips giving them network connectivity (OED, 2001).

The second limitation was in the time scale of this project. The timescale from the conception of the idea to the final product was a little under a full academic year, during which time I also had other coursework, exams, and work to focus on as well as this dissertation. Due to this, the product had to be limited to something achievable within the time scale. If this was not the case, and I had longer to work on this project, I would have enjoyed expanding the scope of this project to include some of the features designated for future work. I would also have changed the way user evaluation was carried out, and instead would have given the system to multiple caregivers to use for a month or so and then get feedback from them on how useful it is and which features they feel would be beneficial to add to the system.

### 8.3 Future Work

This section will talk about other work that could be done in order to further improve this dissertation. I will also include explanations of why these ideas were not implemented.

While I am extremely pleased with the completed requirements, there is still much more that could be implemented in the future. For instance, many of the ideas given to me by caregivers were outside the scope of this dissertation. This includes ideas such as having the system be voice activated allowing the patients to talk to the system, and the system to the patients. Another idea was to allow for the system to order food for the patient, this however, could realistically be done by using online delivery services such as Just Eat for cooked food, or Tesco for everyday shopping and ingredients. Modifying the temperature of the patient's home is another amazing idea, but it would require either custom hardware, or the use of a system such as a smart thermostat and integrate it with the system. Video calling would have also been an interesting feature, but it would require the use of cameras and, in my opinion, cause distress to a patient as to why someone is on their TV, or why they can't call their family members, but further research would be beneficial before deciding if this is a good avenue of expansion for this project.

Some other sources of future work came as ideas while creating the application that I did not have the time to implement such as using a light sensor to turn the patient's system's screen on or off depending

on if the lights are on. Essentially, using the brightness of the room to determine if the patient is in the room or not. This would help to save electricity benefitting the patient financially while benefitting the environment as well.

## 8.4 Topics Learnt

This section will discuss what I learnt while completing this project. It will be separated into technical aspects and domain knowledge.

### 8.4.1 Technical

This section will discuss the main tools and technologies mastered in order to complete this dissertation. It will focus on showing the difference in what I knew before commencing this project compared to now.

**Laravel:** while I had previous experience using the Laravel framework, I had only used it to create simple micro-services. In order to complete this dissertation, the key part of Laravel I had to master was the use of models. The use of models and factories can be seen throughout the web application and allows for interaction with the database without writing SQL queries which could potentially cause security vulnerabilities.

**Vue:** prior to developing this system I had never understood how to use Vue and primarily only used premade Vue components. While developing this system I learnt about how data is stored within Vue, how to properly create custom components, and how to display data to the component's UI.

**Raspberry Pi:** prior to this project, I had only used the Raspberry Pi for one part of university coursework and never truly understood how to program it. In order to create this dissertation, I had to learn how to add sensors to the Raspberry Pi, how to get data from these sensors, and how to send the data to an API.

**Server setup & DNS:** setting up a server is something I had never done before prior to this project. I had configured some DNS settings to point a domain to an IP address, but never truly understood what DNS does nor what other types of records do. In order to develop this dissertation, I had to learn about

what different DNS records were for, along with having to learn how to create a server with AWS, Laravel Forge, and NGINX.

**Security:** before commencing this dissertation, I understood what security vulnerabilities were and how to prevent them, however, I had never tried to test the security of an application. In order to test this, I learnt about how SQL Injections and XSS attacks are performed and how I can test against them.

**HTML Emails:** before commencing this dissertation, I had never created a HTML email before. This was something I wanted to learn how to do, and although making the emails be structured with HTML was not a requirement, it adds a feel of professionalism. In order to make these emails, I had to learn how to send HTML emails, and how to write them.

#### 8.4.2 Domain Knowledge

This section will cover what was learnt about the domain of this dissertation. The domain of this dissertation is the work done by caregivers and the impacts of Dementia and Alzheimer's.

**Caregivers:** prior to this dissertation I knew a lot about caregivers and the jobs they carry out, due to my mother's job. While carrying out research however, I gained a new appreciation for everything they do for their patients and the amount they truly care for their patients.

**Dementia & Alzheimer's:** prior to this dissertation, I had very basic knowledge of Alzheimer's and Dementia due to family members being inflicted with these illnesses. While researching for this dissertation, I developed an understanding of why different issues, such as their homes always being too warm, plague the elderly. I also developed a new appreciation for the difficulty's dementia and Alzheimer's patients feel during the early stages such as confusion, loss of balance, and forgetfulness in regard to basic tasks I personally didn't give a second thought about such as getting out of bed or getting dressed.

## Bibliography

(2020). Retrieved April 16, 2020, from PiHut: <https://thepihut.com/>

Alzheimer's Association. (2019). *Daily Care Plan*. Retrieved November 7, 2019, from Alzheimers.org: <https://www.alz.org/help-support/caregiving/daily-care/daily-care-plan>

Alzheimer's Association. (2019). *Dementia - Symptoms, Causes, Diagnosis, Treatments*. Retrieved November 18, 2019, from Alzheimer's Org: <https://www.alz.org/alzheimers-dementia/what-is-dementia>

Alzheimer's Society. (2010, March). *Simple ways to help someone living with dementia*. Retrieved October 2, 2019, from Whittington NHS: <https://www.whittington.nhs.uk/document.ashx?id=2103>

Bangor, A., Kortum, P., & Miller, J. (2008). An Empirical Evaluation of the System Usability Scale. *International Journal of Human-Computer Interaction*, 574-594.  
doi:10.1080/10447310802205776

Chandrashekhar, R., Mardithaya, M., & Santhi Thilagam, D. S. (2012). SQL Injection Attack Mechanisms and Prevention Techniques. *International Conference on Advanced Computing, Networking and Security*. 7135, pp. 524-533. Springer, Berlin, Heidelberg.

Cheon, Y., & Leavens, G. T. (2002). A Simple and Practical Approach to Unit Testing: The JML and JUnit Way. *ECOOP 2002 - Object-Oriented Programming*, 2374, pp. 231-255. Springer, Berlin, Heidelberg.

dakboard. (2019). *DAKboard - A customizable display for your photos, calendar, news...* Retrieved September 22, 2019, from DAKboard: <https://www.dakboard.com/site>

DeFranzo, S. E. (2011, September 16). *Difference between qualitative and quantitative research*. Retrieved November 19, 2019, from Snap surveys: <https://www.snapsurveys.com/blog/qualitative-vs-quantitative-research/>

- Duere, K. (2016, September 25). *10 daily apps to help caregivers*. Retrieved March 2020, from Mashable: <https://mashable.com/2016/09/25/apps-for-caregiving-caregivers/?europa=true>
- Dujmović, J. (1996). A Method for Evaluation and election of Complex Hardware and Software Systems. *Computer Measurement Group 96 Proceedings: Managing the Evolution* (pp. 368-378). San Diego: Computer Measurement Group.
- Five second tests. (2020). *Five second test*. Retrieved April 20, 2020, from Five second test: <https://fivesecondtest.com>
- Hawkey, K., Inkpen, K., Rockwood, K., Mcallister, M., & Slonim, J. (2005). Requirements Gathering with Alzheimer's Patients and Caregivers. *Proceedings of the 7th International ACM SIGACCESS Conference on Computers and Accesibility*, (pp. 142-149). Retrieved October 22, 2019
- Home Support Services | North Lanarkshire Council*. (2017, March 22). Retrieved March 2020, from North Lanarkshire Council: <https://www.northlanarkshire.gov.uk/index.aspx?articleid=13198>
- Jordan, P., Thomas, B., Weerdmeester, B., & McClelland, I. (1996). *Uability evaluation in industry*. London: Taylor & Francis.
- Kılıçdağı, A., & Yilmaz, H. (2014). *Laravel Design Patterns and Best Practices*. Birmingham: Packt.
- Kaspersky. (2020). *What's a brute force attack?* Retrieved April 20, 2020, from kasperky: <https://www.kaspersky.co.uk/resource-center/definitions/brute-force-attack>
- Kumar, P. P. (2005, July). Effective Use of Gantt Chart for Managing Large Scale Projects. *Cost Engineering*, 47(7), 14-21.
- Laravel. (n.d.). *Database: Query Builder*. Retrieved March 2020, from Laravel: <https://laravel.com/docs/5.6/queries>



- Maksimovic, M., Vujovic, V., Davidović, N., Milosevic, V., & Perisic, B. (2014). Raspberry Pi as Internet of Things hardware: Performances and Constraints . *1st International Conference on Electrical, Electronic and Computing Engineering IcETRAN 2014*. Vrnjačka Banja.
- Maximilien, E., & Williams, L. (2003). Assessing Test-Driven Development at IBM. *Proceedings of the 25th International Conference on Software Engineering* (pp. 564-569). Portland: IEEE Computer Society.
- McCool, S. (2012). *Laravel Starter. The definitive introduction to the Laravel PHP web development framework*. Birmingham: Packt.
- McGovern, G. (2011, November 6). *The vital importance of the first click*. Retrieved April 20, 2020, from Gerry McGovern: <https://gerrymcgovern.com/the-vital-importance-of-the-first-click/>
- MDN contributors. (2020, February 24). *Cross-Origin Resource Sharing*. Retrieved April 12, 2020, from MDN Web Docs: <https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>
- Murray, James; Minor, William Chester. (2019). *Dementia noun - Definition, pictures, pronunciation and usage*. Retrieved October 20, 2019, from Oxford Learner's Dictionaries.
- NGINX. (n.d.). *What is DevOps*. Retrieved April 12, 2020, from NGINX: <https://www.nginx.com/resources/glossary/devops/>
- Nginx. (n.d.). *What is NGINX?* Retrieved April 12, 2020, from NGINX: <https://www.nginx.com/resources/glossary/nginx/>
- OED. (2001). *Oxford English Dictionary* (3rd edition ed.). Oxford: Oxford University Press.
- OWSAP. (n.d.). *Cross Site Scripting (XSS)*. Retrieved April 2020, from Owsap: <https://owasp.org/www-community/attacks/xss/>
- Pearce, M. (2016, December 15). *What are Evaluation Methods?* Retrieved November 19, 2019, from Funding for good: <https://fundingforgood.org/what-are-evaluation-methods/>

Reviews. (2017). *Amazon reviews of DAKboard wall display v2*. Retrieved November 18, 2019, from Amazon: <https://www.amazon.com/DAKboard-inch-Wall-Display-Do/product-reviews/B073RRDR2D/>

Reviews. (2017, October 9). *dakboard reviews*. Retrieved November 18, 2019, from Facebook: <https://www.facebook.com/pg/dakboard/reviews/>

Runeson, P. (2006, July-Aug). A survey of unit testing practices. *IEEE Software*, 23(4), 22-29.

Scallan, T. L. (2017, September 21). *Why are seniors always so cold?* Retrieved October 2, 2019, from Today's Care Giver: <https://caregiver.com/articles/why-seniors-cold/>

Software Testing Help. (2020, March 11). *Functional testing: a complete guide*. Retrieved April 2, 2020, from software testing help: <https://www.softwaretestinghelp.com/guide-to-functional-testing/>

University Research Ethics Committee. (2017, March 29). University Research Ethics Policy.

Usability Hub. (2020). *An introduction to First Click Testing*. Retrieved April 20, 2020, from Usability Hub: <https://usabilityhub.com/guides/first-click-testing>

van Loenhout, J., le Grand, A., Duijum, F., Greven, F., Vink, N., Hoek, G., & Zuubier, M. (2016, October 9). The effect of high indoor temperatures on self-perceived health of elderly persons. (J. L. Domingo, R. Letcher, & A. Wang, Eds.) *Environmental Research*, 146, 27-34.

Vue.js. (2019). *Introduction - Vue.js*. Retrieved November 18, 2019, from Vue.js: <https://vuejs.org/v2/guide/>

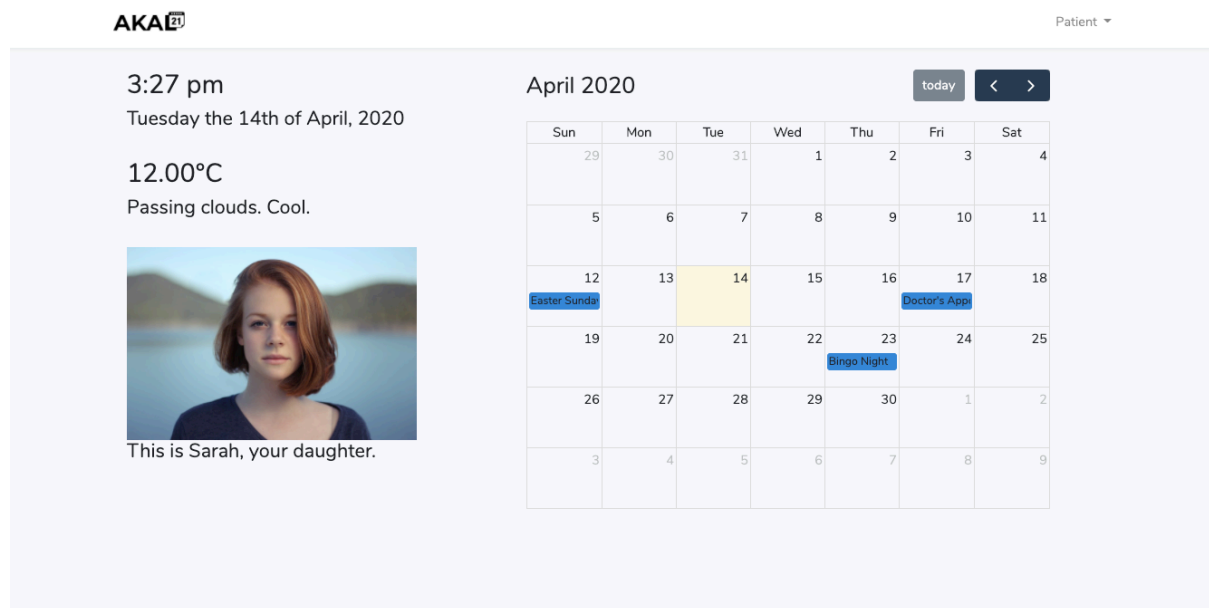
World Health Organisation. (2019). *Statistics About Dementia*. Retrieved November 16, 2019, from Dementia Statistics Hub: <https://www.dementiastatistics.org/statistics-about-dementia/>

Yahnke, K. (2019). *How to use a risk assessment matrix*. Retrieved November 2, 2019, from i-sight: <https://i-sight.com/resources/risk-assessment-matrix/>

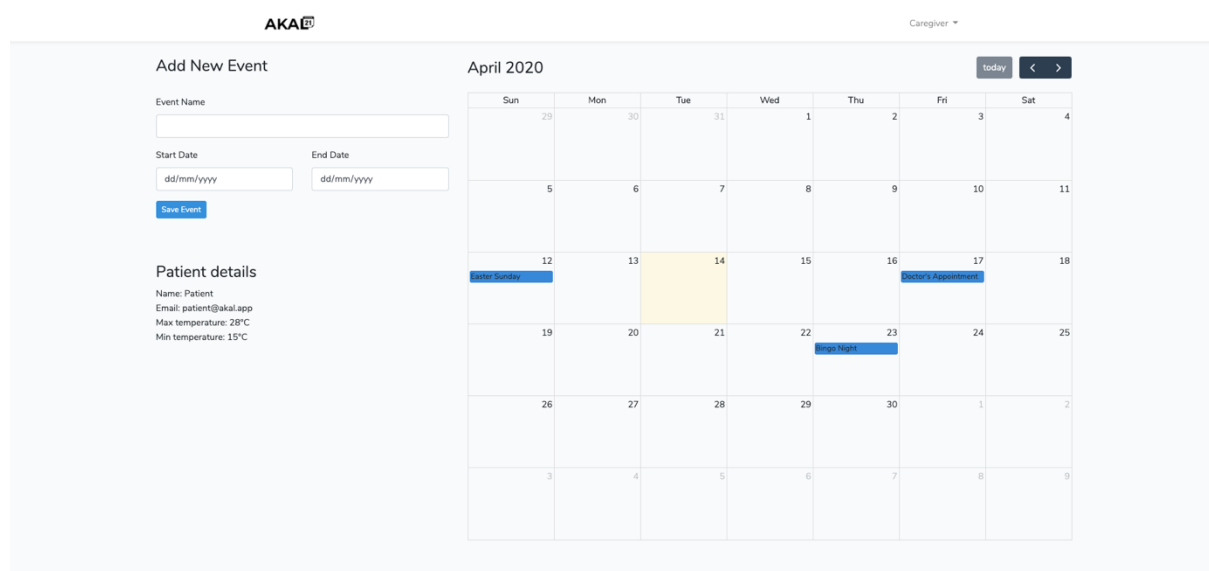
# Appendix

## 1: Images of System

This section of the Appendix includes images of the system. Showcasing the overall system as well as any screenshots taken as part of testing.



**Figure 30:** Screenshot of home page as patient user



**Figure 29:** Screenshot of home page as carer user

## Patient details

Name: Patient

Email: patient@akal.app

Max temperature: 28°C

Min temperature: 15°C

**Figure 31:** Screenshot of patient details feature

### Edit or Delete

Event Name

Start Date

End Date

### Patient details

Name: Patient

Email: patient@akal.app

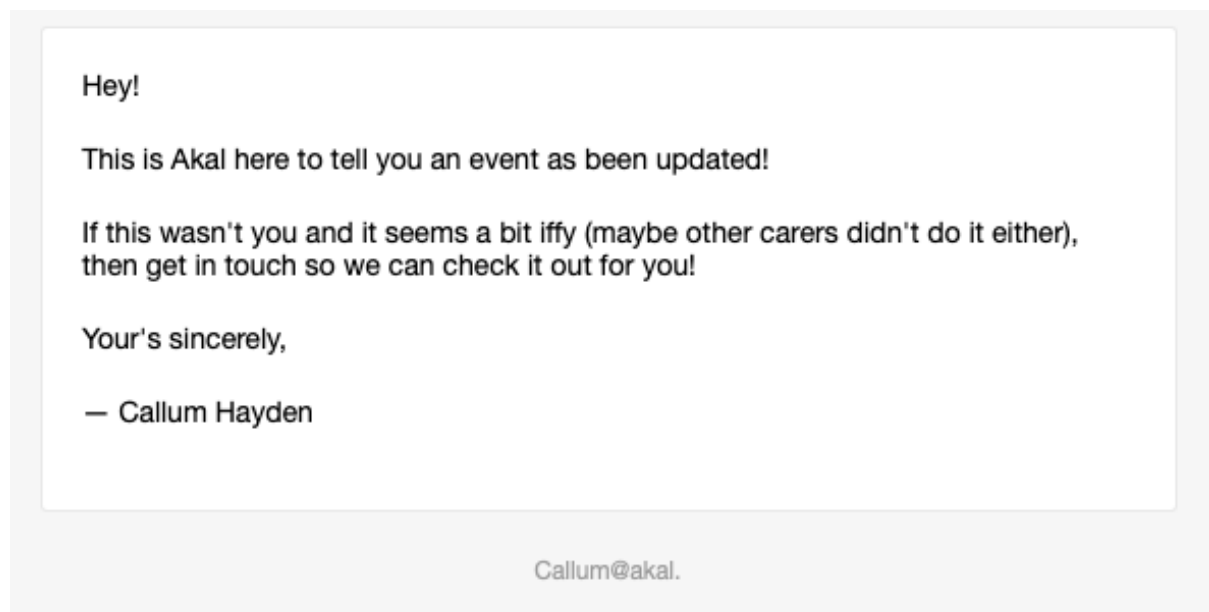
Max temperature: 27°C

Min temperature: 16°C

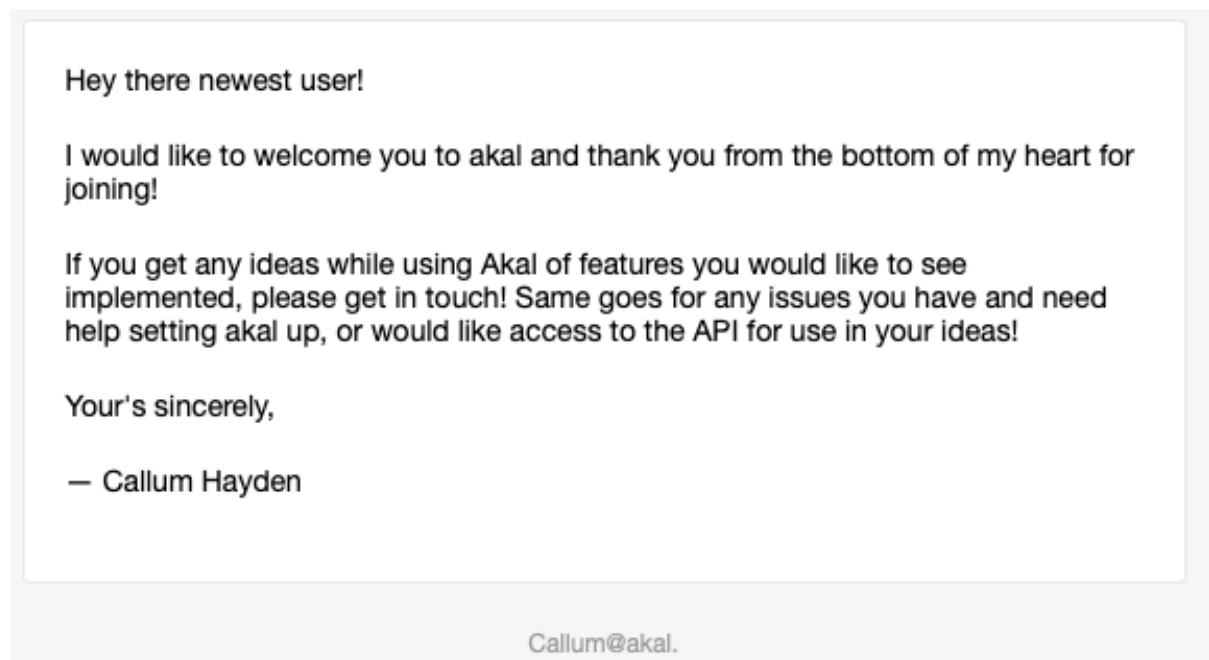
### April 2020

Sun	
29	
5	
12	Easter Sunday
19	

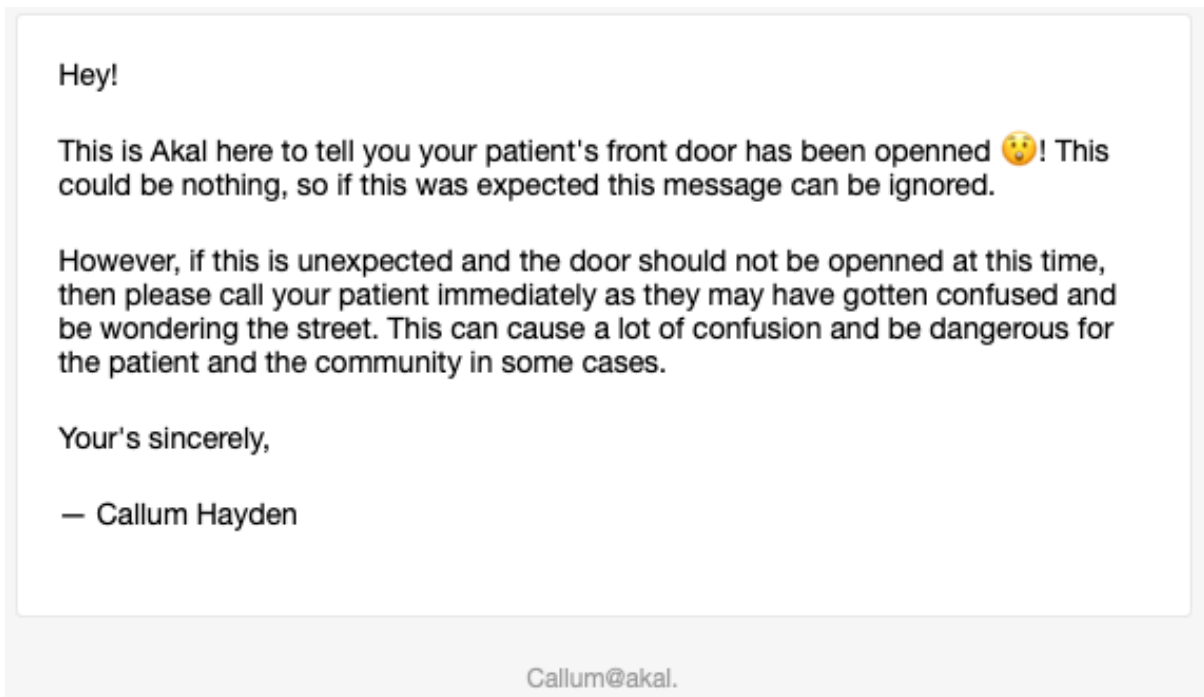
**Figure 32:** Screenshot of updating or deleting an event



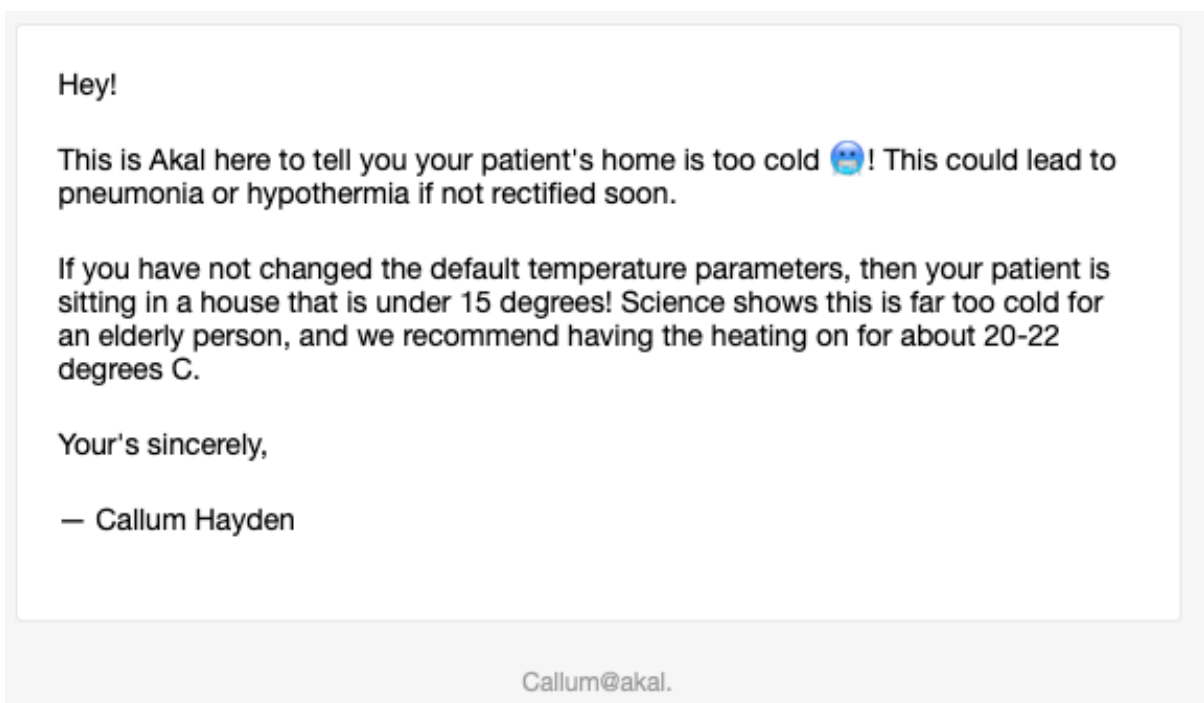
*Figure 33: Screenshot of HTML email for when an event is updated*



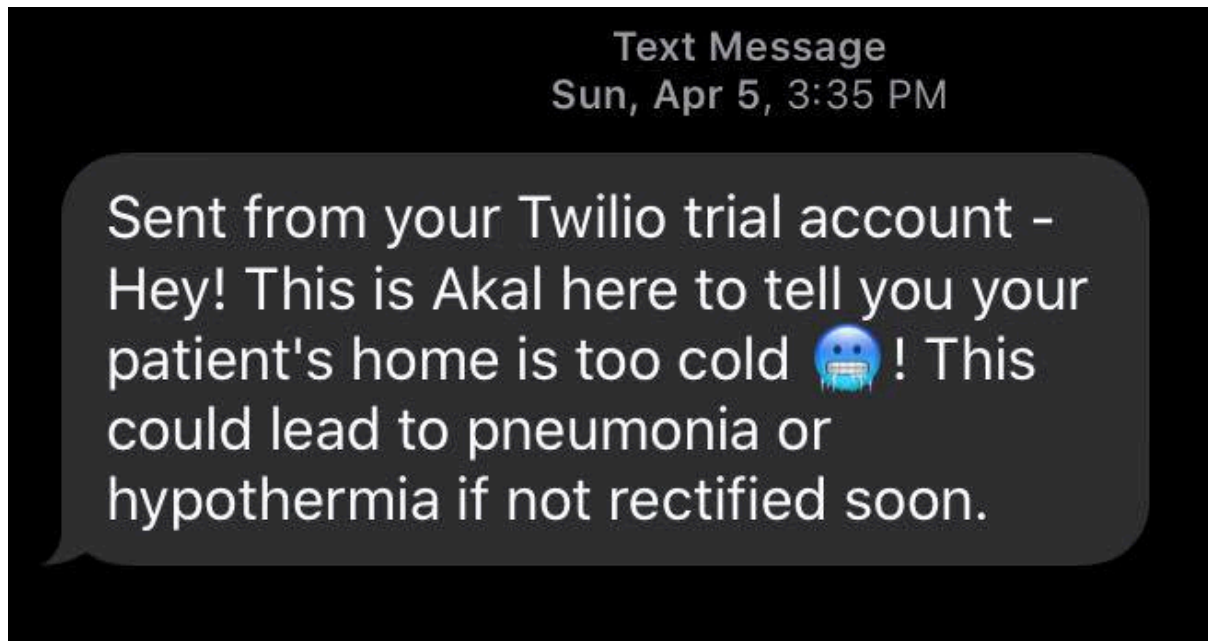
*Figure 34: Screenshot of HTML welcome email sent when a user registers*



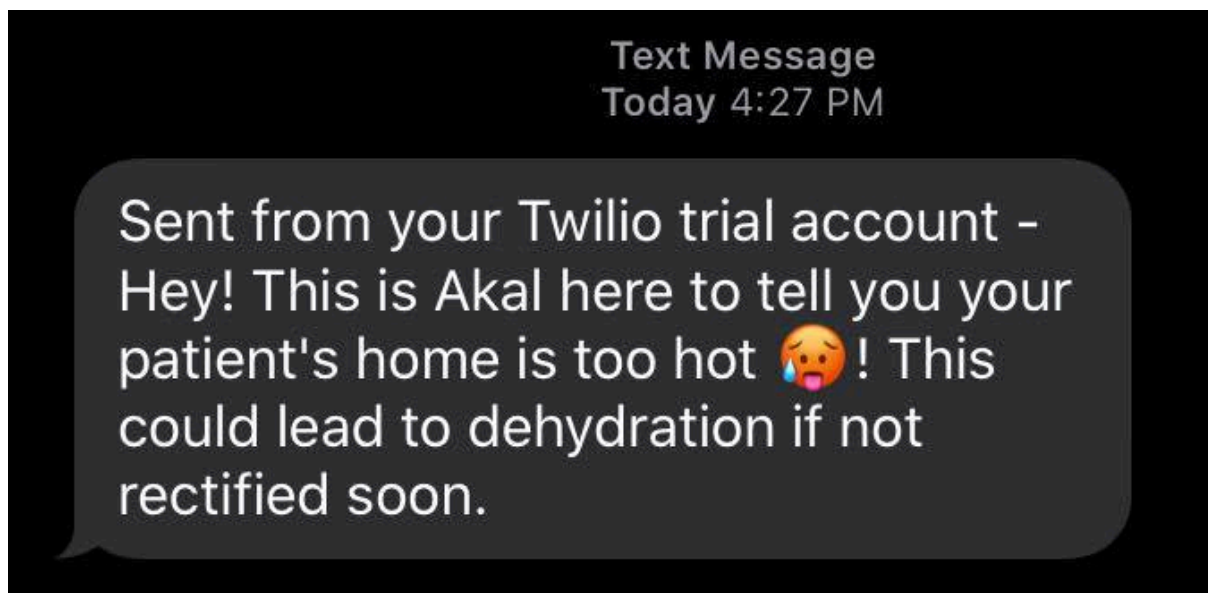
**Figure 35:** Screenshot of HTML email sent to alert user of door being opened



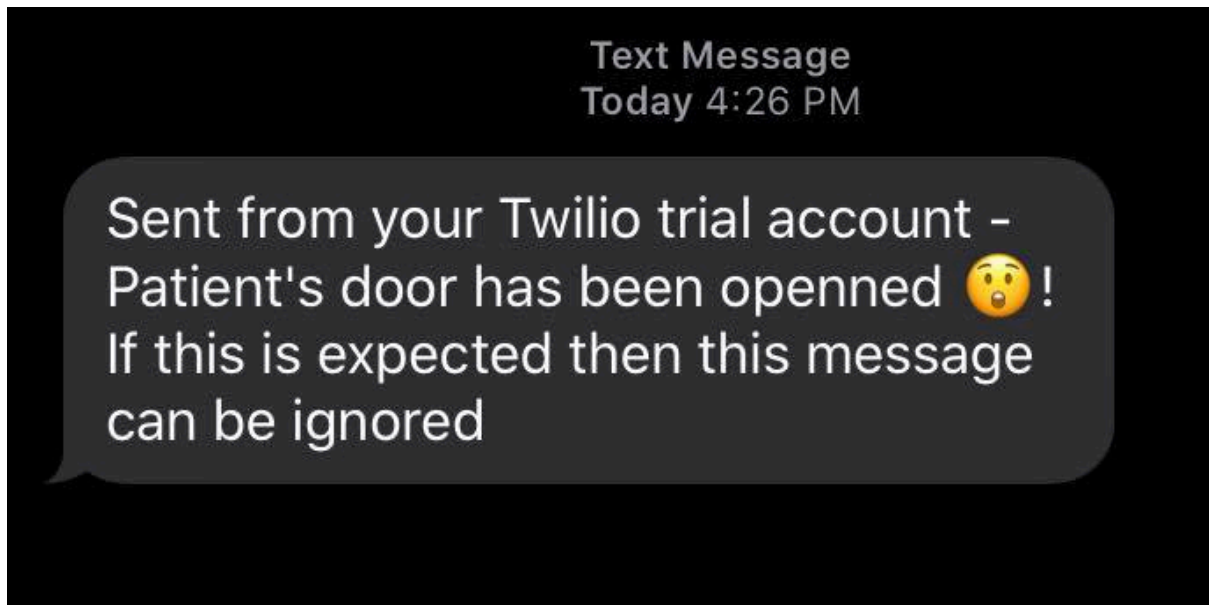
**Figure 36:** Screenshot of HTML email sent if temperature is too cold



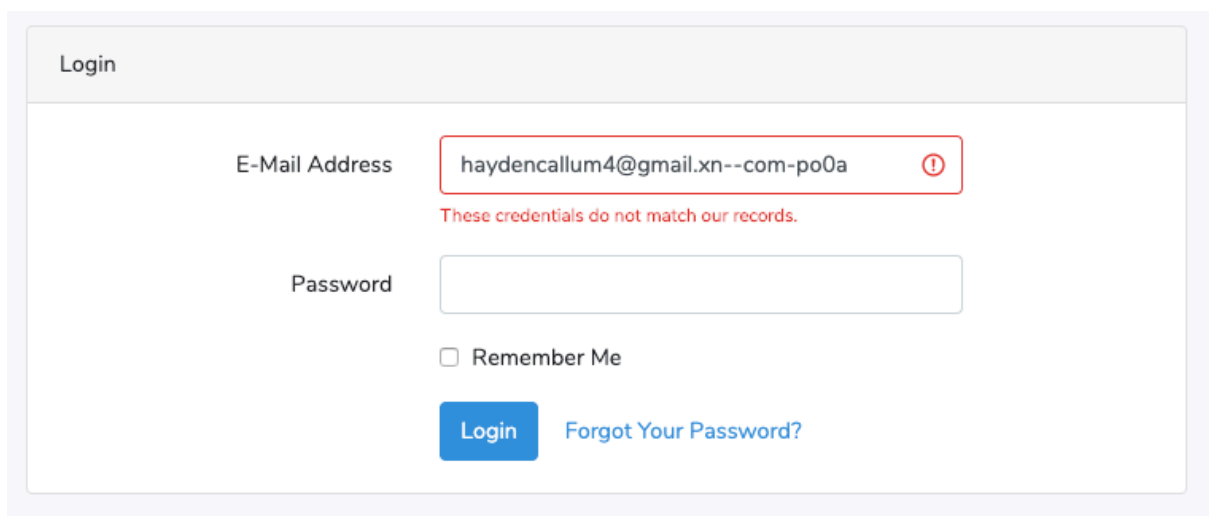
**Figure 37:** Screenshot of SMS sent if temperature is too cold



**Figure 38:** Screenshot of SMS sent if temperature is too hot



**Figure 39:** Screenshot of SMS sent if door has been opened



**Figure 40:** Screenshot of an SQL injection attempt with ' in email



Login

E-Mail Address haydencallum4@gmail.com"

Password .....

☐ Remember Me

Login [Forgot Your Password?](#)

**Figure 41:** Screenshot of an SQL injection attempt with " in email

Login

E-Mail Address lert("XSS")</script>haydencallum4@gmail.com


Password .....

☐ Remember Me

Login [Forgot Your Password?](#)

**Figure 42:** Screenshot of an XSS attack attempt with JS in email field

Login

E-Mail Address  

These credentials do not match our records.

Password

☐ Remember Me

[Forgot Your Password?](#)

**Figure 43:** Screenshot of an XSS Attack attempt with JS in the password field

## 2: Code Snippets

### 2.1 User Model

Code taken from the user model.

```
class User extends Authenticatable
{
    use HasApiTokens, Notifiable;

    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'name', 'email', 'password', 'userType', 'patientEmail', 'phoneNumber'
    ];

    /**
     * The attributes that should be hidden for arrays.
     *
     * @var array
     */
    protected $hidden = [
        'password', 'remember_token',
    ];

    /**
     * The attributes that should be cast to native types.
     *
     * @var array
     */
    protected $casts = [
        'email_verified_at' => 'datetime',
    ];
}
```

## 2.2 Login Controller

Code taken from the login controller.

```
class LoginController extends Controller
{
    /**
     |-----
     | Login Controller
     |-----
     |
     | This controller handles authenticating users for the application and
     | redirecting them to your home screen. The controller uses a trait
     | to conveniently provide its functionality to your applications.
     |
     */

    use AuthenticatesUsers;

    /**
     * Where to redirect users after login.
     *
     * @var string
     */
    protected $redirectTo = '/home';

    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('guest')->except('logout');
    }
}
```

## 2.3 Calendar Vue Component

Code snippet showing the data, created function, the methods and the watch function of the calendar component.

```
data() {
    return {
        calendarPlugins: [dayGridPlugin, interactionPlugin],
        events: "",
        newEvent: {
            event_name: "",
```

```

        start_date: "",
        end_date: ""
    },
    addingMode: true,
    indexToUpdate: "",
    patient_name: null,
    patient_email: null,
    min_temp: null,
    max_temp: null
  };
},
created() {
  this.getEvents();
  this.getPatientDetails();
},
methods: {
  addNewEvent() {
    axios
      .post("/api/calendar", {
        ...this.newEvent
      })
      .then(data => {
        this.getEvents(); // update our list of events
        this.resetForm(); // clear newEvent properties (e.g. title, start_date
and end_date)
      })
      .catch(err => {
        console.log("Unable to add new event!", err.response.data)
      });
  },
  showEvent(arg) {
    this.addingMode = false;
    const { id, title, start, end } = this.events.find(
      event => event.id === +arg.event.id
    );
    this.indexToUpdate = id;
    this.newEvent = {
      event_name: title,
      start_date: start,
      end_date: end
    };
  },
  updateEvent() {
    axios
      .put("/api/calendar/" + this.indexToUpdate, {
        ...this.newEvent
      })
      .then(resp => {
        this.resetForm();
        this.getEvents();
        this.addingMode = !this.addingMode;
      });
  },

```

```

    })
    .catch(err =>
      console.log("Unable to update event!", err.response.data)
    );
  },
  deleteEvent() {
    axios
      .delete("/api/calendar/" + this.indexToUpdate)
      .then(resp => {
        this.resetForm();
        this.getEvents();
        this.addingMode = !this.addingMode;
      })
      .catch(err =>
        console.log("Unable to delete event!", err.response.data)
      );
  },
  getEvents() {
    axios
      .get("/api/calendar")
      .then(resp => (this.events = resp.data.data))
      .catch(err => console.log(err.response.data));
  },
  resetForm() {
    Object.keys(this.newEvent).forEach(key => {
      return (this.newEvent[key] = "");
    });
  },
  getPatientDetails() {
    axios
      .get("/api/patientdetails")
      .then(resp => (
        this.patient_email = resp.data[0].email,
        this.patient_name = resp.data[0].name
      ))
      .catch(err => console.warn(err.response.data));
    axios
      .get("/api/temperature")
      .then(resp => (
        this.min_temp = resp.data.data[0].minTemp,
        this.max_temp = resp.data.data[0].maxTemp,
        console.log(resp.data.data[0])
      ))
  }
},
watch: {
  indexToUpdate() {
    return this.indexToUpdate;
  }
}
}

```

## 2.4 Calendar Controller

This is a snippet from the calendar controller which shows the rest of the functions it has.

```
class CalendarController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        if (Auth::check())
        {
            $userType = Auth::user()->userType;

            if($userType == "carer")
            {
                $patientEmail = auth::user()->patientEmail;

                return
CalendarResource::collection(Calendar::where('patient_email',"{$patientEmail}")-
>get());
            }

            else if($userType == "patient")
            {
                $email = auth::user()->email;

                return
CalendarResource::collection(Calendar::where('patient_email', "{$email}")->get());
            }
        }
    }

    /**
     * Display the specified resource.
     *
     * @param \App\Calendar $calendar
     * @return \Illuminate\Http\Response
     */
    public function show(Calendar $calendar)
    {
        return response($calendar, Response::HTTP_OK);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     */
}
```

```

    * @param \App\Calendar $calendar
    * @return \Illuminate\Http\Response
    */
    public function update(Request $request, Calendar $calendar)
    {
        $patientEmail = auth::user()->patientEmail;

        $users =
UserResource::collection(User::where('patientEmail',"$patientEmail")->get());

        foreach ($users as $user)
        {
            $data = ['name' => $user->name, 'email' => $user->email, 'subject' =>
"Event updated!"];

            try
            {
                Mail::send('emails.eventAlert', $data, function($message) use
($data)
                {
                    $message->to($data['email'], $data['name'])
                        ->subject($data['subject']);
                });
            }
            catch (Exception $e)
            {
                throw new Exception($e);
            }
        }

        $calendar->update(array_merge($request->all(),
['patient_email'=>"$patientEmail"]));
        return response()->json([
            'data' => new CalendarResource($calendar),
            'message' => 'Event updated',
            'status' => Response::HTTP_ACCEPTED
        ]);
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param \App\Calendar $calendar
     * @return \Illuminate\Http\Response
     */
    public function destroy(Calendar $calendar)
    {
        $calendar->delete();
        return response('Event removed.', Response::HTTP_NO_CONTENT);
    }
}

```



## 2.5 Notifications Controller

```
class NotificationController extends Controller
{
    public function alertTemperature($temp)
    {
        if (Auth::check())
        {
            $userType = Auth::user()->userType;

            if($userType == "carer")
            {
                return response()->json([
                    'message' => 'This route is only for patient users.',
                    'status' => 403
                ]);
            }

            else if($userType == "patient")
            {
                $patientEmail = auth::user()->email;

                $users =
                UserResource::collection(User::where('patientEmail',"{$patientEmail}")->get());

                $temperatures =
                TemperatureResource::collection(Temperature::where('patientEmail',"{$patientEmail}")->get());

                if($temperatures[0]->minTemp > $temp){
                    $message = "Hey! This is Akal here to tell you your patient's
home is too cold 🥶! This could lead to pneumonia or hypothermia if not rectified
soon.";

                    $subject = "Patient is too cold 🥶";

                    foreach ($users as $user)
                    {
                        $data = ['name' => $user->name, 'email' => $user->email,
'subject' => $subject];

                        $this->sendTextMessage($message, $user->phoneNumber);
                        $this->sendEmail('emails.coldAlert', $data);
                    }
                } elseif($temperatures[0]->maxTemp < $temp){

                    $message = "Hey! This is Akal here to tell you your patient's
home is too hot 🥵! This could lead to dehydration if not rectified soon.";
                    $subject = "Patient is too hot 🥵";
                }
            }
        }
    }
}
```

```

        foreach ($users as $user)
        {
            $data = ['name' => $user->name, 'email' => $user->email,
'subject' => $subject];

            $this->sendTextMessage($message, $user->phoneNumber);
            $this->sendEmail('emails.hotAlert', $data);
        }

    } else {
        return response()->json([
            'message' => 'Good Temperature 😊',
            'status'=> 200
        ]);
    }

    return response()->json([
        'message' => 'Alerts sent correctly!',
        'status'=> 200
    ]);
}

return response()->json([
    'message' => 'Check sentry ASAP',
    'status' => 500
]);
}

public function alertOpenDoor()
{
    $patientEmail = auth::user()->email;

    $message = "Patient's door has been opened 😱! If this is expected then
this message can be ignored";

    $subject = "Patient's door has been opened 😱!";

    $users =
UserResource::collection(User::where('patientEmail',"{$patientEmail}")->get());

    foreach ($users as $user)
    {
        $data = ['name' => $user->name, 'email' => $user->email, 'subject' =>
$subject];

        $this->sendTextMessage($message, $user->phoneNumber);
        $this->sendEmail('emails.doorAlert', $data);
    }
}

```

```

private function sendTextMessage($message, $recipients)
{
    $account_sid = getenv("TWILIO_SID");
    $auth_token = getenv("TWILIO_AUTH_TOKEN");
    $twilio_number = getenv("TWILIO_NUMBER");
    $client = new Client($account_sid, $auth_token);
    $client->messages->create($recipients, ['from' => $twilio_number, 'body' =>
$message]);
}

public function sendEmail($view, $data)
{
    try
    {
        Mail::send($view, $data, function($message) use ($data)
        {
            $message->to($data['email'], $data['name'])
                ->subject($data['subject']);
        });
    }
    catch (Exception $e)
    {
        throw new Exception($e);
    }
}
}

```

## 3: User Evaluation

### 3.1 Original User Evaluation Plan

To evaluate this project, the system usability scale will be used. This is due to SUS being reliable when used on a small sample size, ensuring that should finding participants prove to be difficult, it will still work. The downfall however is the complexity of the scoring system, and the temptation to look at the scores and interpret them as percentages, as the scores are on a scale of zero to one hundred. Participants of the study will be given tasks to perform with the system, and then given the following ten statements to rate from zero to five based on their level of agreement:

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

Participants will also be asked for their opinion on the system, what they liked, and what they disliked. This will allow for gathering qualitative data about the system and find which parts of the system need improved. The system will be considered successful should the SUS score above 68.

### 3.2 User Evaluation Handouts

Handouts given to participants of the user evaluation.

### 3.2.1 Consent Form

#### Akal – Alzheimer’s Calendar Consent Form

##### Participation No:

Please take time to read through this form; it will provide information about why this research is taking place, what will be involved, and what data needs to be collected. If you have any questions, please ask the facilitator.

##### Purpose of research

The purpose of this research is to gather opinions and attitudes on Akal, a calendar system aimed towards helping caregivers with Alzheimer’s and dementia patients. Due to this, the research looks particularly at caregivers of any age and caregivers in any capacity. The results of this research will help modify the way Akal is used in future iterations.

##### Your participation

In this usability test:

- You will be asked to perform certain tasks on an electronic device using Akal.
- You will be asked to fill in a short questionnaire regarding the tasks you performed.

##### Anonymity and confidentiality

All results will be held in strict confidence, ensuring privacy of all participants. There will be no record kept that would allow your results to be tracked back to you. All data will also be held securely in a password protected computer system or a locked filing cabinet.

##### Right to withdraw

You may withdraw from the experiment at any point without prejudice, and all data recorded will be deleted. If you wish to withdraw please inform the facilitator as soon as possible.

##### Right to information

A feedback sheet containing summaries of the data from this study will be sent to all participants who request it, after the data has been analysed. If you wish to be sent the results, please opt in below.

<b>Name:</b>
<b>Signed:</b>
<b>Date:</b>
<b>Email Address:</b>

I would like to receive information on the results of this study: Yes ☐ No ☐

### 3.2.2 Questionnaire

#### Akal – Alzheimer’s Calendar Questionnaire

## Participation No:

### Tasks

Please take time to attempt the following list of tasks. If you are unable to complete a task, this does not reflect on you, more so, it reflects on the lack of usability of the application.

- 1 Register a new account carer account. (A patient email of [patient@akal.app](mailto:patient@akal.app) may be used).
- 2 Logout of your account.
- 3 Sign back into your account.
- 4 Add a new event to the calendar.
- 5 Update the event.
- 6 Delete the event.
- 7 Change the patient's displayed image.
- 8 Change the patient's maximum and minimum temperatures.
- 9 Change the patient's layout.

### Demographic questions

Please circle the response which best represents your answer to the following questions:

- 1 Do you work in caregiving?  
Yes                  No
- 2 How frequently do you use caregiving applications?  
Never              Occasionally      Daily
- 3 What would your level of IT skills be?  
Beginner      Intermediate      Advanced

### Questionnaire

Carefully read the following statements and circle the response which describes your level of agreement with the statements provided.

- 1 I think I would like to use this system frequently.  
Strongly Disagree      Disagree                  Undecided                  Agree                  Strongly Agree
- 2 I found this system unnecessarily complex.  
Strongly Disagree      Disagree                  Undecided                  Agree                  Strongly Agree
- 3 I thought the system was easy to use.  
Strongly Disagree      Disagree                  Undecided                  Agree                  Strongly Agree
- 4 I think that I would need the support of a technical person to be able to use this system.  
Strongly Disagree      Disagree                  Undecided                  Agree                  Strongly Agree

- |    |   |                   |          |           |       |                |
|----|---|-------------------|----------|-----------|-------|----------------|
| 5  | I found the various functions in this system were well integrated.            | Strongly Disagree | Disagree | Undecided | Agree | Strongly Agree |
| 6  | I thought there was too much inconsistency in this system.                    | Strongly Disagree | Disagree | Undecided | Agree | Strongly Agree |
| 7  | I would imagine that most people would learn to use this system very quickly. | Strongly Disagree | Disagree | Undecided | Agree | Strongly Agree |
| 8  | I found the system very cumbersome to use.                                    | Strongly Disagree | Disagree | Undecided | Agree | Strongly Agree |
| 9  | I felt very confident using the system.                                       | Strongly Disagree | Disagree | Undecided | Agree | Strongly Agree |
| 10 | I needed to learn a lot of things before I could get going with this system.  | Strongly Disagree | Disagree | Undecided | Agree | Strongly Agree |

If you would like to provide further feedback about the system, please feel free to add comments below:

---

---

---

---

---

---

---

---

---

---

---

---

## 4: Considerations

This chapter will reflect on considerations made with this dissertation. The topics of the considerations will be professional considerations, such as codes of conduct; legal considerations, such as the data protection act; Ethical considerations, such as testing on Alzheimer's patients; and social considerations, such as how impactful this dissertation could be for society

### 4.1 Professional Considerations

This section is about considerations made as an IT professional. The main thing in this section is the British Computing Societies (BCS) code of conduct. As a student member of the BCS, I have the duty to adhere to their code of conduct and have done so in every step of this dissertation.

Some of the main professional considerations kept in mind were to ensure I only included requirements that were within the scope of my professional competence. Due to this, some requirements that would have been fantastic additions to the system are labelled as "won't do" requirements. Another consideration made was ensuring I had the right to use certain third-party tools and APIs I intended to use. Some of these were circumvented through buying a license for the product, or paying to use the service, such as Laravel Forge and Amazon Web Services. All tools, APIs, libraries, and framework's used are either open source, allow for usage within the scope of this dissertation for free, or I have a license to use be it a private license or a license granted to me as a developer by a client of mine.

Another part of my professional considerations is to ensure my work is meaningful and the reason behind my dissertation is not to cause harm. I believe my work is meaningful as I have close encounters with Alzheimer's, dementia, and caregivers daily, and strongly believe this dissertation could positively impact their lives.

### 4.2 Legal Considerations

This section outlines considerations taken in regard to different laws. While some computing laws such as the Computer Misuse Act (1990) is not relevant to this dissertation, others are. The main ones I will ensure I adhere to are in regard to people's data.



The Data Protection Act (2018) and the General Data Protection Regulation (GDPR) are both related to the requirements I must follow in regard to the storing of personal data. This is relevant due to requiring users' emails and phone numbers amongst other pieces of data. To adhere to this, some considerations are made to ensure all consent from a user is one while stipulating exactly how and why I need the information. I must also only collect the minimum amount of data needed to achieve functionality of the system.

I must also consider the Privacy and Electronic Communications (EC Directive) Regulations (2003) which state I must have a user's consent in order to send them emails or text messages and must allow for the user to unsubscribe from mailing lists or turn off these notifications.

These will not be complicated to adhere to, however, they are considerations that must be made and must be understood and followed, as it would not only be illegal not to, it would also be unprofessional and unethical.

### 4.3 Ethical Considerations

This section outlines ethical issues considered in this dissertation. The first ethical consideration made was while designing a proposal for this dissertation. I had intended to create a system to benefit patients with Alzheimer's and Dementia. To test this, however, would require testing the system on patients. This would have been complicated when searching for ethical approval for this dissertation. Due to this, the idea behind this dissertation was changed to instead focus on assisting caregivers of patients with Alzheimer's and Dementia.

Another ethical issue was to ensure I obtained informed consent by participants of different interviews and focus groups and will ensure I obtain informed consent when doing the evaluation of the system. To ensure I obtain informed consent, I have and will fully describe the purpose of the interview or focus group. I will fully explain the system and I will allow potentially participants up to two weeks to decide if they would like to participate or not. I will also allow a participant to leave the study or interview whenever they want.

As a student of Heriot Watt, I also had to ensure I followed the universities ethics policy. Some of the ethical principles stated in this ethics policy that have not been mentioned already include the prevention of harm, respect for participants, confidentiality, appropriate use of rewards and incentives, and anti-discrimination (University Research Ethics Committee, 2017). All of these principles are also adhered to, as well as ensuring I obtained ethical approval from the university to carry out this research.

In conclusion, I feel all ethical issues have been considered and I have attempted where possible to ensure I do the right thing. No part of this dissertation has any malicious aims, nor will data gathered from users be misused for my own personal benefit. At all points of this dissertation, any ethical conundrums shall be consulted with my supervisor.

#### 4.4 Social Consideration

This section outlines social issues considered in this dissertation. The key social issue being it may add to unemployment rates as, should this be widely adapted, less caregivers would be needed, hence leading to a loss of jobs.

On the other side however, it would increase the ease of interacting with the elderly and give more opportunities for families who live far apart to interact with their parents' or grandparents' care schedules. In these situations, it would also allow for families to have more involvement in ensuring their patient is in a comfortable environment, at least temperature wise.

In the current climate of the Covid-19 pandemic, the use of this system would hugely benefit society by decreasing the risk of elderly patients catching this virus. It would also reduce the strain on the NHS and local councils through allowing for routine tasks to be performed through the application without risking caregivers' lives.

## 5: Project Management

The project will follow the test-driven development (TDD) programming practice. With TDD, before writing the implementation code, automated unit test cases are created to test the new functionality (Maximilien & Williams, 2003). These tests will be created with Phpunit and Jest. The implementation of the project will also be using the agile methodology, with sprints lasting two weeks. Each sprint will finish with a meeting with Dr. Stefano Padilla to ensure progress is being made efficiently. The first week of each sprint will focus on creating tests for the functionality to be created in the given sprint.

To ensure test driven development is followed, a continuous integration / continuous deployment (CI/CD) practices will be followed. This allows for running the Phpunit and Jest tests to be ran in a different environment upon committing to GitHub. Should the tests pass, the commit on GitHub would then be merged to the master branch and deployed to the live server.

### 5.1 Risk Analysis

The risk analysis includes a likelihood rating, a consequence rating, a risk rating and an action plan.

The likelihood will be evaluated from one to five. One is unlikely, and it is believed to have less than a 10% chance of happening. Two is seldom, and includes risks that can happen about 10 to 35% of the time. Three is occasional, and will happen from 35 to 65% of the time. Four is likely and will happen 65 to 90% chance of happening. Five is definite, and will almost always occur.

The consequences will be evaluated from one to five. One is insignificant, meaning it would cause a near negligible amount of damage. Two is marginal, and may cause minor damage. Three is moderate, and may cause a sizeable amount of damage. This cannot be overlooked. Four is critical, and may cause a great deal of damage. This hazard must be addressed quickly. Five is catastrophic, and may cause an unbearable amount of damage.

The risk rating is assigned based on the likelihood and impact, and will be evaluated from one to four. One is low, meaning there is no significant threat to the project. Two is medium, meaning it requires steps for prevention. Three is high, and calls for immediate action. Four is extreme and is a high priority.

The action plan will be the steps taken to reduce the likelihood or impact of a risk, and what will be carried out should the risk happen (Yahnke, 2019).

### 5.1.1 Risk table

This section will contain the risk table. The risk table shows a list of risks along with the likelihood of the risk happening, the consequence of the risk happening, the risk rating, and a reference to the action plan which link to the action plan section below.

**Table 8:** Table showing risks and information about these risks

<b>Risk</b>	<b>Likelihood (1-5)</b>	<b>Consequence (A-E)</b>	<b>Risk Rating</b>	<b>Action</b>
Loss of data	2	C	Med	Plan A: backup everything on save.
Unethical research	1	E	Med	Plan B: gain ethical approval.
Incorrect test case	3	D	High	Plan C: manual tests as well as automatic tests.
Broken hardware	4	A	Med	Plan D: obtain backup hardware.
Broken backup hardware	3	E	Extreme	Plan E: request extra backup hardware.
Migration failure	2	A	Low	Plan F: use local system for demo.
Underestimating time required	1	D	Low	Plan G: plan for double the expected time.
Breach of user data	1	E	Med	Plan H: ensure high security when designing solution.
Using software without a license	2	E	High	Plan I: ensure any software or libraries used are open source or pay for the license.

### 5.1.2 Risk matrix

The risk matrix is used to graphically calculate the risk rating of a given risk based on the likelihood and the consequence. The four levels are colour coded, where green is low, yellow is medium, orange is high, and red is extreme.

**Table 9: Risk Matrix**

<b>Unlikely (1)</b>				<b>7. Underestimating time required</b>	<b>2. Unethical research 8. Breach of user data</b>
<b>Seldom (2)</b>	<b>6. Migration failure</b>		<b>1. Loss of data</b>		<b>9. Using software without a license.</b>
<b>Occasional (3)</b>				<b>3. Incorrect test case</b>	<b>5. Broken backup hardware</b>
<b>Likely (4)</b>	<b>4. Broken hardware</b>				
<b>Definite (5)</b>					
	<b>Insignificant (A)</b>	<b>Marginal (B)</b>	<b>Moderate (C)</b>	<b>Critical (D)</b>	<b>Catastrophic (E)</b>

### 5.1.3 Action plan

The action plan lists the actions taken to either mitigate the impact of a risk, such as plan E; aim to prevent the risk from happening, such as plan A; or the plan that will be carried out should the risk happen, such as plan C. The plans are designated a letter which links back to the risk table above.

Plan A: all documentation will be backed up on save with auto saved enabled to ensure it saves at regular intervals. Documentation will be backed up to OneDrive. When coding, the code will be auto saved, and upon completing a feature, it will be committed to GitHub to create a backup.

Plan B: ethical approval will be gained before using any human participants in the research of this project. Everything will also be based by Dr. Stefano Padilla to ensure no ethical rules are broken.

Plan C: should the automatic tests fail, everything shall be manually tested to see if the issue lies in the automated tests or in the code. In a worst case scenario, the CI/CD system will have a bypass to force deploy code.

Plan D: a second Raspberry Pi and necessary sensors have been requested from the student equipment fund, to be used should the original Raspberry Pi or sensors break.

Plan E: should the original Raspberry Pi or sensors break, to mitigate the impact or backup hardware breaking, new backup hardware shall be requested from the student equipment fund, or ordered on amazon with next day delivery.

Plan F: should the database migrations not work correctly, and cannot be fixed in time, a demo of the project will always be available on a local system to show proof of concept and to show the system works.

Plan G: to lower the chance of underestimating required time, all sprints are planned with double the time for each piece of functionality as expected to be needed.

Plan H: to ensure data is not breached by external entities, many security protocols will be in place such as not storing API keys on GitHub, using Laravel for its built in security tools, using an environment file on Laravel forge when deploying application to ensure there is no access to API keys, or database information.

Plan I: to ensure software without a license is not used all software will be monitored and licenses will be checked prior to using the software.

## 5.2 Plan

This section will contain and explain the sprint plan, along with explaining the Gantt chart. Essentially, this section is all about what will be done and when it will be done. This section will be divided into two sub sections, one about the sprint plan and one about the overall plan.

### 5.2.1 Sprint plan

The sprint plan will outline what programming tasks will be done and when. Each sprint will be divided into two weeks, where the tasks associated to the sprint should take two weeks to complete.

**Table 10:** Sprint plan numbering sprints and stating tasks to be performed in each sprint

Sprint	Tasks	FR
1	Set up CI/CD system, domain, sub domains, GitHub repositories, GitHub branches, Laravel Forge, the database and hosting. Commence the iterative process of designing the front end of the system. Set up Laravel projects and database connectivity. Test all systems are set up correctly.	FR 1
2	Create unit tests for creating new user, updating user details, user login, and authentication for different user types. Create API functionality for creating new user, updating user details, user login, and authentication for different user types. Create front end for creating new user, updating user details and logging in.	FR 2 FR 2.1 FR 4
3	Create unit tests for adding events to calendar, getting events from calendar and updating a calendar event. Create API functionality for adding event to calendar, getting events from calendar, and updating a calendar event. Create front end for a patient view, ie, calendar, time, date, and weather forecast.	FR 3 FR 3.1 FR 3.2 FR 5 FR 6
4	Create unit tests for sending alerts, setting thresholds for sensor data, and getting data from a sensor. Create API functionality for sending alerts, setting thresholds for sensor data, and receiving data from a sensor. Create front end for a caregiver view of the system.	FR 7 FR 9
5	Create Python system's hardware (connection between sensors, raspberry pi, monitor and internet). Create Python application to gather data from sensors and send the data to the API. Create Python application to show the patients view.	FR 8 FR 8.1
6	Create unit tests for uploading an image. Create API for storing, getting, and changing an image. Create front end for uploading an image and for changing an image. Perform evaluation.	FR 10 FR 10.1 FR 10.2
7	Test the system. Fix any bugs. Make changes based on the evaluation.	

### 5.2.2 Gantt chart

This section is about the overall plan of this dissertation. The main part of this section is the Gantt chart.

“A Gantt chart is a powerful and the most preferred visual reporting device. It is also a project planning tool used for conveying a project’s schedule” (Kumar, 2005).



## Akal

## Alzheimer's Calendar

Start date:

[illegible]