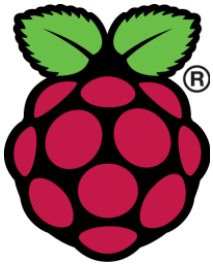
Institute of Technology Carlow
Kilkenny Road,
Carlow

Bsc (Hons) Software Development



Home SecuriPi

Project Report

Author: Karl Redmond
Student Number: C00196815
Submission Date: 18/04/2018
Supervisor: Paul Barry

Abstract

The purpose of this document is to provide a description of the final status of the project. The document will contain an introduction to the project, a detailed description of the submitted project, conformance to specification and design, learning achievements and an overall review of the project.

Table Of Contents

1. Introduction

The following document will provide a detailed description of the overall progress and final status of the Home SecuriPi project undertaken by the author. The document will be discussed under 5 headings including a description of the submitted project, description of conformance to specification and design, description of learning outcomes, a review of the project success and finally some acknowledgments.

The project description will cover a detailed outline of the features and functionality that the product aimed to satisfy, including hardware, software, networking and user interfaces.

Description of conformance to specification will discuss the final outcome of the project with regards to the initial proof of concept and anticipated design of the project. Justifications of any changes to specifications throughout production will be provided in this section, if any exist.

As mentioned, the description of the learning outcomes experienced by the author and developer of the Home SecuriPi project will be discussed under two main areas of focus, personal learning outcomes and technological learning outcomes.

The review of the project success will cover areas such as what went right, what went wrong, what is still outstanding, would the approach to the undertaking of the project differ in the future with the experience gained throughout production, what advice would be given to another party attempting a similar project and finally, were the technology platforms chosen the correct ones for production.

2. Project Description

2.1 Overview

The Home SecuriPi product is a system which can be installed in any persons home who has an internet connection and a smartphone or PC. The main function of the system is to monitor motion around different points of the home, taking still images once motion has been detected, and providing notifications via SMS upon detection. The system stores the images for later viewing. The product also allows end users to mimic movement around the home, by controlling appliances remotely, thus providing a means to deter intruders. Live image updates included on the home page, allow users to view motion activity on their home at any time, from anywhere around the globe. Also included in the project is the functional ability to remotely communicate, verbally, with guests (wanted or unwanted) at the home. A final functionality of the project is the provided ability to open a dog kennel. This functionality could be easily extended to the opening of the front door for family and welcome guests.

The systems functionality is provided through a web application. Once the users have successfully logged in to the application, using any device capable of running a web browser, all of the functionality is available to them. The system runs two Flask servers, one hosted on Pythonanywhere and the other hosted on the Raspberry Pi, which catches commands to be carried out through an API implementation.

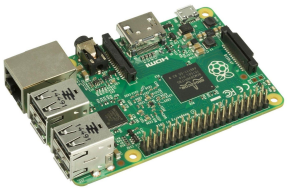
2.2 System Description

This system is comprised of hardware devices, including motion sensors, cameras, speakers, and microphones, which are mostly controlled by software running on a Raspberry Pi. The interaction with the system is provided through a web application running on a users PC or smartphone, which is in turn communicating bi-directionally with a Raspberry Pi and a cloud server hosted on Pythonanywhere. A Networking aspect involving a Dynamic Domain Name System(DDNS) was also necessary for communicating with the Raspberry Pi from a connection over the internet.

2.2.1 Hardware Devices & Components

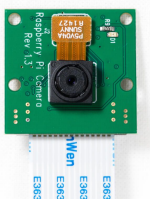
All hardware devices are described in full detail in the Design Document, section 2.1, and are included in this document to provide a more complete description of how the product functions. Software used is described in deeper detail in the Research Manual, section 7.

The Raspberry Pi - As mentioned in the Research Document (section 3.2), the Raspberry Pi is a credit card sized computer originally designed for education. Boasting the capability to run a Linux based operating system, with the ability to run several different processes, the Pi is the central component to the entire project. The Pi provides the ability to listen to the motion sensors, and provide SMS messages through the utilization of a third party platform as a service(PaaS), called Twilio. The Pi provides the capability of controlling the cameras, the speakers, as well as providing the additional functionality of opening a door or controlling a light.



Motion Sensors – The motion sensors provide the functionality of detecting motion. While no motion is detected, a 3.3V LOW(0) digital signal is sent to the Pi, upon detection of motion, a 3.3V HIGH(1) digital signal is received by the Raspberry Pi, ushering the Pi to take a picture and store it on the cloud server. The module is powered by one of the Pi's 5V output pins, however whenever discussing pins we feel it is important to note that the General Purpose Input Output(GPIO) pins **are only capable of receiving a 3.3V digital signal.** In the case of the motion sensor used for this project, the output pin does provide the signal over 3.3V, however, not all modules will do this, and it is important to **check the specifications of the module before proceeding with any circuitry connections.** If using a module which supplies a 5V output it is possible to reduce it to a 3.3V with the use of resistors. Once successfully stored, the Raspberry Pi sends an SMS message to an end users mobile phone.

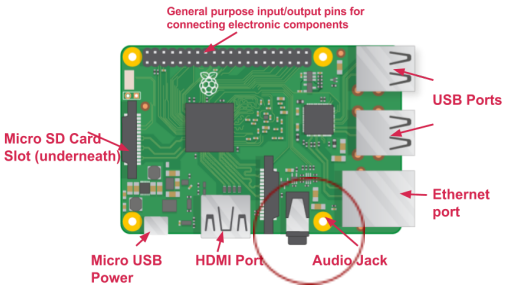
Cameras – The camera provides the capability of taking pictures, and also provides live updating of still images on the home screen of the application. This capability will allow the users to view who is at their home, upon notification or at any time they feel the need, and respond by asking the individual who set off the sensor to leave the property, call back later, or mimic movement by turning on/off appliances in an effort to deter any intruders discovered. If all else fails when an intruder or unwanted guest viewable in the still image taken by the camera does not leave the property, the dogs can be released.



Relay Switches – The relay switch is a crucial aspect to the system, allowing the control of higher voltage appliances situated around the home. In the case of this project two relay switches are utilized, one for controlling the state of the light(On/Off) and the second for controlling the solenoid door release of the dog cage(Locked/Released). The reason for

the use of the relay switch is to protect the Raspberry Pi's onboard circuitry, due to the fact that the Pi's GPIO pins are **only capable of handling 3.3V-5V signals to or from an external module** such as the relay switch under discussion. While the Raspberry Pi is capable of providing a 5V supply to external modules, it is important to note that the **receipt of digital signals coming from a 5V module to a GPIO pin will burn out the connection of the Pi.** In the case of the relay switch, a signal is sent *to* the relay rather than the Pi receiving a signal *from* the relay, which is fine, but we felt it a necessary warning for any other person who might attempt a project of similar ambition. When the relay receives a HIGH input, an internal circuit of the relay gets closed through magnetisation caused by the HIGH(Live) digital signal, thus allowing power to a device to be controlled by either completing the circuit (HIGH signal) or breaking the circuit(LOW signal). The mechanics are described in deeper detail in the Design Document(2.1).

Audio – The Raspberry Pi's audio jack shown in the image to the right(circled) is utilised to play audio which is recorded over a devices native microphone. The device which records the voice is any device capable of running a web browser. The Raspberry Pi receives this recording in wav format and plays it via a speaker attached to the audio jack.



Solenoid door lock – The dog release mechanism is employed through the use of a solenoid door lock, pictured on the right, which is powered over a 12V feed and broken through a relay described earlier. The mechanics of this put simply; breaking the live feed to the solenoid causes the door to lock, while completing the circuit causes the door to open. When a user wishes to "Release the Hounds", they send a message to the Raspberry Pi telling the Pi to send a HIGH digital signal to the relay. This causes a magnetic phenomenon to happen through the mechanics of the solenoid, pulling the latch inwards, thus opening the door. The light works in the same fashion and is described in deeper detail in section 2.1 of the Design Document.

2.2.2 Software

Microphone – Most personal devices these days capable of running a web browser will have a microphone attached, specifically smartphones, laptops and tablets. HTML5 now provides a means to access native hardware devices like the microphone, the camera, the speakers, GPS, accelerometers and

GPU's. The Home SecuriPi system utilizes the access provided by HTML5 to the microphone. By allowing the user to record a message on their personal device and send it to the Raspberry Pi, the Pi can play the file received via the audio jack described in the previous paragraph. The Home SecuriPi application uses the microphone to record a stream from the microphone and format it to wav format for transmission to the Pi. Other formats are available such as mp3, however a python library capable of playing wav files was discovered during research for the project, thus our reasoning for formatting to wav.

DDNS – As mentioned in the overview(section 2.1) of this document, there is a networking aspect involved with the Home SecuriPi project. This is necessary as users need to be able to communicate with the Raspberry Pi situated on their premises or property, from outside the internal network. To allow commands to be sent to the Raspberry Pi from the outside world, we employed the use of a DDNS provider. As any home's router gets assigned an IP address by their Internet Service Provider(ISP), which is subject to change, we needed a way for our cloud server to always know the IP address of the home router. How this works put simply is; when the ISP changes the IP address of the router, the router notifies the DDNS provider of the change, who then update the IP address on record, meaning our cloud server doesn't have to change anything, it simply points to the DDNS address(xxx.gotdns.com) which always returns the correct IP address of the home router. From here the traffic is sent to a port on the home router eg. xxx.gotdns.com:8087, which will forward the information to our server running on the Raspberry Pi, which is assigned an IP address through the Dynamic Host Configuration Protocol(DHCP) of the router. The server on the Pi will also be configured to listen on a port, which can be contacted by combining the internal IP address eg. 192.168.1.1, with the port number that the server is listening on, becoming 192.168.1.1:8090(for example). As internal IP addresses on a network can change also, an IP address is reserved for the Raspberry Pi, meaning whenever the Pi is turned on it will always be allocated this address. A rule was added to the home routers configuration, which instructs the router to forward any traffic coming in over port 8087 to the Flask server running on the Raspberry Pi.

Twilio – As mentioned earlier in this document, the Home SecuriPi project harnesses the Twilio PaaS to send SMS messages upon detection of motion and successful storage of a still image. A Python helper library installed on the Raspberry Pi allows a simple method for sending SMS messages to desired phone numbers. When you register for a free account on Twilio you receive an account sid, and an authentication token. Using these credentials, the Pi sends a message via HTTP to Twilio's servers, who forward the message in SMS format. Embedded in this message is a link to the Pythonanywhere server which the user can click on using their smartphone to access the functionalities of the application.

Flask Servers – The system comprises of two Flask servers, one running in Pythonanywhere, with the other running on the Raspberry Pi. Flask is a microframework, used for building web applications. This framework was considered the best option for development on this project, as it is extremely lightweight, and is specifically for Python, which is a core development language of the Raspberry Pi. Flask is described in more detail in the Research Document section 7.8.

2.2.3 Architecture

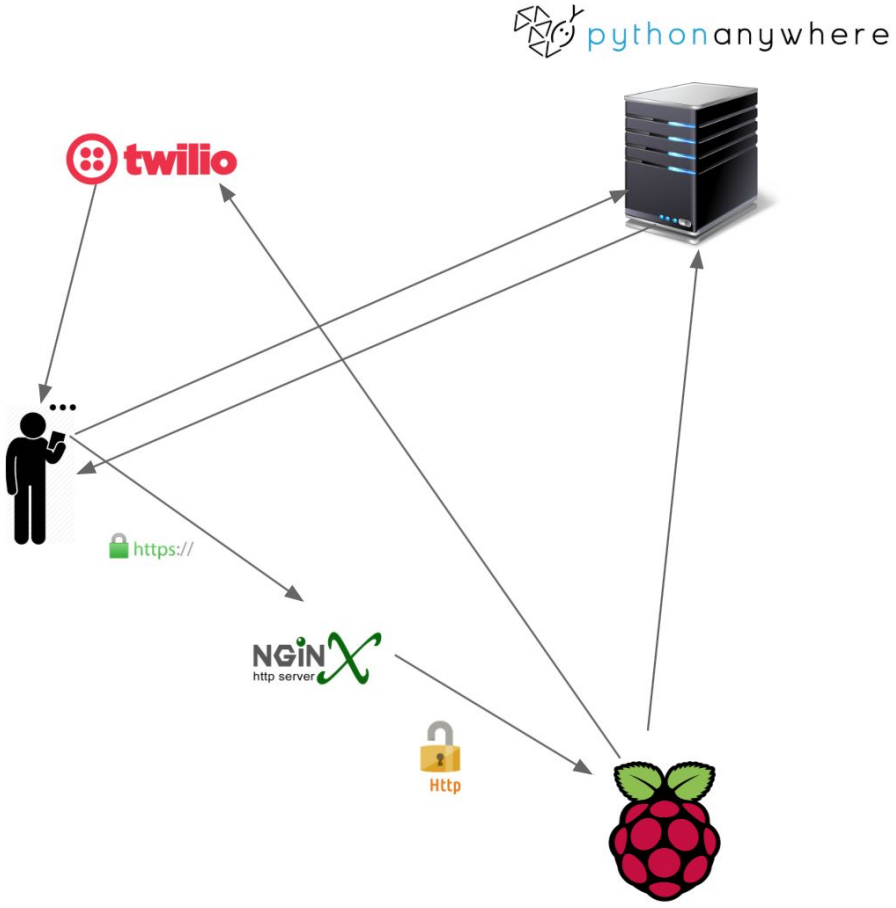


Fig. 2.2.3.1 System Architecture

To describe how the architecture of the system works we will focus on Fig. 2.2.3.1 with reference to a scenario from Section 3 of the Functional Specification.

Functional Specification(Section 3) Scenario 3.1 – "*A "bad guy" is scoping out your property, and you are not home. In a rural area there may be no one else around to notice the activity, and the only deterrents are the flimsy door locks, the mild indifferent traffic on the main road, and the distant barking dog. The bad guy is alone. Taking his time to survey the weak spots of your home, peering in windows to take stock of valuables, and totting up how much money he can make from your possessions, he (or she) rub their hands together.*
But wait, a T.V. just turned on inside the house, maybe they are not alone after all. The bad guy scarpers, startled by the sudden activity. As referenced in the Problem Statement in the Functional Specification, a T.V. turning on is one of the main deterrents reported by convicts.
Notifications of movement around your home through the use of motion sensors, along with providing the necessary means to control your appliances, our application will become your first line of defence with regards to protecting your home".

With this scenario in mind, the following will take place in the Home SecuriPi system architecture. The motion sensor attached to the Raspberry Pi will detect motion and send a HIGH digital output to a configured GPIO pin(see 2.2.1). Upon receipt of this signal the Pi takes a photo using the attached camera. The Pi then stores the image locally, and sends a HTTP POST request to the Pythonanywhere Flask server. The server stores the image and saves the image details(date, time, location of Home SecuriPi device and location of file in the system) to a database. Responding to the HTTP request with the size of the image received in the POST request, if the Pi ascertains that the size of the image is not correct, it will send the POST again until such a time that the Pi can confirm the image has been stored successfully. Once confirmation is achieved, the Pi POSTs a HTTP request to Twilios servers, with a message body and the correct credentials attached, Twilio then authenticates the credentials and sends an SMS message with the desired text intended for the recipient. The user gets a text message notification, opens the message and clicks on the embedded link. From here the user has access to the functionality of the Home SecuriPi application. If the above described scenario is taking place, and the individual who set off the motion sensor is not wanted around the property, the user can defend their home in a number of ways. They can use the application to tell the Pi, which is also running a Flask server to catch commands, to turn on a light in an effort to mimic movement in the home. If this does not work they can either send a pre-recorded voice message warning the individual that they are being watched, or they can record a unique live voice message which may be more descriptive of the individual, ie. "You in the striped shirt carrying the bolt cutters, leave or I will release my dogs". The application can forward the desired voice message and play the message via the audio jack. If all of this fails, the user can use the application to send a HIGH digital signal to the relay attached to the solenoid door lock, thus releasing the dogs. This should be enough to get the intruder to leave, provided they can outrun the dogs…

While this is one possible scenario, the system covers other scenarios whereby the visitor to the home may not be unwanted. In such a case Home SecuriPi provides the same voice functionality where you could inform the guest where the key is located for the house, ask them to give you a phone call or, with a relocation of the solenoid door lock, we could open the front door for them. We could obviously add more solenoid door locks around the property allowing access to specific parts of the house, as the Pi still has numerous GPIO pins that can be utilized.

2.2.4 Application User Interface

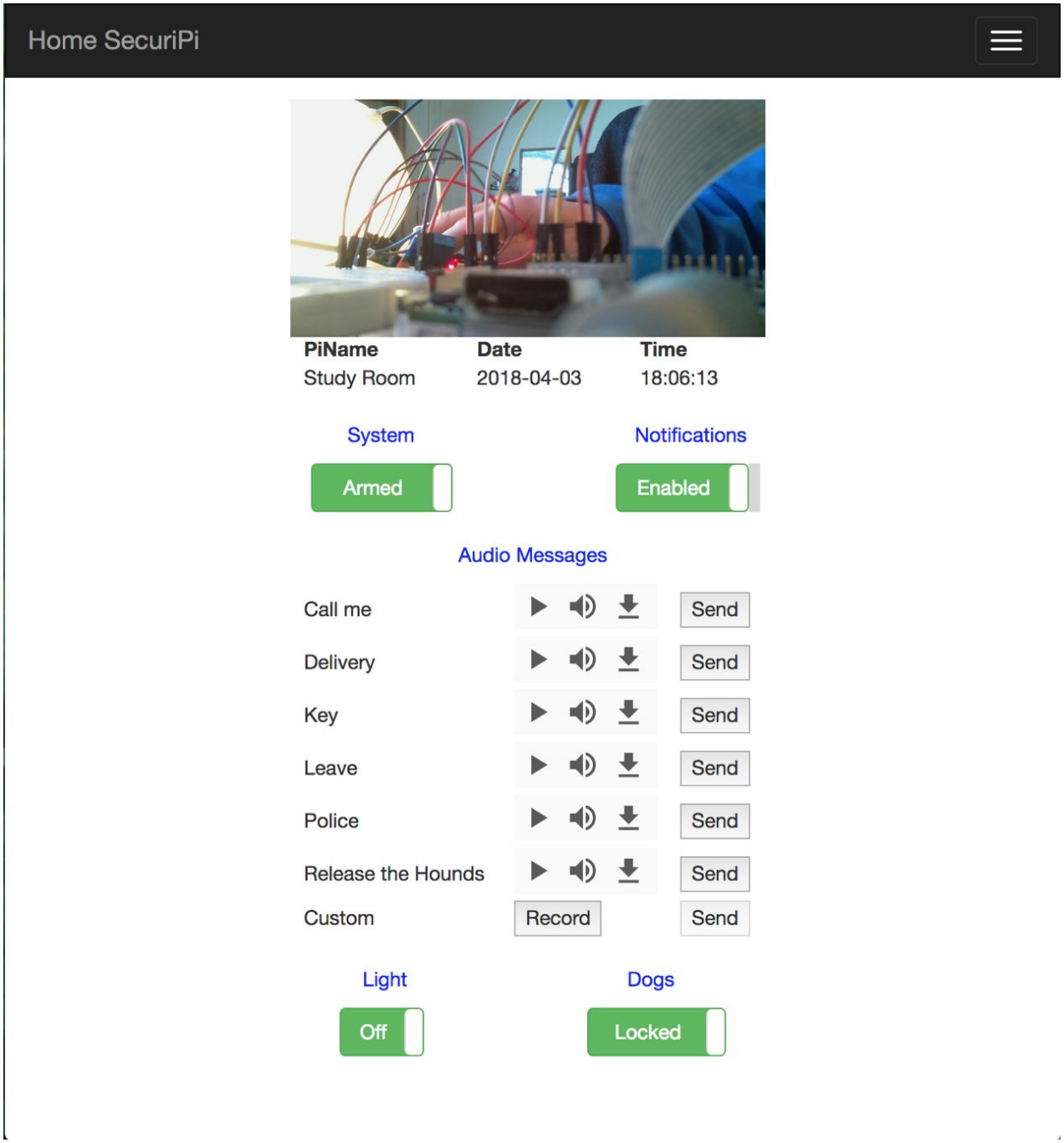


Fig. 2.2.4.1 Home SecuriPi Home Screen (SmartPhone)

Figure 2.2.4.1 shows the landing page of the Home SecuriPi system. When the user clicks on the link embedded in the text message, they are presented with this screen. From here they can access all the functionality that Home SecuriPi offers.

System – From this screen the user can arm or disarm the system, meaning if the system is in a disarmed state it will no longer take pictures or look for motion. A user may wish to disarm the system for any number of obvious reasons.

Notifications –The user also has the option of enabling or disabling notifications, meaning that the system will keep taking pictures but will not send any notifications. A user may wish to do this if they are aware that there will be a lot of motion around the home for any given reason, but they still want to take pictures, which can be viewed in the History section of the application which will be described later in this document.

Light – Clicking the "Light" toggle is self explanatory(I hope), turning On/Off the light.

Dogs – Likewise with the "Dogs" toggle, once clicked, the dogs are released!

Audio Messages – There are a number of audio messages visible on the home screen, which include a brief description of the message along with the option to "Send". From here the user can play the message whenever they please, to hear what the message dictates. Once the user clicks the "Send" button associated with a message, then the pre-recorded wav file will be sent to the Raspberry Pi, which will play the message through its attached speaker. The option to "Record" a custom message is also given, allowing the user to record a unique live message for transmission. When they have finished recording the message, they can click "Send" and the Pi will play the message.

The image displayed on the page is the latest entry to the database, and gets updated through ajax calls to the Flask server running on Pythonanywhere. This image gets updated live, meaning that the user can see a live feed of still images as there is movement discovered around the property, within reach of the motion sensor. The location of the security device is also updated under the heading "Pi Name", along with the "Time" and "Date". With this in mind, if more than one device is installed on the property, we can see whereabouts on the property the movement is occurring ie. Study Room, Front Door, Back Door etc.

The user(s) of the system are also made constantly aware of the current status of the system, through the toggle switches "System", "Notifications", "Light" and "Dogs" which get updated through Ajax calls made to the Home SecuriPi device. To elaborate, if there is more than one user of the system at any given time, and one user decides to "turn on" a light, the other user(s) will be made aware of this action, as the toggle will automatically switch state. The same can be said for each toggle. For illustrative purposes, see Figure 2.2.4.2 to see the system in a different

state. This image also offers a view of the drop down list provided once the “Hamburger” icon is clicked.

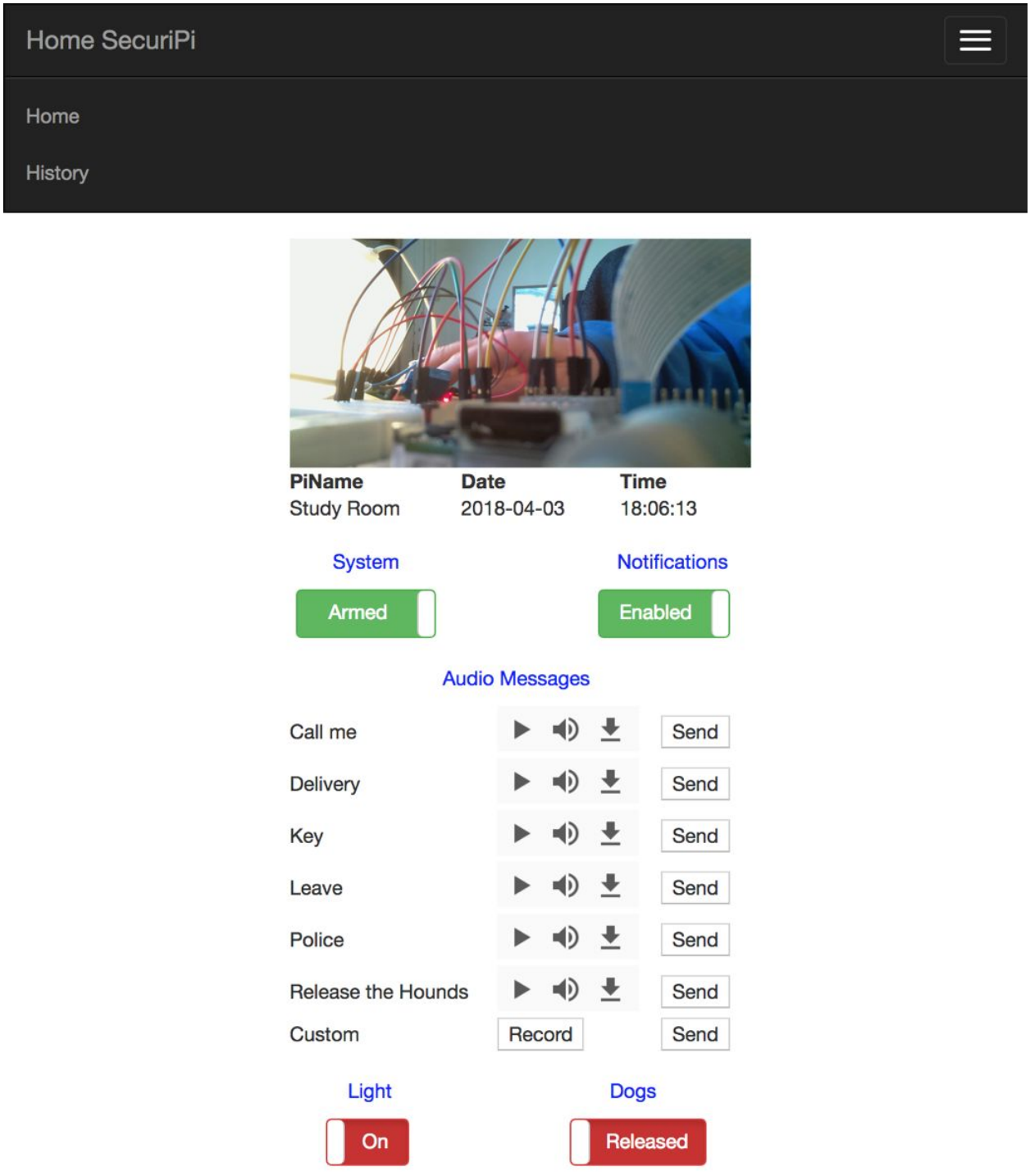


Fig. 2.2.4.2 System State Change & Drop Down List(SmartPhone)

The application also offers the ability to view all the images that have been taken. As can be seen in Fig. 2.2.4.3, the application provides two arrows which can be clicked to scroll through the images from start to finish or vice versa. The location, date and time of the image capture are also displayed. This could be useful for police investigation should they need to identify any intruder who has caused damage to an individual's property or otherwise infringed upon someone's rights. Fig. 2.2.4.3 is a screenshot taken from a laptop device, whereas the other figures have been screenshots taken on a smartphone. This indicates the responsive nature of the design whereby the screen real estate on a laptop allows for all the navigation options to be displayed in the navigation bar, as opposed to the smartphone, Fig. 2.2.4.1, where the "Hamburger" icon in the top right hand corner must be clicked to view navigation options.

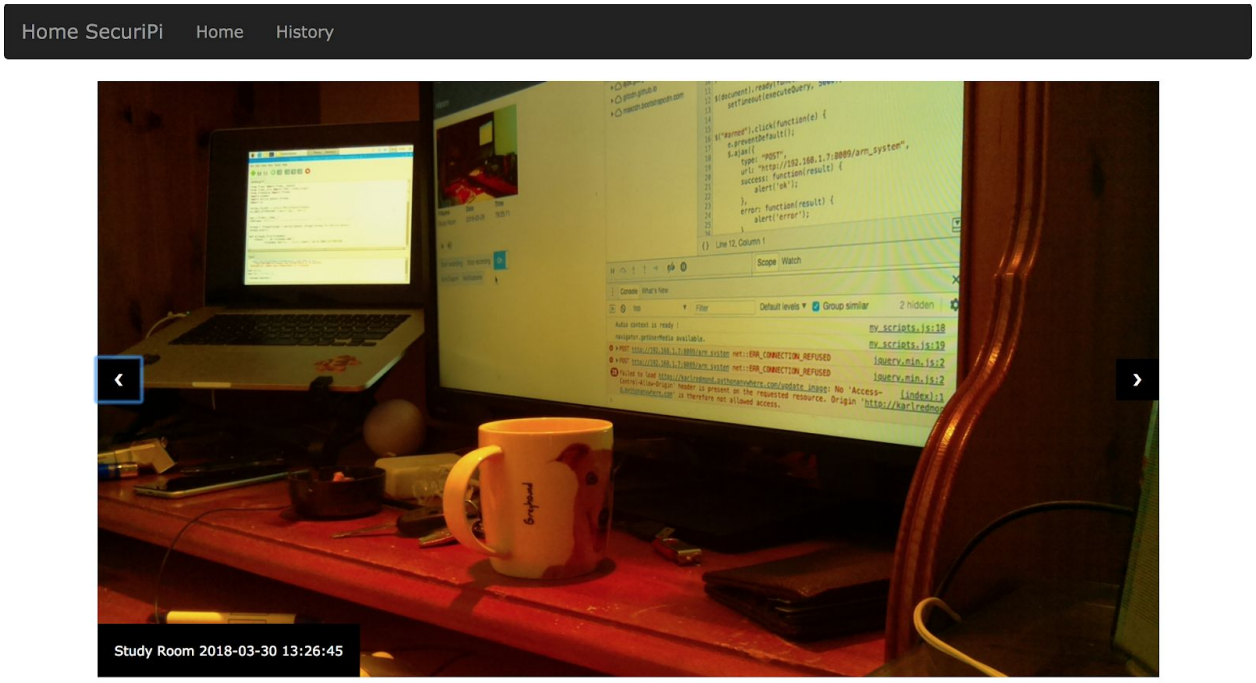


Fig. 2.2.4.3 History Screen(Laptop)

3. Conformance to Specification and Design

The following will describe how closely the final project adheres to the proposal that was submitted in October, 2017. Any changes in specification will be outlined, with justification provided as to the reason why there was any deviation from the proposal.

During October, 2017, all students were given the option to make a project proposal, or to choose from a selection of project specifications supplied by a number of delegated supervisors. Before the supervisors made available their proposals, I had an idea of my own which was submitted. It was very brief, but significantly conveyed enough complexity to be considered worthy of this fourth year project. Provided below is this brief project proposal:

> *"Briefly, my idea would evolve around a home security web application utilising the Raspberry Pi. My initial thoughts are to have a sensor(s) and a camera(s) attached to the Pi. When a sensor is activated the corresponding camera will turn on and the user will be notified using some form of VoIP, thus allowing the user to potentially communicate verbally with whoever set off the sensor. If it is someone the user knows, they will be able to open the front door, possibly using an actuator or something similar, or initiate a "Release the Hounds" scenario(I jest), from an endpoint call. To do this the Pi will be set up as a dedicated server.*
> *That's it in a nutshell. Tech used will obviously be the Pi and accessories, probably coded using python and Django.*
> *I've never done anything like this, but would love to do something involving the Pi as the IoT is the future!"*

This brief proposal was accepted, and I was assigned Paul Barry as my official supervisor.

3.1 Specification Refinement

When the proposal was accepted, Paul and I set out to refine the project into something that was considered to be achievable, ambitious and useful. For a fourth year project to be considered acceptable it must meet all of those criteria.

Initially we set out some primary goals that would be enough to provide a substantial project. We produced a list of crucial features that the application should provide as a minimum, with additional "nice to have" features to be added if time permitted.

The core features included motion detection and notification, live stream of video, voice communication, mimicking of internal movement and the ability to view all of the pictures that

have been taken and stored. The nice to have functionality was that of being able to open a door. All of these features are described in detail in the Functional Specification, section 5.

3.2 Conformance

Overall, the final project submission conforms almost precisely to what was intended in the initial proposal and subsequent final specification. All of the core functionality is provided in the application, along with the fringe functionality, which in the beginning was considered unachievable given the time frame. Any end user of the device will be able to recognise all of the functionality mentioned in the functional specification, with a few slight alterations which will be discussed next.

3.2.1 Alterations

Initially it was considered to have a means to communicate, verbally, bi-directionally i.e. An individual would be able to speak back to the owner of the home over a microphone attached to the Raspberry Pi, which would be hosting a VoIP server. However, during research it quickly became apparent that this was out of scope given the time frame, and in order to achieve this, too much time would have to be invested into this functionality, meaning the other functionality would have suffered. It was decided that allowing the owner of the property to speak to the individual would suffice, as it is a security device, not a social device. This is a slight alteration, however we feel that it still meets the requirements of the functional specification which states(see Functional Specification, 5.1.3):

> *"Our system will provide a means to communicate verbally with a person at your house, be they friend or foe. If the visitor found at the home is considered unwanted, they can be asked to leave, or informed that the Gardai are on their way, likewise if the visitor found is friendly, they can be asked to call back later*".

An other alteration, which is not apparent to the end user but we feel necessary to mention, is that of the initial project proposal stating the use of Django as our application framework. It was decided instead to use Flask, as this is a microframework. Given that the Raspberry Pi is of limited resources, we felt that the use of Flask was more appropriate considering its light weight on resources.

A final alteration was that of the Live Stream functionality, which states(see Functional Specification, 5.1.3):

> *"Another important feature of the application will be the ability to view a live stream from the camera(s). This feature will be accessible at any time, allowing the user to monitor what's going on around their home, whenever they desire".*

This functionality was setup during development, however the image displayed(Fig. 2.2.4.2) was grainy and of low quality, and tended to freeze for extended periods of time. This was overcome through the utilization of Ajax calls made to the database, which retrieves and updates the image as it comes into the database without any user interaction or refreshing of the page. This is a "Live Stream", however not the way it was originally intended. We feel it still meets the requirements of allowing an end user to monitor motion as it occurs.

This sums up the alterations made with consideration to the original proposal. Overall we feel the project provides all the functionality we set out to achieve, and any alterations made have no obvious bearing to the end user.

4. Learning Outcomes

The main purpose of the fourth year project, in my opinion, is to test the students ability to create a unique application, from start to finish, encompassing all aspects of the Software Development Life Cycle. From the beginning I wanted to push my boundaries, step outside my comfort zone, and put myself in a position where I would be forced to do heavy research into electronic circuits. Having no experience with electronic circuits, and considering that IoT is an area where I would like to focus my career path, this was the perfect opportunity to see was I up to the task.

4.1 Technical Achievements

As previously stated I had no prior experience with electronic circuits. Not only that, I had no idea how information could be retrieved via digital or analog signals from an electronic module. Coupled with the fact that Networking was only covered at a high-level in the first year of my college life, getting my system to work was a huge undertaking, one of which I am proud to have completed.

4.1.1 Electronic Circuits

At the beginning of the year, I undertook some external online courses to familiarize myself with electronics. These courses were undertaken through the online learning resource, Lynda.com. The two courses I undertook were:
Electronics Foundations: Fundamentals, available at:
https://www.lynda.com/Development-Tools-tutorials/Electronics-Foundations-Fundamentals/197537-2.html.

Electronic Foundations: Basic Circuits, available at:
https://www.lynda.com/Software-Development-tutorials/Electronics-Foundations-Basic-Circuits/507570-2.html.

While time did not allow me to complete these courses, the time that was invested into these courses gave me the confidence to move forward with my project, without the worry of burning out the Raspberry Pi circuitry, which was a huge concern as being a student I am of limited financial resources!
The knowledge I have gained in this area is an exciting prospect for me, and I can definitely see myself pursuing this area, even if it exists as merely a hobby. I would hope however that the knowledge gained will lend itself to employment options in the future.

4.1.2 Networking

While getting the Raspberry Pi or microcontroller to send information to the outside world is a relatively simple task, getting information back to the Pi to carry out commands proved a difficult obstacle. However, once one request was successfully achieved, completing the circle of the architecture shown in Fig. 2.2.3.1, the rest was relatively straightforward from there onwards. I have learned a huge amount about how networking works, whereby my system comprises of two servers communicating bi-directionally, along with client interaction from a browser. Integrating the Twilio PaaS was an insightful discovery, and I am no longer intimidated by the employment of external platforms. Port Forwarding, static IP address', DDNS and Reverse Proxying are all terms which I am no longer scared of!

4.1.3 Browser Security

HTTPS…. An issue facing all developers in the future, as most popular browsers such as Chrome, Firefox and Safari no longer advise the use of transmission over HTTP, giving warnings to the user that the site they are using, if hosted over HTTP, is insecure. Which it is. This will scare the uninitiated into using any site or web application which is implemented in this way. A minimal requirement for security now for an online application is the implementation of HTTPS, which encrypts transmission from the browser to the server. This proved a massive headache from the beginning of development. To gain access to the microphone or native hardware of a device, Chrome, Firefox and Safari *insists* that the channel of communication be over HTTPS, not allowing access to the hardware otherwise. During development I learned how to make my own Root Certification Authority, how to make a self signed cert, and how to add the cert into the trusted certificates of a browser, as well as informing the browser that it can use the Root Certification Authority created by myself. This allows a developer to create a local HTTPS environment where they can access all the hardware of the device, however for deployment, a certificate must be issued by a recognised Certification Authority. With this knowledge I can now create a HTTPS local environment which will be crucial as I move forward into application development.

4.1.4 HTML5

When conversing with my colleagues about HTML5 I was met with a considerable amount of "scoffing". But I believe that considering HTML5 is in its infancy, there will be huge leaps in the near future where developers will no longer have to create "native" applications, which were once needed to gain access to the hardware of a device. HTML5 eliminates this necessity, meaning web applications can be produced that can potentially run across all Operating Systems and hardware devices. Having learned about this technology, I look forward to gaining more knowledge into what it is capable of.

4.1.5 Raspberry Pi

With the creation of this project I have learned the technological aspect of using GPIO pins to control electronic devices. I now have the technical ability to read output from an electronic module and give input to an electronic module. Controlling electronic devices is extremely fun and interesting, and I never would have believed at the beginning of my college life that by the end of fourth year I would be able to open a door in my house, from any position on the planet. All through the use of a commodity hardware, the Raspberry Pi.

4.1.6 Reverse Proxying with NGINX and Apache

During development, as previously stated, I discovered that browsers will no longer send HTTP POST requests to an unknown or untrusted server living in a different domain. This instigated me to implement a reverse proxy using NGINX. I configured the proxy to work over HTTP, forwarding the requests to the specified internal device(the Pi) with the intention of extending the proxy to create an encrypted channel to the outside world, while forwarding the same request over HTTP on the internal network as shown in Fig. 2.2.3.1. I made it quite far in this, however at the last step while trying to gain a certificate from a Certification Authority, I found that you cannot use a DDNS hostname, it must be a static hostname. You can configure the hostname to point to your DDNS, however this requires the purchase of a hostname, which I could not afford. That being said, I now know how to configure a reverse proxy, which will no doubt come in handy at some stage in the future.

4.2 Personal Achievements

As with the completion of any difficult project or task, this project has brought a few personal achievements.

4.2.1 Go Hard, or Go Home

Go hard, or go home! This was a term bandied about while I was constructing roads and building runway strips in Alice Springs, the middle of Australia. It was hot, very hot, and this term was used to “toughen up” people on the verge of heat stroke. While there may not be an obvious connection with building a runway strip and producing an application, they are in fact almost identical. You have a desired outcome, you follow a specification, you use the correct tools for the job. If a rock has to be moved, you move it, regardless of how hot it gets. There were a lot of times throughout the development of this project where I felt like giving up. Obstacles in the way which I felt could not be moved, allowed doubt to creep in concerning my technical ability. But I have learned that with true grit and determination, almost every obstacle can be overcome.

4.2.2 Career Navigation

This project has given me the personal achievement of knowing where I want my career to go. This is a huge achievement, and to me the most important, as before now I had no idea what area I wanted to focus on. Not only that, but with this knowledge I can make adjustments to my CV to focus on certain companies, whereby I will hopefully find somewhere that I can get involved with IoT devices and programming.

4.2.3 Python

I cannot move past the personal achievements without mentioning Python. While the system I have created is relatively simple in its code(now that I know how), the amount I have learned about the power of Python is immense. During my internship in Unum, I came up against the need to send HTTP requests to Fitbit.coms servers, using C#. It was tedious, agitating and unnecessarily complex with no defined "rules" found while researching. There were many different ways to achieve the same result and no-one seemed to agree on the correct way to do it. Python provides a library which can be imported, and allows requests to be sent with one line of code! Through development of the Home SecurPi project I have learned that Python is, without a doubt, my personal programming language of choice, and I would jump on an opportunity to develop applications using Python in the future.

5. Project Success

Overall I am extremely happy with the final result of the project. We managed to develop all of the core functionalities along with the addition of providing the fringe features that we originally thought we wouldn't have time to do. I've learned a massive amount about Python, server configuration, networking, HTTPS, electronic circuits and more. Personally I feel I have grown as a developer, with the confidence to undertake any(reasonable) project or job.

5.1 Successful Aspects

While I would like to say everything was successful, that is rarely the case, as will be discussed in the next section. That being said, most things were achieved. We successfully implemented the motion sensor, with relative ease. The image capture went smoothly with no hiccups, using an imported library. Sending the notifications was much easier than anticipated, through the use of Twilio previously mentioned in this document. I think one of the main successful aspects is the fact that we created all of the intended core functionality, and then some more. Reaching the fringe feature of opening the dog cage was a huge success in my opinion, and something that I am proud of. This may be viewed by some as an easy thing to do, and for certain people it would be, but given my inexperience with such things it should certainly be deemed a correct entry to the successful bracket. Providing the ability to communicate verbally with someone on my premises from anywhere I have an internet connection is something I am also very proud of. This was one of the first things implemented, and getting this to work in a live, deployed environment , gave me a huge boost in morale, providing the fuel needed to keep going.

The most important successful aspect concerns one of the reasons I decided to undertake this project in the first place. As stated in my presentations, there were personal reasons regarding this project. My Father had expressed concerns about his personal safety. Being left a widower a couple of years ago has allowed my Father too much time to overthink things. When he expressed his concern to me and my siblings about his safety at home, we investigated existing home security solutions. One which was of particular interest was a company which provide a surveillance service. This service monitors motion on an individual's property, and will contact specific people or the Gardai if deemed necessary. That's it. That's all they offered. The price of this service was €30 a month, with installation fees leaving the cost for the first year at around €600.
Home SecurPi can be installed for less than €100, and provides users with options to defend their home other than calling the Gardai or friends, who may take some time to reach the property. I will provide a breakdown in cost for the Home SecuriPi solution as it stands:

1. Raspberry Pi @ €35.00
2. Motion Sensor @ €2.00
3. Camera @ €6.00
4. Solenoid Lock @ €6.00
5. Relay Switches @€7.00(for two)
6. Twilio Cost @€40.00 per annum
7. No-IP DDNS @€0.00
8. Total @€96.00

As is shown, the total cost of the system as it stands is €96.00. This is with a subscription to Twilio which could potentially be done away with, instead using an email address which can be utilized to provide push notifications to an end users smartphone, for free. Meaning the cost of one Home SecuriPi device could potentially be as little as €56.00, providing the protection of the existing mentioned security solution, with a host of extra features for almost €550.00 less, with no annual fees. This is potentially a huge saving and with some more development, I think this is a viable business opportunity.

5.2 Troublesome Aspects

Fig. 5.2.1 Home SecuriPi Model

Even though we managed all the functionality, not everything went smoothly. As mentioned previously in this document, HTTPS caused huge trouble from day one. A HTTPS connection was needed to secure access to the microphone of a users device. When this was first encountered, we deployed the application to Pythonanywhere to use their certificates, which allowed an encrypted channel to be accomplished and allowed access to the microphone. From here all development took place in a live environment. Everything functions in a live environment, however, for the purposes of demonstration we built a model of a home property, pictured in Fig. 5.2.1, with a test network environment where everything is connected through a router. This router provides IP addresses, using DHCP, to both the Pi and the laptop which is acting as the cloud server hosted on Pythonanywhere. This local test network could not be configured to allow access to the internet, which causes issues with both the sending of the SMS, and access to the microphone. As mentioned in section 4.1.3, we were able to make our own certificate to run a HTTPS server. This works fine for working the application on the device that the certificate was created on, but when the Raspberry Pi tries to send images to the server on the laptop, it first requests the certificate from the server on the laptop, which it does not trust at the TLS layer of the device, thus the request fails. It is possible however to demonstrate the microphone functionality through a file constructed on the local machine, which can be loaded in the URL bar of the browser with file://, which the browser trusts, rather than http://, which the browser does not trust.
All things considered we feel that this is a minor setback, considering that the application works in a live environment, which is where it needs to work.

5.3 Improvements

Time ran short towards the end meaning the implementation of the login was not successfully completed, and we would have liked to allow the users to edit the pre-recorded audio messages. As neither of these were a core functionality originally anticipated, and require only time to implement as the knowledge has been gained over the course of this year, we decided to focus on getting the application in as solid a state as possible rather than adding extra functionality. One other further development that we would like to implement would be the ability for the individual on the premises to be able to speak back to the user of Home SecuriPi, this was out of scope for this project, but would certainly be a valuable addition.

5.4 Approach

If I was to start the project again, there is not much that I would change with regards to the approach. A lot of research was needed and the time spent on this could not be avoided. The choices in technology were mostly delegated by the Raspberry Pi community, who nearly all employ the use of Python, with a rare few exceptions using Java or C++. Perhaps one thing I

would change was my approach to the documentation and my time management. I tend to work in chunks of time in different areas of the course modules, and maybe time could have been better spread amongst each module. It has been a very intense year, with no breathing room, so you tend to go into autopilot which can be a bad thing if a mountain is in front of you that you don't see. But with regards to the project I feel that enough time was given to it, considering any of the holiday time allocated throughout the year was all spent on the development of the project.

5.5 Advise

For anyone taking on a similar project I would advise researching circuitry before undertaking the construction of any circuits on the Raspberry Pi or similar device. It took me a lot of researching to find out that a 5V output sent from an electronic module to a GPIO pin of the Raspberry Pi can burn out the device. There are stories out there of peoples devices frying, and fires starting, so be wary.

With general regard to the fourth year project, I would say try not to be too hard on yourself, everyone will encounter problems, but the solution is always out there.

Be patient, work hard, don't give up and you will get through it. If you find yourself getting bogged down on a certain aspect, move on to a different one.

Don't be afraid to ask for help. Speak to your supervisor regularly, and if you feel you should head in a different direction, speak up, the supervisors are there to help and will listen to your ideas.

As a current student and from listening to past students there have been times when a student will not say that they are encountering problems, which is never a good idea. Speak up and someone will almost always have a solution.

Speak to your fellow students, bounce your ideas off of one another. I often found walking someone through a problem I was facing prompted a solution to pop into my own mind.

Other than that, look after your well being first and foremost. It is not the end of the world if things don't go to plan.

5.6 Acknowledgements

The support and friendship of my fellow students is something I will always remember. The camaraderie can bring you through the tough days, and was vital for my survival this year. Speaking to you about my progress brought fresh inspiration at times when I needed it most. I would particularly like to thank Ger Dobbs, who has been both a worthy adversary and a valued friend. I would also like to tip my hat to Mark Kelly, David Kelly and Ronan Timmons, who have all taken the time to listen to my ideas and provide valued feedback.

All of the lecturer's from first year to this year have my respect, and I would like to thank you all for sharing your knowledge and enthusiasm.

Dr. Joseph Kehoe, where do I start. You have my thanks for listening to us mere mortals, and taking into consideration the pressure we are all under. Fair play.

Finally I would like to say a huge thank you to Paul Barry, my supervisor. You first inspired me in third year when you turned me from the dark side and I made the switch to Mac OS, the best OS on the planet. Since then I took further inspiration and have adopted Python as my language of choice. Making yourself available over Christmas, Easter, and at all other times, night or day, has not gone unnoticed by any one of us students or, I'm sure, any of your colleagues in the department. Thanks again boss, couldn't have done it without you.

References

- [Raspberry Pi image 2.2.1] Wikipedia(n.d.). Raspberry Pi [online], available: https://en.wikipedia.org/wiki/Raspberry_Pi [accessed 15 Apr, 2018].
- [Motion Sensor image 2.2.1] Oregon State University(n.d.). Pyroelectric Infrared IR PIR Motion Sensor Detector Module HC-SR501 [online], available: http://eecs.oregonstate.edu/tekbots/modules/pir [accessed 15 Apr, 2018].
- [Camera image 2.2.1] Dexter Industries(2018). Raspberry Pi Camera [online], available: https://www.dexterindustries.com/shop/raspberry-pi-camera/ [accessed 15 Apr, 2018].
- [Relay image 2.2.1] Jaycar Electronics(2018). Arduino Compatable 5V Relay Board[online], available: https://www.jaycar.co.nz/arduino-compatible-5v-relay-board/p/XC4419 [accessed 15 Apr, 2018].
- [Audio Jack image 2.2.1] RaspberryPi.org (2018). Getting Started with the Raspberry Pi[online], available:https://projects.raspberrypi.org/en/projects/raspberry-pi-getting-started/3 [accessed 15 Apr, 2018].
- [Solenoid Door Lock image 2.2.1] Amazon(2018). Solenoid Lock[online], available:https://www.amazon.in/E-RACHIT-Solenoid-lock/dp/B01N4C16VH [accessed 15 Apr, 2018].
- [Chrome/HTML5 image 2.2.1] The Drum(10th Dec, 2016). Flash Loses out to HTML5 with Launch of Chrome 55[online], available: http://www.thedrum.com/news/2016/12/10/flash-loses-out-html5-with-launch-chrome-55 [accessed 15 Apr, 2018].
- [DDNS image 2.2.1] YouTube(5th Mar, 2015). How To Use DynDNS Service For Remote Access Freely [online], available: https://www.youtube.com/watch?v=TRj60_4qcvU [accessed 15 Apr, 2018].
- [Twilio image 2.2.1] Mautic(2018). Twilio SMS[online], available: https://www.mautic.org/mixin/twilio-sms-responder/ [accessed 15 Apr, 2018].
- [Flask image 2.2.1] Unix Stickers(2018). Flask Logo Badge Sticker[online], available: http://www.unixstickers.com/stickers/coding_stickers/flask-python-framework-badge-sticker [accessed 15 Apr, 2018].