

National College of Ireland
BSc in Computing – Internet of Things
2017/2018

Ryan Bannon
14488478
x14488478@student.ncirl.ie



VanWatch

Final Report

Date: 13/05/2018

Table of Contents

Executive Summary.....	3
1 Introduction	4
1.1 Background	4
1.2 Aims	5
1.3 Technologies	5
2 System	10
2.1 Requirements	10
2.1.1 User requirements	11
2.1.2 Functional requirements	12
2.1.3 Data requirements	20
2.1.4 Environmental requirements	20
2.1.5 Usability requirements	20
2.2 Design and Architecture	21
2.3 Graphical User Interface (GUI) Layout	22
2.4 Implementation	24
2.4.1 Python Implementation	24
2.4.2 Android Implementation	34
2.5 Testing	43
2.6 Customer testing	45
3 Conclusions.....	50
3.1 Product comparison table.....	50
3.2 Further development or research	51
3.3 Final thoughts.....	52
4 References.....	53
5 Appendix	55
5.1 Project Proposal	55
5.2 Project Plan	59

Executive Summary

Developing a surveillance system is always something I thought would be extremely fun, challenging and useful. As a result of the immense increase in popularity of the Raspberry Pi and the area of the “Internet of Things” in the world of I.T, this has now become a much more achievable thing to do. However, there are so many surveillance systems currently out there, such as home surveillance, store/ shop security systems and many more. So why would this be a good idea for me to consider for my final year project? The reason for this is, I believe I have found a niche in this market. Van theft has been around for a long time, however, it appears to be somewhat out of hand in Ireland at this moment in time.

With the use of a Raspberry Pi and other hardware components and software services, I will develop a system that will hopefully prevent this crime. What it will guarantee however, is that should the owner of a van with this system installed fall victim to theft, they will have a multitude of tricks up their sleeve to catch the culprit red handed. The system will sense the movement of the criminal when they break-in and record their actions. All of this footage will be safely stored and easily retrievable. Also, many other features will help increase the efficiency and effect of the system.

With this in mind, and a system such as this in play, I believe that it will hugely decrease the chance of people attempting this crime.

1 Introduction

1.1 Background

Break-ins to vans are extremely common in Ireland today. And unfortunately, at the forefront of these robberies are mainly tradesmen. Highly expensive tools and products are stored there, so they can work and make a living. The design and objectives of a van is too carry items in it, to many destinations. Sadly, they are seen as an easy target for some that think they can grab some extra cash by breaking in and stealing what is not theirs.

This project idea came to my mind because quite recently, a relative of mine had fallen victim to this crime and was then left largely out of pocket because thousands of euro worth of tools and appliances had been stolen from them. This experience is the reason I feel strongly about this topic, because I know I have the capability to help prevent such crimes from happening to others.

From my research, it is clear that this is an issue that has to be addressed. I happened to come across a number of different sources that reported this crime to be out of hand at this moment in time. With a quick search on Facebook, I also found that there are Facebook pages for people to post their personal situation in regard to robberies, in the hope that someone who sees it may know or have seen something useful to them. The worst part about this is, there is not a whole lot the Gardai can do. They can ask questions, take statements and search through CCTV footage, if there is any, but they cannot do much more.

The following statement from a report on Independent.ie news, states:

“Tools worth around €26,000 were found during a checkpoint on Stockhole Lane, Cloughran, Co Dublin in May.” [1]

I hate to say, but this figure suggests that these criminals are continuing with these acts because there is in fact great money involved if you are good enough at committing the offenses. This then also raises the question of whether people are claiming on their insurance when they have been robbed, because it may then increase the price of their premium. This is a terrible situation that some people have been put in, and the rate in which they are occurring does not seem to be going down anytime soon. This is why I propose that a system such as mine may benefit the hard-working man.

1.2 Aims

My aim coming into this project initially was to build a project based around the area of the “Internet of Things”. It was key that when I was coming up with my idea, that I found something challenging and fun. Otherwise, I knew that I would not enjoy my project, which is a recipe for disaster. Then after a long time of hard thought, I finally found a problem that I can offer a solution for. This idea is also one that is challenging and I am extremely passionate and enthusiastic about, so I know that I will thoroughly enjoy creating it.

My next aim was to then run this idea by my supervisor and confirm that it has the potential to be my final year project, and that it meets all criteria or requirement standards necessary. To my delight, I discovered that this in fact did just that and as a result, my next aim was to research in depth the market that this system will be entering.

My overall aim of the system itself is that it helps bring the number of van thefts in Ireland down by proving that the completed system can not only carry out all of the features that I will discuss in this paper, but also be a tool for catching criminals in the act.

1.3 Technologies

Raspberry Pi

A Raspberry Pi is a small credit card sized mini-computer. [2] It has great ability to interface with many components, and as a result of its size, is very portable. The newest model, Raspberry Pi 3 Model B, is the one that I will be using in this project.



Python Programming Language

The scripts that will be in control of recording the footage of the crime and all other features that will be running on the Raspberry Pi, will be programmed with the Python programming language. I have found this language to be wonderful for connecting Raspberry Pi projects to the real world and one that is extremely powerful to program with.



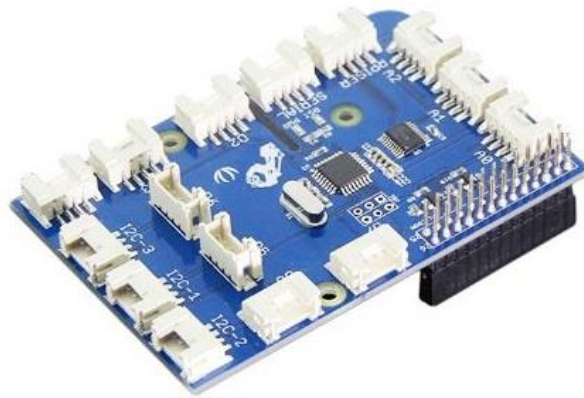
Java in Android Studio

An Android application will provide the user with an interface to view and alter data captured by the observing raspberry pi on the vans environment.



GrovePi

GrovePi+ is a board that is also the same size as a credit card. [3] It is an add-on board, which essentially means that it sits on top of the Raspberry Pi. The two boards are connected through the GPIO pins and the GrovePi+ offers an easy and flexible solution for interfacing sensors with the Raspberry Pi. In my project, it will be used to easily connect and disconnect the Passive Infrared (PIR) motion sensor [4] to and from the Pi. As a result, there is no soldering or breadboards necessary to interface this sensor, which is a huge bonus.



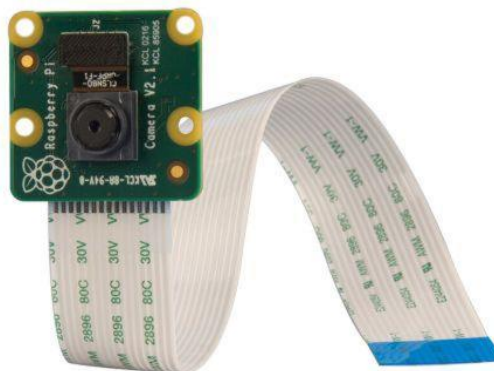
Motion Detector

The PIR motion detector will connect to the GrovePi+ and will detect any movement that is in its range. This sensor will detect the motion of any intruders.



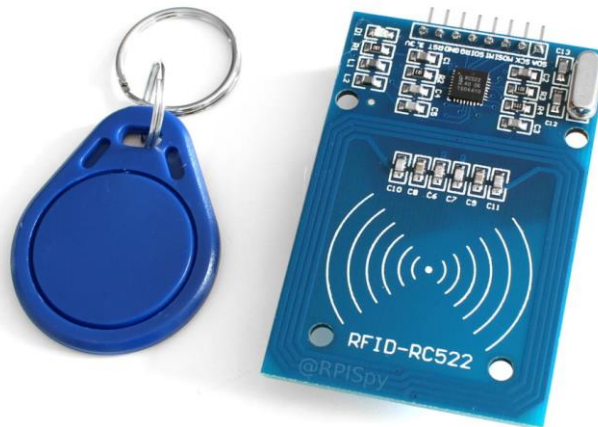
Pi Camera

This is the Raspberry Pi camera module which has the ability to record and take pictures like a normal camera does. This camera captures an impressive 8 mega pixels (MP). This will be used in my project to record any intruders and for observing the van through a live stream.



RFID RC522 receiver

Radio Frequency Identification (RFID) uses electromagnetic fields to automatically identify and track tags attached to certain objects. In this system, the RFID reader (receiver) will be connected to the Raspberry Pi. The RFID tag will be attached to the user's keys for their vehicle. If the reader receives a signal from a tag, the system will turn off. I will be using the RC522 receiver alongside an RFID tag(s).



Adafruit Ultimate GPS

The ultimate GPS offers a solution for gathering geospatial data by communicating with at least three satellites that are orbiting earth approximately 20,000 kilometers above ground. [5] And the antennae on this device is smaller than a cola-cola bottle cap, so as you can imagine it is a delicate piece of hardware. With this in mind, the ideal location for this once the system was installed to a vehicle, would be on the roof in a casing that would prevent damage as it needs a good clear look at the sky to get any sort of a signal back.



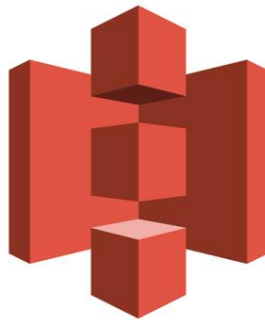
Google Firebase

I will make great use of three sections from within the Google Firebase Cloud Service, the Real-time database, Storage and Authentication.



AWS S3 bucket

I will push all recordings to an Amazon Web Services S3 bucket. This will be a backup cloud platform containing all recordings to fall back on, should data be lost from the Google storage platform for any reason whatsoever.



One final piece of technology that is worth mentioning is GitHub. This version control system will be used to ensure I am always working on the most recent version of the code I write throughout this project. It is also important because it means a copy of all of my files will be stored in the cloud, just in case anything happens to my own machine or it was damaged beyond repair, I know a simple “git pull” command will retrieve everything.

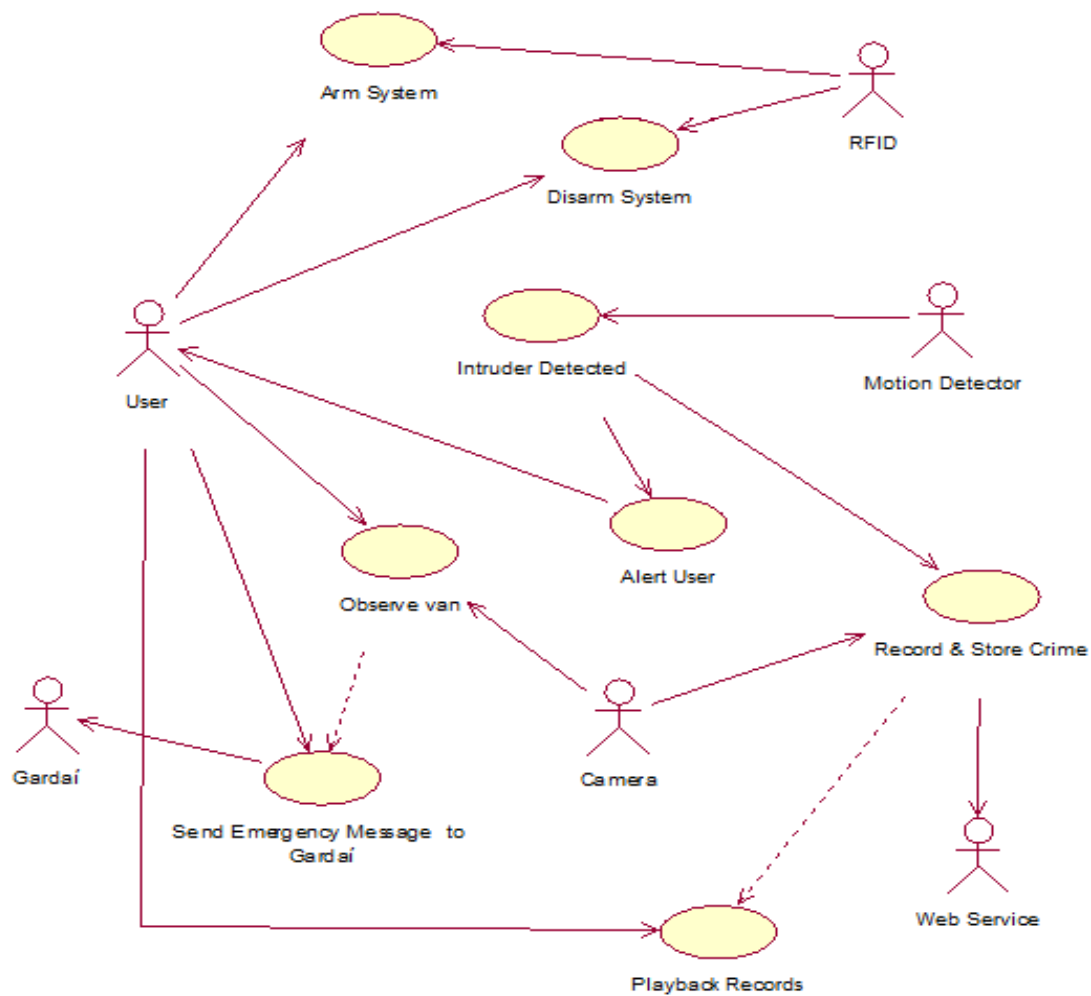
VanWatch GitHub Repo: <https://github.com/ryanbannon/VanWatch-SoftwareProject>

2 System

2.1 Requirements

The below requirements are functional features that this system must employ in order for the user requirements to be met. Any features after these, are extra functions that are integrated to further improve the system. I found that eliciting these requirements to be a very enjoyable process.

The Use Case Diagram below will give a broad visual of these requirements and how they will work together to create my desired system.



2.1.1 User requirements

Performance/Response time requirement

The performance and response time of this system will need to be extremely good. As I will be creating a solution for a crime that can be done in a very short space of time, this systems response time in particular will need to be of a high standard. If it is not, the camera may not capture footage of the heist or the user could not be alerted of the theft, and much more. These are all situations and outcomes that must be nullified, in order for this project to have the effect it can.

Security requirement

This project's main goal is to provide security for its users. They should be able to rest assured, knowing that their vehicle and all items inside of it, are secure. In terms of security to the systems software, the Pi will be connected to a private network and all data that will be transferring to the internet will be stored in Google Firebase, a secure cloud service.

Reliability requirement

Reliability is possibly the most crucial entity within this project, as if this project is not reliable, then what is the point in it? If the surveillance footage of a vehicle only works sometimes, or if the motion detector doesn't always work, this project will become a lot less desirable to people. It must be systematically sound, robust and durable. Should it tick all of these boxes, it will certainly be reliable too.

2.1.2 Functional requirements

Requirement 1 System Armed

Description & Priority

The priority of this use case will be to ensure that the system is activated and ready to detect and react to any suspicious activity in the owner's van.

Use Case

System Armed

Scope

The scope of this use case is to provide the functionality for the system, that can arm the security surveillance of the van.

Flow Description

Precondition

The system awaits a loss of connection between the RFID tag and receiver.

Activation

This use case starts when either the user manually activates the system, or if the user is out of range of the vehicle.

Main flow

The system will recognise that the RFID tag attached to the user's keys is out of range of the receiver.

The system will be armed, which means that it will be on the lookout for any unwarranted behaviour.

Any activity found as a result of this use case will initiate the Intruder Detected Use Case.

Alternate flow

The user manually arms the system through the controlling application.

Exceptional flow

Error may occur due to lack of the internet connection or RFID tag/ receiver fails.

Termination

The system terminates this use case when the user manually disarms the system through the application, or if the RFID tag comes within range of the receiver and vehicle.

Post condition

The system awaits a loss of connection between the RFID tag and receiver.

Requirement 2 Intruder Detected**Description & Priority**

The priority of this use case will be to ensure that the system is capable of detecting movement within the van when it is armed.

Use Case

Intruder Detected

Scope

The scope of this use case is to provide the system with the ability to detect an intruder to one's van with the use of a motion detector.

Flow Description**Precondition**

The system must be armed in order for this use case to be able to perform.

Activation

This use case will immediately be activated as soon as the motion detector realises there is movement within its range.

Main flow

The system is armed and waiting to detect any activity.

The motion detector recognises movement.

The Record & Store Crime Use Case will be initiated.

The Alert User Use Case will be initiated.

Alternate flow

The motion detector does not pick up any activity.

Exceptional flow

Error may occur due to a failure of a hardware component.

Termination

The system terminates this use case when the motion detector realises there is no more movement within its range.

Post condition

The system awaits a detection of any movement once again.

Requirement 3 Record & Store Crime**Description & Priority**

The priority of this use case will be to ensure that the system records any thefts and stores these recordings in the cloud.

Use Case

Record Crime

Scope

The scope of this use case is to provide the system with the ability to capture the event with a camera and store the recordings in a service in the cloud.

Flow Description**Precondition**

The system must sense an intruder.

Activation

This use case will immediately be activated as soon as the motion detector realises there is movement within its range.

Main flow

The system is armed and waiting to detect any activity.

The motion detector recognises movement.

The camera records the footage.

The Alert User Use Case will be initiated.

Alternate flow

The motion detector does not pick up any activity.

Exceptional flow

Error may occur due to a failure of a hardware component.

Termination

The system terminates this use case when the motion detector realises there is no more movement within its range.

Post condition

The system stores this recording in the cloud and awaits a detection of any movement once again.

Requirement 4 Alert Owner**Description & Priority**

The priority of this use case will be to ensure that the owner of the vehicle that the system is installed into, is notified of any intruders.

Use Case

Alert Owner

Scope

The scope of this use case is to provide the system with the ability to send an SMS message to the owner, making them aware of the intrusion.

Flow Description**Precondition**

The system must sense an intruder.

Activation

This use case will immediately be activated as soon as the motion detector realises there is movement within its range.

Main flow

The system is armed and waiting to detect any activity.

The motion detector recognises movement.

The camera records the footage.

The alert message is sent to the user.

Alternate flow

The motion detector does not pick up any activity.

Exceptional flow

Error may occur due to a failure of a hardware component.

Termination

The system terminates this use case when the alert message has been sent.

Post condition

The system carries on recording the heist.

Requirement 5 System Disarmed**Description & Priority**

The priority of this use case will be to ensure that the system is deactivated when necessary.

Use Case

System Disarmed

Scope

The scope of this use case is to provide the functionality for the system, that can disarm the security surveillance of the van.

Flow Description**Precondition**

The system awaits communication between the RFID tag and receiver.

Activation

This use case starts when either the user manually deactivates the system, or if the user comes within range of the vehicle.

Main flow

The system will recognise that the RFID tag attached to the user's keys come within range of the receiver.

The system will be disarmed, as the owner is now in a close enough proximity to notice and react should a theft be attempted.

The system awaits a loss of connection between the RFID tag and receiver.

Alternate flow

The user manually disarms the system through the controlling application.

Exceptional flow

Error may occur due to lack of the internet connection or RFID tag/ receiver fails.

Termination

The system terminates this use case when the user manually arms the system through the application, or if the RFID tag goes out of range of the receiver and vehicle.

Post condition

The system awaits communication between the RFID tag and receiver.

Requirement 6 Observe the van**Description & Priority**

The priority of this use case will be to ensure that the user can observe the van when they wish.

Use Case

Observe the van

Scope

The scope of this use case is to provide the functionality for the system to display the current state of the inside of the van, when the user requests to do so through the application.

Flow Description**Precondition**

The system and all hardware components must be in full working order.

Activation

This use case starts when user requests to view the inside of the van through the application.

Main flow

The user opens the application on a device.

They choose to view current footage of the inside of the vehicle.

The camera records the footage.

Playback is displayed to the user

Exceptional flow

Error may occur due to a failure of a hardware component.

Termination

The system terminates this use case when the user longer wishes to observe their vehicle.

Post condition

The application and system continues as normal.

Requirement 7 Playback a record

Description & Priority

The priority of this use case will be to ensure that the user can watch any previous recordings, if there happens to be any.

Use Case

Playback a record

Scope

The scope of this use case is to provide the functionality for the system to display any recordings that may have been previously stored.

Flow Description

Precondition

The system and all hardware components must be in full working order.

Activation

This use case starts when user requests to view previous recordings.

Main flow

The user opens the application on a device.

They choose to view previous recordings.

Playback is displayed to the user

Exceptional flow

Error may occur due to a failure of a hardware component.

Termination

The system terminates this use case when the user longer wishes to watch any recordings.

Post condition

The application and system continues as normal.

Requirement 8 Send emergency message to Gardaí

Description & Priority

The priority of this use case will be to ensure that the user has the ability to quickly contact An Garda Síochána, should they need to.

Use Case

Send emergency message to Gardaí.

Scope

The scope of this use case is to provide the functionality for the user to contact the Gardaí in the case of an emergency. Having a form of “Panic Button” such as this, will save time if a theft does occur.

Flow Description**Precondition**

The user must notice that there is an intruder in their vehicle.

Activation

This use case starts in the event of an emergency and when the user selects the option to contact the Gardaí on the application.

Main flow

The system is armed and happens to detect intruder activity.

The system alerts the user.

The user responds to the crime by selecting the contact emergency services button on the application.

Alternate flow

The motion detector does not pick up any activity.

Exceptional flow

Error may occur due to a failure of a hardware component.

Termination

This use case is terminated when the Gardaí have been contacted in relation to the break-in.

Post condition

The application and system continues as normal.

2.1.3 Data requirements

The data requirements of this project will consist of collecting the correct data that will be captured from the sensor nodes and distributing the results in a precise and accurate fashion to the appropriate platforms or services. Also, data that must be retrieved in the systems setup, for instance, name and phone number of the user and other similar information.

2.1.4 Environmental requirements

As my system will be focused on a specific space in a vehicle there are many environmental requirements I must factor in for this project. The objective of a vehicle is to move from one place to another, meaning its environment is constantly changing. Also as people put and take large or heavy items into and out of the van, the pi and sensors are vulnerable to being damaged should they be hit. Therefore, carefully designing the case that the unit can slot into and be protected with during rough journeys and avoiding damage from being knocked, is an important requirement to bare in mind.

As the pi will be placed in a vehicle, it will be limited to its power supply. My favoured solution for this would be to provide power through the vehicles cigarette lighter socket using a USB adapter. However, it is imperative that the adapter has an output current of around 2 Amps. The reason I state this is, all of the components that I will need to use to build this system will exceed 1.5 Amps. The voltage of the Pi is 5 volts, which is a popular output range for many adapters, so ensuring that I reach the 2 Amp target I have set myself, will mean that I will not run into any difficulties supplying power to the system.

2.1.5 Usability requirements

Portability requirement

The portability of this system will be provided to the user in the form of an application. A unit will have to be fitted into their vehicle, therefore this element of the system cannot be portable, but the application on the other hand will be.

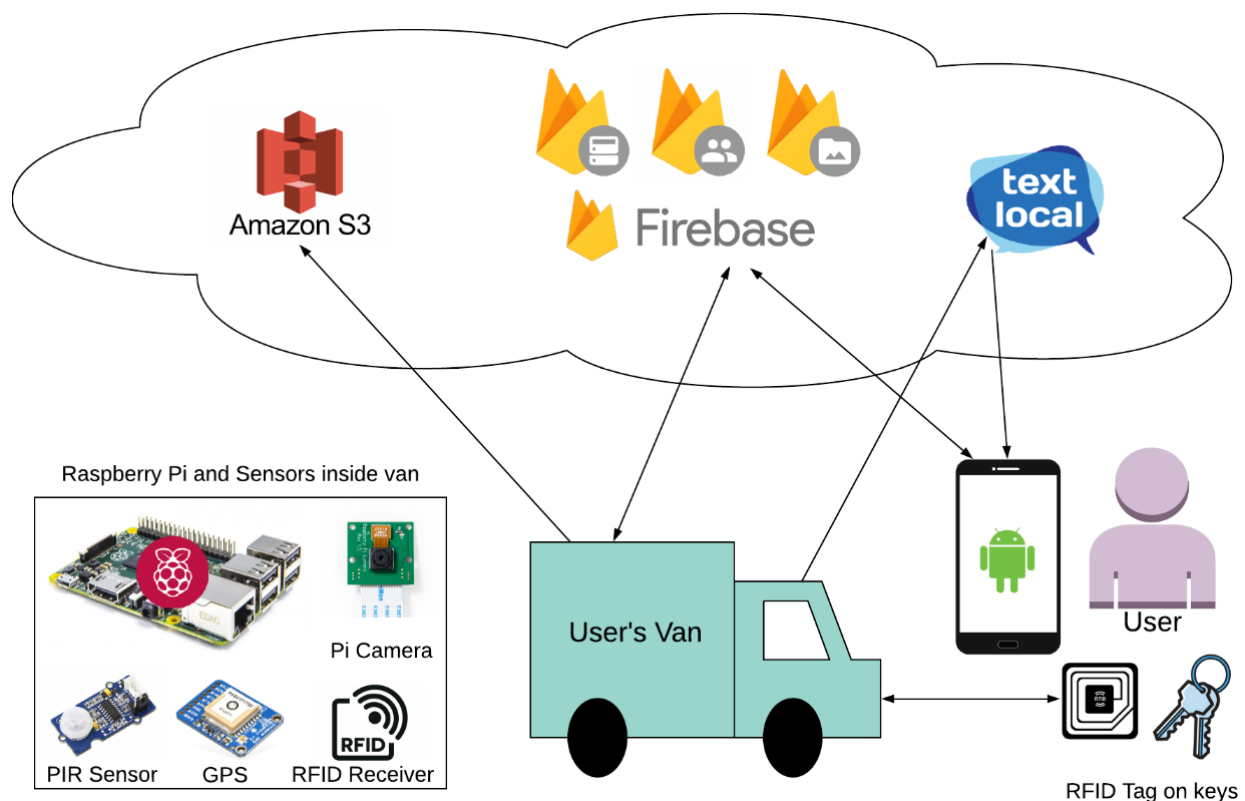
Extendability requirement

This system will be developed with extendability and scalability in mind. There are many other features that could potentially be added to the project, therefore providing the capacity to extend and grow is a requirement that I feel is extremely important to take into consideration. This is a prime example of why I believe using the GrovePi+ in the project will help, because should I want to attach more sensors further down the line, I can do so with great ease.

Resource utilization requirement

The utilization of all resources will be key in creating this system. From the user's perspective, having the capacity to control the unit from an application, to the sensors acting accordingly to its environment, all resources will play a critical part in combining these components into one solid and reliable system.

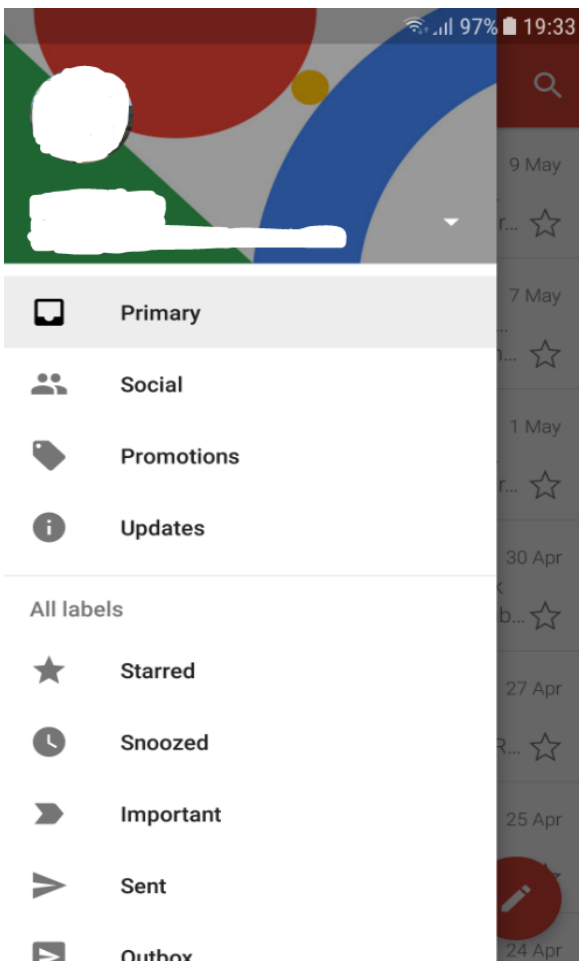
2.2 Design and Architecture



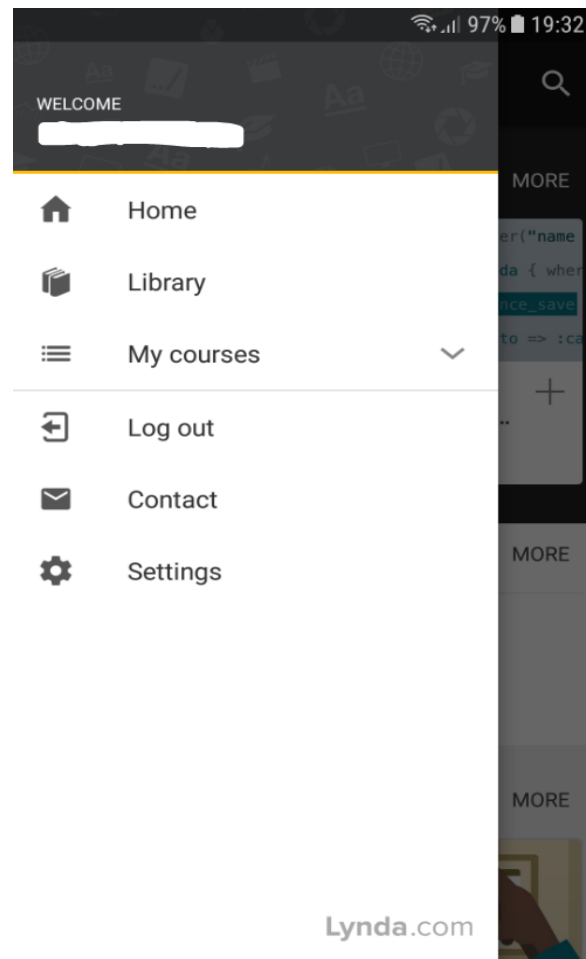
2.3 Graphical User Interface (GUI) Layout

The design and UI of the app is something that I gave a great deal of thought and consideration. I found that after testing many different themes within Android Studio, I was drawn to a consistent colour scheme of orange and blue. I felt that these colours complimented each other nicely and therefore, I implemented this into my UI. I also took inspiration from other apps I had on my phone and after spotting a design that is present in two frequently used apps of mine, Gmail & Lynda.com, I knew that it was a design I wanted to implement into my project – the Navigation Drawer.

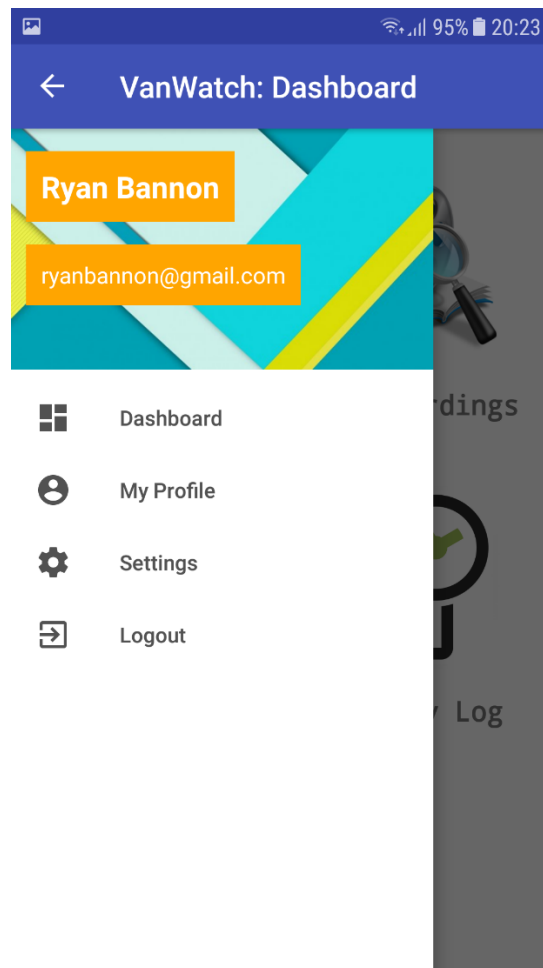
Gmail:



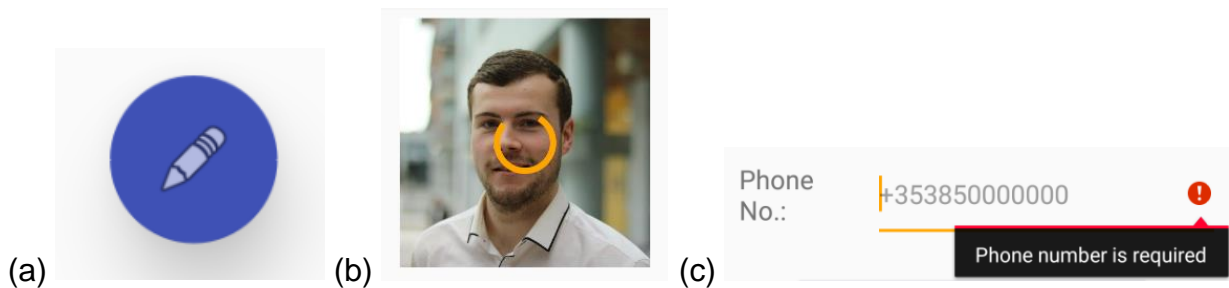
Lynda.com



VanWatch's navigation drawer:



I also wanted to add other attractive design elements to the application, for instance the floating action button (a), spinning progress bar (b), edit text error handlers (c) and more, which were all achieved.



2.4 Implementation

2.4.1 Python Implementation

To begin explaining the implementation of the features of this project, I will first explain in detail what resources they need and what it consists of. Firstly, I'll speak about the Python scripts that were developed to enable the raspberry pi unit with the ability to act and behave as the intended security system should. Then I will move on and discuss the Android application that I developed for this project. All code snippets only consist of the most important elements of each file.

Sensor_scripts.py

I began by implementing the passive infrared (PIR) motion sensor, as if you strip the whole idea back, it is this functionality that truly drives this project and ultimately the most crucial part of the system – because it will detect moving objects that emit heat energy, or in this case people. This PIR sensor was easily connected to the raspberry pi through the Grovepi+ HAT (Hardware Attached on Top) that sits above the pi.

```
previousState = False
motionState = False

grovepi.pinMode(pir_sensor,"INPUT")

previousState = motionState
motionState = grovepi.digitalRead(pir_sensor)

if motionState != previousState:
    newState = "HIGH" if motionState else "LOW"
    if motionState:
        print ('Motion Detected')
        *** Execute security functions ***
    else:
        *** Motion has stopped - Execute clean-up functions ***
```

The above snippet demonstrates how I managed to program the pir sensor to react when it senses movement in its environment. I initialised two variables called previousState and motionState to false. This tells python that these variables will be of Boolean data type and therefore will only ever have a value of either True or False. Then I set the pin mode of the pir sensor to INPUT. Next, I set the previousState equal to motionState because as we iterate through a while True (never ending) loop, the previousState will be updated to the value that was stored in motionState during the

previous iteration. The motionState variable is set to the value that is observed from the pir sensor. So, at this stage we have our two variables, previousState which contains the value of the previous iteration, and motionState which contains the current value of the sensor. Next, I defined an if else statement which will only be entered should a condition be met. This condition is if motionState is not equal to previousState, or in other words, motionState = True and previousState = False. These values of True and False are represented by the digital signal that is passed back from the sensor, "HIGH" being a large signal when infrared radiation is found (True), and "LOW" being a small signal when there is no motion in the sensors path (False). With this in mind, I created another if else statement. It is also important to bare in mind that Python will not reach this statement unless it passes through the first if else statement that determines whether there was movement found or not. This if statement tells Python what to do when the digital signal is "HIGH" which will be to execute all of the security functions e.g. start recording the crime. The else statement following this will be reached when the sensor finds a "LOW" signal, meaning that the infrared heat can no longer be detected. When this occurs, we want Python to execute all clean-up functions such as stop recording the crime etc. This snippet contains a powerful yet efficient way off utilising the passive infrared sensor and as I have previously mentioned, contains essential functionality in order for this system to work.

The next part of the system will talk about the camera that is connected to the pi with a flex ribbon cable through the camera port.

```
camera = PiCamera()

def getFileName():
    return datetime.now().strftime("%Y-%m-%d_%H.%M.%S.h264")

if motionState:
    filename = getFileName()
    camera.start_recording(filename)
    print('Starting Recording of '+filename)

else:
    camera.stop_recording()
    print('Stopped Recording')
```

The above snippet demonstrates how I managed to use the Pi Camera module to record videos. First, I initialised the camera by creating a variable that holds an

instance of the PiCamera module. [6] Then if we jump into the if statement that I discussed previously which tells Python that there has been motion detected, I create another variable, called filename, that calls the getFilename() method. This method simply uses the datetime module to create a string of the current time (.now()) in the format (Year-Month-Day_Hour.Minute.Second) and then finally adds (.h264) at the end. The pi camera records files in the .h264 format, therefore the .h264 at the end of the string represents the extension that the file being created will need. I also chose to format the current time all the way down to the second and name the file after this timestamp for the simple reason of uniqueness. Next, I called the .start_recording() method on the camera and passed in the filename so Python knows what to name the file once it is created. As the name of the method suggests, this started a recording through the camera. Then when we reach the else statement when there is no longer any movement detected by the sensor, the stop_recording() method is called which stops the recording.

The next step was to ensure the user of the system was immediately notified when the sensors detected movement. This was achieved through the TextLocal API once I created an account. The send SMS API documentation for the Python language can be found and was obtained through the following link:

<http://api.txtlocal.com/docs/sendsms>. [7] There were a few changes I had to make in order for this to work however. I am running Python 2.7 from my pi and the documentation was specifically for Python 3.0. As a result, libraries and modules that are used for sending an SMS were altered between the versions.

```
import urllib
import urllib2

def sendSMS(apikey, numbers, sender, message):
    data = urllib.urlencode({'apikey': apikey, 'numbers': numbers,
                            'message' : message, 'sender': sender})
    data = data.encode('utf-8')
    request = urllib2.Request("http://api.txtlocal.com/send/?")
    f = urllib2.urlopen(request, data)
    fr = f.read()
    return(fr)

if motionState:
    number = firebase.get('/number', 'number')
    response = sendSMS('o5ZZfZRb04UwWpJkJJJDw3owPAZmPsKL
jJNCfJHTI', number, 'VanWatch', 'EMERGENCY ALERT: Our system has
detected movement in your van!')
    print (response)
```

The above snippet is also inside the if statement when movement is found by the pir sensor. Both imports are the required libraries that are needed to complete the SMS action. Now it is stating that before you do anything at all such as start recording, send an SMS to the owner of the vehicle first. This is the first time we have come across firebase in these code snippets. The reason I am retrieving a value from firebase is because it is the cloud platform that stores the users phone numbers. Therefore, I am getting the users phone number from firebase and passing it and the rest of the parameters (apikey, sender, message) into the sendSMS() method so it can be sent. Finally, I am printing the response which will state whether the message was sent successfully or not. Below is a screenshot taken on my phone after testing this functionality.



Moving on, I am going to explain the process in which I designed to store all of the videos that are recorded by the pi in my chosen cloud service, Google's Firebase as previously stated. In order to do this, I had to utilise two sections within Firebase, the Real-time database and Storage.

```

from google.cloud import storage
from firebase.firebase import FirebaseApplication

firebase = FirebaseApplication()
storage_client = storage.Client.from_service_account_json()
bucket = storage_client.get_bucket()

if motionState:
else:
    blob = bucket.blob(os.path.basename(filename))
    blob.upload_from_filename(filename)
    photo_url = bucket.get_blob(filename)
    url = photo_url.public_url
    post=firebase.post('/videos/',{'start':start,'end':end,'URL':url})
    print (post)

```

The above snippet imports the firebase real-time database and storage python modules, so we can use them in our script. I then initialised an instance of them both (firebase & bucket). This functionality if we take a step back and think about what we want to achieve, needs to be executed when the camera has recorded a video. Therefore, I put this in the else section that will run when the sensor has received a low signal after previously receiving a high one. We know that this will then stop the recording but now we want to push this up to the cloud. I did this by specifying the path to the file through its filename and transformed it to a Blob object. A blob object is simply an immutable object of an array of bytes. [8] Next, I called the .upload_from_filename() method and passed in the file which sends this blob object to my firebase bucket. Once this has been achieved, I then retrieve the url for this specific blob object and store it in a variable called url. The final thing to do now is to push this url, alongside the start and end times of the video that are obtained through datetime module we have seen before, to a table in the real-time database. This functionality was developed using the firebase .post() method which take in the parameters of the table “videos”, followed by the data in JSON format, that it is to store “{'start':start,'end':end,'URL':url}”. The reason for this data to be sent to the real-time database will become clearer as we proceed through this implementation chapter of the document, but ultimately it is done so I can gather and display this data in the Android application that will be discussed further down the line.

I also thought it would be a good idea to store a backup copy of each recording in a separate cloud platform altogether. The reason for this is in the highly unlikely event where data is erased, there will always be a backup to revert back to if needed. The platform I chose for this is another extremely secure cloud service, Amazon Web Services.

```
import boto3
s3 = boto3.resource('s3',aws_access_key_id=ACCESS_KEY_ID,
aws_secret_access_key=ACCESS_SECRET_KEY)
if motionState:
else:
    data = open(filename, 'rb')
    key = str('VanWatch_Recordings/'+filename)
    s3.Bucket(BUCKET_NAME).put_object(Key=key, Body=data)
    print ("Object Uploaded to S3 bucket as a backup")
```

The snippet above provides the solution I implemented to push each recording up to Amazon Web Services, the service in particular is an S3 bucket. The library that I used to make this submission to S3 was boto3. Boto3 contains a number of resources that are well documented, to make working with Amazon S3 buckets quite seamless. [9]

The final part of the sensor_scripts.py file that is important to detail is the activity log. This is another table in the firebase real-time database that stores information of actions that were executed.

```
from firebase.firebase import FirebaseApplication

firebase = FirebaseApplication()

if motionState:
    date = getDate()
    log = firebase.post('/activitylog/', {'date':date,
    'message':'Motion has been detected in the vehicle!'})
else:
    date = getDate()
    log = firebase.post('/activitylog/', {'date':date,
    'message':'A new recording has been stored in the database'})
```

The above snippet uses methods we have already seen in this implementation section. Again, I am using the datetime module to get the date and then posting a message to the “activitylog” table in the real-time database when these actions occur.

Gps_location.py

This file is the one that utilises the functions made available by the GPS Breakout module. The gps was connected to the pi through a USB to TTL serial cable which converts transistor to transistor logic (logic high ‘1’ & logic low ‘0’) communication at a serial level (one bit at a time) in a way that the receiving device (the pi) can read through its USB port. [10] The end of the cable that connects to the gps consists of four wires, red (power), black (ground), white (USB input) and green (USB output). The wires needed to be attached to their appropriate header pins, which I first had to solder to the gps module.

```
while True:
    try:
        track = firebase.get('/coordinates', 'track')
        if track == "true":
            findCoordinates()
        time.sleep(10)
```

The above snippet is a never ending while True loop that executes whenever the file is run. It simply gets the value of the track row in the coordinates table. Then if the value is equal to true, it will execute the findCoordinates() method that I will speak about next. If the value is anything else, then Python will sleep for 10 seconds and then loop through this process again and again.

```
def findCoordinates():
    while True:
        try:
            track = firebase.get('/coordinates', 'track')
            if track == "false":
                break
            if report['class'] == "TPV":
                if hasattr(report, 'lat' and 'lon'):
                    latitude = report.lat
                    longitude = report.lon
                    put = firebase.patch('coordinates',
                                         {'latitude':latitude, 'longitude':longitude})
            time.sleep(10)
```

As mentioned, the above snippet is of the findCoordinates() method that will be executed when the track value is equal to “true”. Again, I created another while True loop that also gets the value of the track row in the coordinates table, the reason for this is as we iterate through this over and over, the value could change to NOT “true” e.g “false”, and therefore I have created an if statement that will handle this event by breaking out of the loop and returning to the first one that waits until the value is “true”. The action that is to be taken when it is in fact “true” though, is to receive the gps report. The initialising and generating of a report was documented at the following link: <https://learn.adafruit.com/adafruit-ultimate-gps-on-the-raspberry-pi?view=all>. [11] If the report we receive is a ‘TPV’ report, Python will then check if the report has ‘lat’ and ‘lon’ attributes. If it does, I created variables latitude and longitude that can store this data from the report and then finally push this data to firebase. I opted for using the .patch() for inserting these values because I want the values to be constantly overwritten instead of storing rows upon rows of coordinates. Then this process will be repeated again after 10 seconds.

Live_stream.py

This file contains a linux shell command that is to be executed when the script is run. This file handles the live streaming feature of the system. I discovered that for what I was aiming to achieve, it was in fact most efficient for me to set up a live stream from my pi to my Youtube channel. [12] This was made possible by enabling live streams on my Youtube account and acquiring an RTMP address, which is essentially a specific media URL. I also noted my secret-key that was assigned to my account, this key ensures no one else can stream to my channel. Lastly, I installed ‘Libav’ which, “provides cross-platform tools and libraries to convert, manipulate and stream a wide range of multimedia formats and protocols”. [13] The most important library from Libav being avconv. [14]

```
import os

def stream():
    cmd = 'raspivid -o - -t 0 -hf -fps 30 -b 6000000 | avconv -re -ar 44100 -ac 2 -acodec pcm_s16le -f s16le -ac 2 -i /dev/zero -f h264 -i - -vcodec copy -acodec aac -ab 128k -g 50 -strict experimental -f flv rtmp://a.rtmp.youtube.com/live2/[secret-key]
    os.system(cmd)
```

Write.py and Read.py

These two files manage the reading and writing of data from the RFID receiver to the tag. Before I began, some soldering was required for this component just like the gps module. This RFID component however, requires 7 pins to be connected to the pi's GPIO (General Purpose Input/ Output) pins. These pins are the SDA (Serial data signal), SCK (Serial clock), MOSI (Master out slave in), MISO (Master in slave out), GND (Ground), RST (Reset-circuit) and 3.3v (3.3v power). Once the pins were soldered onto the RFID reader, I connected them to the pins on the pi with female-to-female jumper wires. Moving on from the hardware, I imported two Python files to help with the development of the scripts for reading and writing to the tag(s). These files can be found at the following link: <https://github.com/pimylifeup/MFRC522-python> [15]. These files were made available from PiMyLifeUp.com. At this stage I was now ready to begin utilising this RFID component into my project. Taking a quick look at what I wanted to achieve using RFID is, the user has an RFID tag on their keys, if the reader picks up a signal, this means that the owner is in close proximity of the vehicle, so I want the system to be disabled. If no such signal is found, then the system will activate in the event that a criminal attempts a break-in.

```
reader = SimpleMFRC522.SimpleMFRC522()
try:
    text = raw_input('Your Name:')
    print("Now place your tag to write")
    reader.write(text)
    print("Written")
    id, text = reader.read()
    post = firebase.post('/tags/', {'id':id, 'name':text})
    print (post)
```

Starting with the write.py file, I specified that the object reader is an instance of SimpleMFRC522() which offers simpler control over the RFID reader in the snippet above. Then I tell Python to accept an input from the shell and store that as a variable called text. Now we are waiting to execute the reader.write(text) line, which cannot write to a tag unless there is one within range. As soon as the reader picks up a tag, it writes this text to it. After that, I then retrieve the id and the text that was just set by calling reader.read(). The values are stored in the id and text variables that I then send up to the firebase table "tags".


```

import signal
signal.signal(signal.SIGALRM, timeout_handler)
def checkForTag():
    while True:
        try:
            signal.alarm(60)
            id, text = reader.read()
            put = firebase.patch('system', {'state':'off'})
            time.sleep(60)
        except TimeoutException:
            put = firebase.patch('system', {'state':'on'})
            checkForTag()

```

The read.py was slightly tougher because I wanted to check for a tag, which means using the .read() method and if one is found, do something, if not, do something else. But the issue being, once the .read() is called, Python will not proceed any further until a tag is in fact read from. So to overcome this, I discovered a library called signal [16] and this library contains a class called TimeoutException and a method called .alarm(). So, in my while True loop inside the checkForTag() method, I set an alarm to trigger a signal after 60 seconds. This signal will then raise a TimeoutException. So, specifying an except for this Exception allowed me to tell Python what to do when this happens. My final result calls the read() method and if a tag is found, then I push the state to Firebase as “off”. If a tag is not found in 60 seconds, the exception will be raised and a state of “on” is pushed to Firebase followed by a call to execute checkForTag() again and again.

2.4.2 Android Implementation

This section describes the Android application I created in the Java language for this project. This was developed in Android Studio, and for visual aid and greater understanding of the Java code snippets in the following section, I mutated the code slightly to fit in the boxes and remove the bulky lines that come with Java such as, `.addOnCompleteListener()` etc. In other words, the code snippets are not copied to this document verbatim from the actual files.

MainActivity.java:

This java class is as the name suggests, the main activity. It implements the navigation drawer which has a huge effect on how a user navigates through the app. However, the implementation of this design was not as straight forward as one would hope. The standard android java class would render in a layout view after calling the `setContentView()` method in the `onCreate()` stage of the android lifecycle. My navigation drawer would consist of 4 elements, Dashboard, My Profile, Settings and Logout. In order for the navigation drawer to work, each of these elements classes had to extend the fragment class in order to inherit from it, and I then created a switch statement that controlled which fragment was to be passed in when it was clicked. I set the Dashboard fragment as the default, which means that this is the class that will be executed when the user logs into the application. Also, the header of the navigation drawer has a colourful image in the background where the logged in user's name and email address values are set to text views after being retrieved from the firebase real-time database.

LoginActivity.java

I decided right from the off that I wanted to implement authentication in my app, and the best possible way for me to do so was also using Firebase. This time however, using the Authentication portion of the cloud service. I was also keen on using this solution because it provides consistency throughout the project and Firebase is natively friendly to implement directly from Android Studio.

```

private FirebaseAuth mAuth;

protected void onCreate(Bundle savedInstanceState) {
    mAuth = FirebaseAuth.getInstance();
}

private void userLogin(){
    mAuth.signInWithEmailAndPassword(email,password) {
        Intent loginIntent =new Intent(LoginActivity.this, MainActivity.class);
        startActivity(loginIntent);
    }
}

public void onClick(R.id.buttonLogin){
    userLogin();
}

protected void onStart() {
    if(mAuth.getCurrentUser() != null){
        finish();
        startActivity(new Intent(LoginActivity.this, MainActivity.class));
    }
}

```

First, I declared and initialised the mAuth object which is an instance of the FirebaseAuth object. Then I have set an onClick method for the login button which will initiate the userLogin() method. The userLogin method calls the signInWithEmailAndPassword() method that firebase provides us with. This takes two parameters, email and password, which are retrieved from the edit text views. If the user provides the correct credentials, an intent is made to bring the user into MainActivity of the app. As the LoginActivity is set as the launcher activity, every time the app icon is pressed on the device, the user will be brought to this class. So, I have set an if statement in the onStart() activity method which will check if there is currently a user object in the application – meaning a user has logged in but not logged out. If that is true, then finish() the LoginActivity and bring the user straight to the MainActivity. This helps with user satisfaction because it means that they do not have to log in every time they open the app if they choose to remain logged in.

SignupActivity.java:

```
private FirebaseAuth mAuth;

protected void onCreate(Bundle savedInstanceState) {
    mAuth = FirebaseAuth.getInstance();
}
private void registerUser (){
    mAuth.createUserWithEmailAndPassword(email,password) {
        Intent loginIntent =new Intent(LoginActivity.this, MainActivity.class);
        startActivity(loginIntent);
    }
}
```

The snippet above is how a user can create an account on the app through firebase authentication. It is very similar to LoginActivity.java however, we are calling the createUserWithEmailAndPassword() method inside the registerUser() method when the sign up button is pressed.

Dashboard.java:

```
view.findViewById(R.id.liveFootageBtn).setOnClickListener(new
View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent footageIntent = new Intent(getContext(), LiveFootage.class);
        startActivity(footageIntent);
    }
});
```

This class simply contained four intent actions that start other activities when the buttons are pressed.

Profile.java:

This class displays the logged in user's information – their profile picture, name, email and phone number.

```

user = mAuth.getCurrentUser();
database = FirebaseDatabase.getInstance();
databaseReference = database.getReferenceFromUrl("URL")+user.getUid());

private void loadUserInformation() {
Picasso.with(getContext()).load(user.getPhotoUrl().toString()).into(profileIm
ageView);
name.setText(user.getDisplayName());
databaseReference.child("email").addValueEventListener(new
ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        if (dataSnapshot.getValue() != null) {
            String mail = dataSnapshot.getValue(String.class);
            email.setText(mail);
        }
    }
}
}

```

In the snippet above, I first initialised my user object as the current user and my firebase database reference is targeting the table that is associated with that user's id that will be used further down in this paragraph. Next, I used Picasso, an image downloading and caching library for android [17], to slide the image into an image view in the loadUserInformation() method. I did this by grabbing the URL of the users profile picture using the user.getPhotoUrl() method and then setting this .toString() to avoid any unwarranted errors occurring. Next, I loaded the user.getDisplayName() that is associated with the users profile image, into the a name textview. The next two details (email and phone number) had to be stored separately as there was no such option to add them to firebase alongside the profile image and display name. As such, I targeted a table that is specific to that exact user by stating "databaseReference = database.getReferenceFromUrl("URL")+user.getUid());". Then I turn the focus to the child of this table (email) and then later (phone number). However, for now I will speak about the email row. Firebase offers a nice listener (.addValueEventListener()) which works well with the onDataChange() method sitting inside the listener. What this does is monitors the targeted table and if there is a change, it will execute whatever commands are set. In this case I called the setText() method on the email Textview object to display the result. Once this value is changed in the database, it will also be changed in the application in real-time. Finally, I repeated the same process for the user's phone number (this is not shown in the snippet above).

EditProfile.java:

This class displays the logged in user's information – their name, email and phone number in edit text views and the user's profile picture. The user can edit their details and can also select their profile picture image to open their phones gallery. Once they select another photo, it will be uploaded to firebase and updated as their profile picture. This class is naturally very similar to Profile.java. I have set that all the values of name, email and phone number are displayed as text and the user can simply change them before pressing submit. The submit button works by having an `onClickListener()` that runs a `saveToFirebase()` method when called. `saveToFirebase()` then amends any changes in the database. The only part of this class that differ a great deal from the Profile.java class is that I had to create image view that will update the user's profile image.

This was trickier than the other elements in this class to implement because selecting the image would initiate an intent that then accesses the devices gallery. Once an image has then been selected by the user, I had to call an `onActivityResult()` method that first validates that the user selected an image, second that the image was stored in a `Bitmap` variable and displayed back to the user in the image view, and lastly that this image was then submitted to firebase as that current users profile picture.

```
private void uploadImageToFirebaseStorage() {
    firebaseProfileRef=FirebaseStorage.getInstance().getReference("ProfilePics/"
+ System.currentTimeMillis() + ".jpg");
    firebaseProfileRef.putFile(imageUri)...{
        UserProfileChangeRequest profile = new
        UserProfileChangeRequest.Builder()
            .setPhotoUri(taskSnapshot.getDownloadUrl())
            .build();
    }}
}}
```

In the snippet above, these lines are the most crucial for submitting this image as the user's profile picture, but in particular `".setPhotoUri(taskSnapshot.getDownloadUrl())"`. This essentially states that when you are pushing this file up, retrieve the download URL of this image and then set it as the `photoUri` associated with this user. This is important because this is what gives us the ability to call Picasso on the `.getPhotoUrl()` – which is essentially the get method on the URL that was set by the `.setPhotoUri()` method.

LiveFootage.java:

As we saw earlier in the Python implementation, I set up a live stream from my pi to Youtube. Now there is no point in doing this unless I create section in my application where the user can view this stream, and this class handles just that. This was not a difficult thing to achieve with a webview in android.

```
protected void onCreate(Bundle savedInstanceState) {
    String url = "https://www.youtube.com/live_dashboard";
    WebView webView;
    webView = findViewById(R.id.webview);
    webView.setWebViewClient(new WebViewClient());
    webView.loadUrl(url);
    webView.setVisibility(View.VISIBLE);
    emergencyBtn = findViewById(R.id.emergencyBtn);
    emergencyBtn.setOnClickListener() {
        String phone = "999";
        Intent callIntent = new Intent(Intent.ACTION_DIAL,
            Uri.fromParts("tel", phone, null));
        startActivity(callIntent);
    });
}
```

The snippet above demonstrates how I managed to load the live feed url into the webview. It also contains the `onClickListener()` that I set in the case of an emergency. If the user is watching the live stream and notices criminal activity such as a break-in, they can quickly press the “Contact Emergency Services” button which will invoke an intent and pass the phone number “999” to the device’s dial service where they can call An Garda Siochana immediately.

Location.java:

When this class is created it renders a Google Map object as a fragment into its layout.

```
protected void onCreate(Bundle savedInstanceState) {
    database = FirebaseDatabase.getInstance();
    databaseReference = database.getReferenceFromUrl("/coordinates/");
    databaseReference.child("track").setValue("true");
    databaseReference.addValueEventListener... onDataChange() {
        Double latitude = dataSnapshot.child("latitude").getValue();
        Double longitude = dataSnapshot.child("longitude").getValue();
        LatLng location = new LatLng(latitude, longitude);
        mMap.addMarker(position(location));
        mMap.moveCamera(CameraUpdateFactory.newLatLng(location));
    }}}

public void onStop(){
    super.onStop();
    databaseReference.child("track").setValue("false");
}
```

The above snippet tells Java what I want it to do when the Google Map has been successfully passed into the layout. I call my firebase reference to the coordinates table and set the track value to true. If you remember, this is the row Python is checking in the gps_location.py file. This will initiate Python to read the gps and submit the coordinates back to this same table. In the mean-time the android application will get the values of these latitude and longitude coordinates and add a marker and move the camera to the position of these coordinates. This is also placed inside an onDataChange() listener, so every time Python updates the coordinates, the marker and camera will be readjusted. Finally, I specified in all of the stages of the android lifecycle what the track value should be, for example in the snippet we can see that I tell android if the activity is stopped, to set the value to false. This means that Python will only read the gps and update the latitude and longitude coordinates when the user is actually inside the Location.java activity.

Recordings.java & ActivityLog.java:

I will speak about these two class in parallel as they are both very similar in nature. Both of these classes are serving as an interface in the application as a means of displaying the recordings taken from the pi and the activity log that is generated when an event is triggered also on the pi. I set a listview as the target object for both of these classes to display their information. I will speak about the Recordings.java class but at the same time, everything will I detail will also apply to ActivityLog.java unless I say otherwise.

```
listView = findViewById(R.id.listview);
recording = new Recording();
database = FirebaseDatabase.getInstance();
databaseReference = database.getReferenceFromUrl("/videos/");
list = new ArrayList<>();
adapter = new ArrayAdapter<String>(this,R.layout.recording,R.id.video, list);
databaseReference... onDataChange(DataSnapshot dataSnapshot) {
    list.clear();
    for(DataSnapshot ds: dataSnapshot.getChildren()){
        recording = ds.getValue(Recording.class);
        list.add(recording.getStart().toString()
+recording.getEnd().toString());
        listView...onItemClick(...) {
            Intent i = new Intent(Intent.ACTION_VIEW);
            i.setData(Uri.parse(recording.getURL().toString()));
            startActivity(i);
        }
    });
    listView.setAdapter(adapter);
}
```

At the top of the snippet above, I am initialising my listview and database reference objects. I also create a new instance of an ArrayList called list, an instance of an ArrayAdapter called adapter which takes in the textview layout and ArrayList, and an instance of my model class Recording. My model class Recording consists of three declared strings, two constructors – one empty and another that initialises the arguments that are passed into it, in my case (start & end, which are strings representing the start and end times of a recording), and getters and setters for each variable. The model is in place to ensure that the data being received from firebase is parsed into a structure that the application can then use. Next, I created another onDataChange() method. Inside of this I cleared the list, then I created a for loop that iterates through the children of the table videos in firebase. Each iteration adds the values of the start and end strings to the ArrayList and then I set the adapter to the

listview, which displays each child of the videos table into listview. Going back to when I cleared the ArrayList, I did this because every time that the database is changed, Java will loop through the it and add all of the rows to the list. If this list is not cleared before that happens, then it will result in these rows just being added to the bottom of the current list, thus duplicating data again and again. As I mentioned, this is similar to the ActivityLog.java class, as it also displays a listview in the same manner and also has its own model MyEventLog.java. However, there is one thing that does set these classes apart, which is present in the Recordings.java class. During the for loop in the onDataChange method, I set an .onItemClickListener() which gets the value of the recording url from firebase that was previously submitted by Python in the sensor_scripts.py file when the motion sensor triggered and a recording was taken and pushed to storage. I also wrote an intent within this onItemClickListener which takes the user to this url that was received, thus downloading the video to the user's device for them to view.

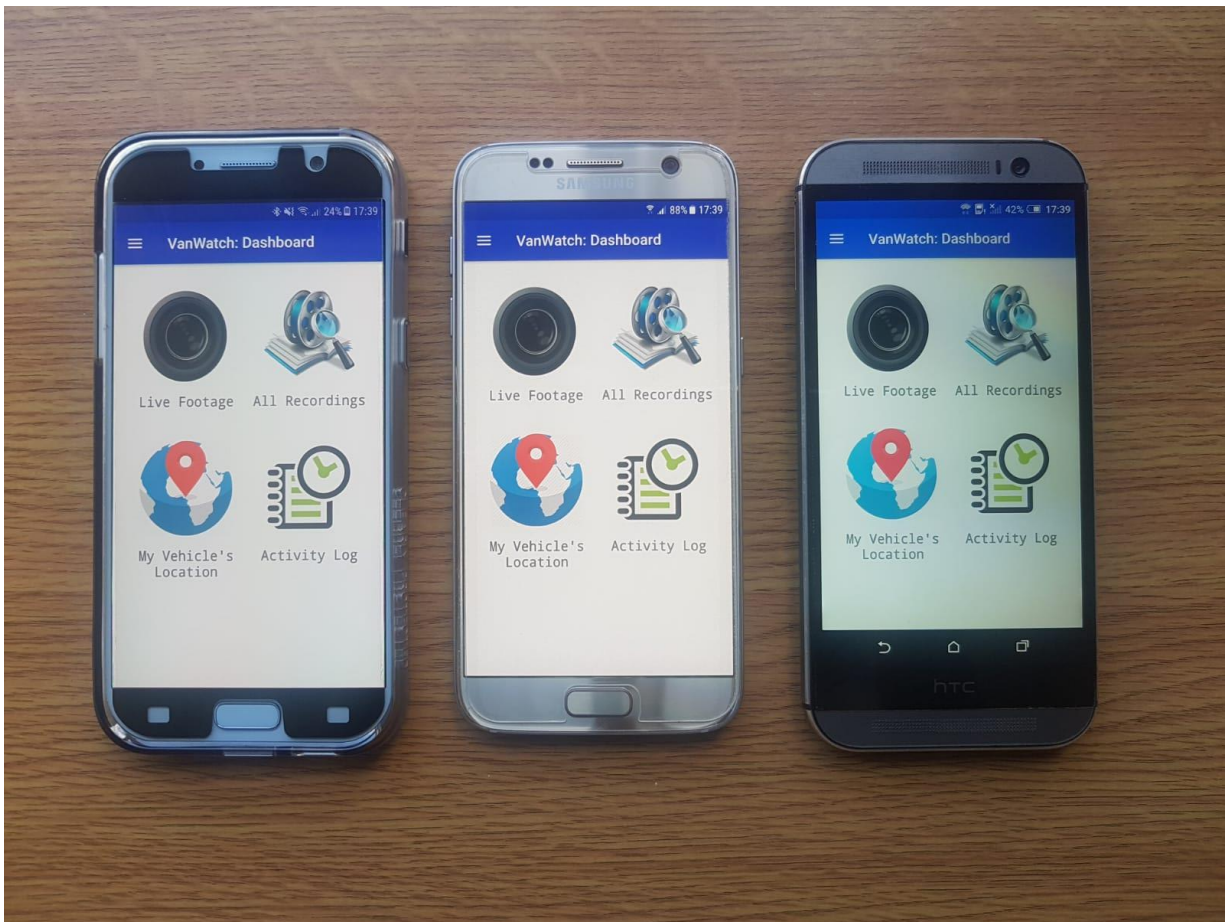
An attempt was made to display a list of recordings stored in the Amazon S3 bucket, with the intention to have a choice and test which platform (Firebase or AWS) performed better. However, after implementing Firebase first and spending unnecessary time, I moved on to other features of the app. I have left these attempts in my project which can be found in the following files:

- AWSManager.java
- DownloadedObject.java
- GenerateKeys.java
- S3ListAdapter

2.5 Testing

The testing of this project was something that I enjoyed because I got to see the system in action as a whole. Numerous unit tests were run throughout the whole development stage of the project but the one part of it that I felt had been tested the least is the Android application. So, in order to rectify this, I installed the app onto four different Android phones and tested each of the apps features on all of them. The phones that I did this on were:

- Samsung Galaxy A5 (2017) – Android 7.0
- Samsung Galaxy S7 – Android 7.0
- HTC One M8 – Android 6.0



These devices are in the same order (from left to right) as specified in the list above this photo.

The result of this testing went extremely smooth and every element of the application passed with flying colours, even on the HTC One M8 device which was running an Android version below the device I had been unit testing with throughout the whole development cycle of the app (the Samsung Galaxy A5 2017).

I also decided that testing the system as a whole, fully implemented was something that would enable me to conclude with a degree of certainty that the project was developed well and all elements were implemented correctly. I did this by arranging people to participate in testing the system in an environment I had set up. I re-enacted a scenario in which would trigger the system to respond. Each participant also had access to the application and when they were finished, they filled out a survey. The feedback I received from this was amazing and really worth-while. All of the survey results will be in the next section of this report “Customer testing”.

2.6 Customer testing

VanWatch

Consent to take part in the following research study.

- I Linda..... voluntarily agree to participate in this study.
- I understand that even if I agree to participate now, I can withdraw at any time or refuse to answer any questions without any consequence of any kind.
- I understand that no such personal data of myself will be collected or held by the study conductor.
- I have had the nature of the study explained to me and I am confident that I understand everything I need to know before beginning this study.
- I understand that I will not benefit directly from participating in this research.
- I understand all information that is gathered will be kept anonymous and treated confidentially.
- I understand that the conductor of this research will comply with NCI's Data Retention Policy.
- I understand that under the freedom of information legalisation I am entitled to access the information I have provided at any time while it is in storage in NCI.
- Finally, I understand that I am free to contact anyone involved in this research to seek further clarification and information.

Signature of researcher: Ryan Bannon Date: 12/05/18
Signature of participant: L. Moore Date: 12/5/18

Survey Questions:

1. On a scale of 1-10 how easy was it to navigate through the overall application?
2. On a scale of 1-10 how easy was it to create an account and login?
3. On a scale of 1-10 how nice did you find the design and user interface of the app?
4. On a scale of 1-10 how effective did you find the Live Footage feature?
5. On a scale of 1-10 how effective did you find the Contact Emergency Services button?
6. On a scale of 1-10 how effective did you find the list of All Recordings feature?
7. On a scale of 1-10 how effective did you find the Vehicle's Location feature?
8. On a scale of 1-10 how effective did you find the response from the system when a criminal event was acted out?
9. On a scale 1-10 how effective did you find the SMS messaging service to your mobile?
10. On a scale 1-10 how much would you rely on this unit to deliver this service in the real world?
11. Any additional comments

10
10
10
10
10
10
10
10
10
10
10

WELL DONE RYAN GREAT PROJECT.
THE 'IDEA' IS EXCELLENT!

VanWatch

Consent to take part in the following research study.

- I ...Peter..... voluntarily agree to participate in this study.
- I understand that even if I agree to participate now, I can withdraw at any time or refuse to answer any questions without any consequence of any kind.
- I understand that no such personal data of myself will be collected or held by the study conductor.
- I have had the nature of the study explained to me and I am confident that I understand everything I need to know before beginning this study.
- I understand that I will not benefit directly from participating in this research.
- I understand all information that is gathered will be kept anonymous and treated confidentially.
- I understand that the conductor of this research will comply with NCI's Data Retention Policy.
- I understand that under the freedom of information legalisation I am entitled to access the information I have provided at any time while it is in storage in NCI.
- Finally, I understand that I am free to contact anyone involved in this research to seek further clarification and information.

Signature of researcher: Ryan Bannon Date: 13/05/18

Signature of participant: Peter Kelly Date: 13/5/18

Survey Questions:

1. On a scale of 1-10 how easy was it to navigate through the overall application?

10

2. On a scale of 1-10 how easy was it to create an account and login?

10

3. On a scale of 1-10 how nice did you find the design and user interface of the app?

10

4. On a scale of 1-10 how effective did you find the Live Footage feature?

10

5. On a scale of 1-10 how effective did you find the Contact Emergency Services button?

10

6. On a scale of 1-10 how effective did you find the list of All Recordings feature?

10

7. On a scale of 1-10 how effective did you find the Vehicle's Location feature?

10

8. On a scale of 1-10 how effective did you find the response from the system when a criminal event was acted out?

10

9. On a scale 1-10 how effective did you find the SMS messaging service to your mobile?

10

10. On a scale 1-10 how much would you rely on this unit to deliver this service in the real world?

10

11. Any additional comments

Great idea and definitely something a lot of people need. Brilliant work Ryan, well done.

VanWatch

Consent to take part in the following research study.

- I ANDREA... voluntarily agree to participate in this study.
- I understand that even if I agree to participate now, I can withdraw at any time or refuse to answer any questions without any consequence of any kind.
- I understand that no such personal data of myself will be collected or held by the study conductor.
- I have had the nature of the study explained to me and I am confident that I understand everything I need to know before beginning this study.
- I understand that I will not benefit directly from participating in this research.
- I understand all information that is gathered will be kept anonymous and treated confidentially.
- I understand that the conductor of this research will comply with NCI's Data Retention Policy.
- I understand that under the freedom of information legalisation I am entitled to access the information I have provided at any time while it is in storage in NCI.
- Finally, I understand that I am free to contact anyone involved in this research to seek further clarification and information.

Signature of researcher: Ryan Barron Date: 13/05/18

Signature of participant: ANDREA BYRNE Date: 13-05-18

Survey Questions:

1. On a scale of 1-10 how easy was it to navigate through the overall application?
2. On a scale of 1-10 how easy was it to create an account and login?
3. On a scale of 1-10 how nice did you find the design and user interface of the app?
4. On a scale of 1-10 how effective did you find the Live Footage feature?
5. On a scale of 1-10 how effective did you find the Contact Emergency Services button?
6. On a scale of 1-10 how effective did you find the list of All Recordings feature?
7. On a scale of 1-10 how effective did you find the Vehicle's Location feature?
8. On a scale of 1-10 how effective did you find the response from the system when a criminal event was acted out?
9. On a scale 1-10 how effective did you find the SMS messaging service to your mobile?
10. On a scale 1-10 how much would you rely on this unit to deliver this service in the real world?
11. Any additional comments

10
10
10
10
10
10
10
10
10
10

really cool app Ryan. Definitely something a lot of kids can do with. The logo is really cool too! Good job

VanWatch

Consent to take part in the following research study.

- I PARIS voluntarily agree to participate in this study.
- I understand that even if I agree to participate now, I can withdraw at any time or refuse to answer any questions without any consequence of any kind.
- I understand that no such personal data of myself will be collected or held by the study conductor.
- I have had the nature of the study explained to me and I am confident that I understand everything I need to know before beginning this study.
- I understand that I will not benefit directly from participating in this research.
- I understand all information that is gathered will be kept anonymous and treated confidentially.
- I understand that the conductor of this research will comply with NCI's Data Retention Policy.
- I understand that under the freedom of information legalisation I am entitled to access the information I have provided at any time while it is in storage in NCI.
- Finally, I understand that I am free to contact anyone involved in this research to seek further clarification and information.

Signature of researcher: Ryan Bannon Date: 12/05/18

Signature of participant: Paris Moon Date: 12-05-2018

Survey Questions:

1. On a scale of 1-10 how easy was it to navigate through the overall application?

10

2. On a scale of 1-10 how easy was it to create an account and login?

10

3. On a scale of 1-10 how nice did you find the design and user interface of the app?

10

4. On a scale of 1-10 how effective did you find the Live Footage feature?

10

5. On a scale of 1-10 how effective did you find the Contact Emergency Services button?

10

6. On a scale of 1-10 how effective did you find the list of All Recordings feature?

10

7. On a scale of 1-10 how effective did you find the Vehicle's Location feature?

10

8. On a scale of 1-10 how effective did you find the response from the system when a criminal event was acted out?

10

9. On a scale 1-10 how effective did you find the SMS messaging service to your mobile?

10

10. On a scale 1-10 how much would you rely on this unit to deliver this service in the real world?

10

11. Any additional comments

excellent project.
Really innovative and I love the idea behind
the design.

VanWatch

Consent to take part in the following research study.

- I Anne..... voluntarily agree to participate in this study.
- I understand that even if I agree to participate now, I can withdraw at any time or refuse to answer any questions without any consequence of any kind.
- I understand that no such personal data of myself will be collected or held by the study conductor.
- I have had the nature of the study explained to me and I am confident that I understand everything I need to know before beginning this study.
- I understand that I will not benefit directly from participating in this research.
- I understand all information that is gathered will be kept anonymous and treated confidentially.
- I understand that the conductor of this research will comply with NCI's Data Retention Policy.
- I understand that under the freedom of information legalisation I am entitled to access the information I have provided at any time while it is in storage in NCI.
- Finally, I understand that I am free to contact anyone involved in this research to seek further clarification and information.

Signature of researcher: Ryan Bannon Date: 12/05/18

Signature of participant: Anne Murphy Date: 12/5/18

Survey Questions:

1. On a scale of 1-10 how easy was it to navigate through the overall application?

10

2. On a scale of 1-10 how easy was it to create an account and login?

10

3. On a scale of 1-10 how nice did you find the design and user interface of the app?

10

4. On a scale of 1-10 how effective did you find the Live Footage feature?

10

5. On a scale of 1-10 how effective did you find the Contact Emergency Services button?

10

6. On a scale of 1-10 how effective did you find the list of All Recordings feature?

10

7. On a scale of 1-10 how effective did you find the Vehicle's Location feature?

10

8. On a scale of 1-10 how effective did you find the response from the system when a criminal event was acted out?

10

9. On a scale 1-10 how effective did you find the SMS messaging service to your mobile?

10

10. On a scale 1-10 how much would you rely on this unit to deliver this service in the real world?

10





















11. Any additional comments

Great idea Ryan. Very innovative. You should try and implement this. Very valuable for Thamesmer. Well done.

3 Conclusions

3.1 Product comparison table

Whilst eliciting my requirements for this project, I also researched the market that this system would be entering. Upon this research I found one such company named Vanlock [18], that specialise in the area of van security. They are based in Dublin, so they would appear to be my main rival should I roll this system out into production. With that in mind, I decided that one of the ways I would like to conclude this document is with a visual comparison of the two products side-by-side.

Features	VanWatch	Vanlock
Security Van Locks		
Vehicle Compromisation Awareness		
Smart Activation/ Deactivation System		
Video Recording Technology		
Emergency Alert System		
Live Stream Technology		
GPS Tracking		
Recorded Event Log		
Website		
Mobile Application		

3.2 Further development or research

There are so many avenues that can be explored to increase the productivity and improve and add other features to this system. It is important to create a project that has the ability to expand, such as this one does.

A great feature I plan to integrate in the future would be a more stable arming and disarming process for the system. Although the one that I have laid out will suit my needs right now, I'm sure that going forward, any improvements to the architecture will always be welcomed. This is why I think having the capability to switch the system on and off when the vehicle owner locks and unlocks the van, would be something to consider.

To further valid a break-in and provide the ability for quicker reactions, contact plates could be installed onto the doors of the van. This would help if the contact plates were being tampered with if the intruder was attempting to force the doors of the van open, because we would know they are trying to get in before they manage to do so, which as I have mentioned means that we can react faster.

Another feature I believe will help the user is offering them the capability to attach RFID tags to their most precious and expensive tools or appliances. This way should someone manage to steal these items, they can be tracked and as a result the culprit can also be caught.

A NoIR camera offers the same functions as the regular Pi camera, with the only difference being, it does not use infrared filtering. Without this filtering, the camera has greater vision in the dark, but lesser vision in daylight. So, implementing a Pi NoIR camera to record a crime at night time, would also improve the system.

Lastly, one possible feature to implement in the future could be to allow the user to speak into their mobile device and output this audio through some sort of speaker in the van. This could work well should a thief break-in, because if they hear the vehicle owner speaking to them through a speaker and threatening to call the Gardai immediately etc., I would imagine that this would scare away most if not all of the criminals that attempt a break-in on a van with this system installed into it.

3.3 *Final thoughts*

Overall, this project was something that I thoroughly enjoyed creating. It was very satisfying to see the end product after all of the hours that was put into making it what it is. I'm thrilled to say that I believe the system has achieved more than I had anticipated at the beginning and I was even hesitant at the beginning with doubt over my ability to produce a project such as this. The reason being I aimed to challenge myself and incorporate a multitude of different technologies that were unfamiliar for me. However, looking back now I completely stand by my choice of direction and my idea for my final year project. I have designed this whole system in a way that is certainly flexible and above all scalable. The feedback I have received from those who were kind enough to participate in my testing stage was fantastic and the support I have received throughout this project from my supervisor, Dominic Carr and all staff in the National College of Ireland was tremendous and something I am very thankful for.

4 References

- [1] "Tool theft victims say gangs are 'ruining' their livelihoods - 'These gangsters are targeting hard-working people,'" *Independent.ie*. [Online]. Available: <https://www.independent.ie/irish-news/crime/tool-theft-victims-say-gangs-are-ruining-their-livelihoods-these-gangsters-are-targeting-hardworking-people-36102213.html>. [Accessed: 12-May-2018].
- [2] "Raspberry Pi - Teach, Learn, and Make with Raspberry Pi," *Raspberry Pi*. [Online]. Available: <https://www.raspberrypi.org/>. [Accessed: 12-May-2018].
- [3] "GrovePi Internet of Things Robot Kit." [Online]. Available: <https://www.dexterindustries.com/grovepi/>. [Accessed: 12-May-2018].
- [4] adafruit, "PIR Motion Sensor Tutorial," *Instructables.com*. [Online]. Available: <http://www.instructables.com/id/PIR-Motion-Sensor-Tutorial/>. [Accessed: 12-May-2018].
- [5] A. Industries, "Adafruit Ultimate GPS Breakout - 66 channel w/10 Hz updates." [Online]. Available: <https://www.adafruit.com/product/746>. [Accessed: 12-May-2018].
- [6] "10. API - picamera.camera Module — Picamera 1.10 documentation." [Online]. Available: http://picamera.readthedocs.io/en/release-1.10/api_camera.html. [Accessed: 13-May-2018].
- [7] "Sending SMS." [Online]. Available: <http://api.txtlocal.com/docs/sendsms>. [Accessed: 11-May-2018].
- [8] "Interface: Blob," *Firebase*. [Online]. Available: <https://firebase.google.com/docs/reference/js/firebase.firestore.Blob>. [Accessed: 11-May-2018].
- [9] "Amazon S3 — Boto 3 Docs 1.7.19 documentation." [Online]. Available: <https://boto3.readthedocs.io/en/latest/guide/migrations3.html>. [Accessed: 13-May-2018].
- [10] A. Industries, "USB to TTL Serial Cable - Debug / Console Cable for Raspberry Pi." [Online]. Available: <https://www.adafruit.com/product/954>. [Accessed: 11-May-2018].
- [11] "Introduction | Adafruit Ultimate GPS on the Raspberry Pi | Adafruit Learning System." [Online]. Available: <https://learn.adafruit.com/adafruit-ultimate-gps-on-the-raspberry-pi?view=all>. [Accessed: 11-May-2018].
- [12] "Live Stream to YouTube With a Raspberry Pi." [Online]. Available: <https://www.makeuseof.com/tag/live-stream-youtube-raspberry-pi/>. [Accessed: 11-May-2018].
- [13] "Libav." [Online]. Available: <https://www.libav.org/>. [Accessed: 11-May-2018].
- [14] "Libav documentation: avconv." [Online]. Available: <https://libav.org/avconv.html>. [Accessed: 11-May-2018].
- [15] *Contribute to MFRC522-python development by creating an account on GitHub*. Pi My Life Up, 2018.
- [16] "17.4. signal — Set handlers for asynchronous events — Python 2.7.15 documentation." [Online]. Available: <https://docs.python.org/2/library/signal.html>. [Accessed: 13-May-2018].
- [17] "Picasso." [Online]. Available: <http://square.github.io/picasso/>. [Accessed: 11-May-2018].

- [18] "Van Locks Dublin | Vanlock Supply and Installation Dublin," *Vanlock*. [Online]. Available: <http://www.vanlock.ie/>. [Accessed: 13-May-2018].

5 Appendix

5.1 Project Proposal

Objectives

For my final year project, I will create a system that offers a solution around the area of vehicle theft. But to be more specific, van theft. The word theft can be misunderstood as it refers to a couple of different instances. The theft of an automobile can mean that the culprit has stolen the vehicle, or it can also be taken that they have robbed some things from the vehicle. I want to focus primarily on the latter, the theft of items from within one's van. The main objective is to give people in danger of being victims to van theft, an element of security.

Another objective is also to try and prevent break-in cases. I believe that this is achievable because if a thief was aware that the van they were targeting was being monitored, they are far less likely to proceed with the heist. Unfortunately, this will not stop them from moving onto a van that doesn't have this equipped, however, should all vans be fitted with this technology, I believe there would be a decline in the number of thefts occurring.

This is why I propose that a system such as this, needs to be in place. With the emergence of the technologies surrounding the "Internet of Things", and how popular this field has now become, it is important that we use these technologies to our advantage. Especially when the solution can help with issues such as the one I have explained above.

The way I hope to achieve these objectives is to implement a surveillance system that detects break-ins. A camera will be fitted in the vehicle which will capture footage of the crime, and a mobile application will manage the system appropriately.

Motivation

My motivation for this project is simply through personal experience. A member of my family had fallen victim to this crime not too long ago. The person responsible left us devastated, by stealing thousands of euro worth of tools and appliances. For this reason, I feel strongly and am very passionate about preventing this from happening to others if possible. On that note, as I am specialising in the “Internet of Things” stream and developing first hand applications with the likes of Raspberry Pi’s, I know that it is in fact possible to offer this services to those who may in danger. This is why and how I have found my motivation to pursue this idea for my final year project.

Technical Approach

My approach coming into this project will be to clearly identify goals and targets to hit in certain timeframes. My project plan will include detail of the major and minor tasks that must be completed. I will also be sure to continuously test elements of the system, while working on the documentation of the project as I go along also.

I will continuously research the technologies and other styles that I can approach aspects of the project with. As I will be developing a surveillance system of vans, hardware components and accessories will be its focal point. I will use a Raspberry Pi and a number of different sensors and cameras to monitor the chosen environment. I must ensure that all of these components are working and capable of consuming the data that I require. With that in mind, I must also research the different type and models of vans. Gathering the dimensions, shapes, sizes and layout of completely different models of vans is very important because it affects the physical architecture in which these components are to be arranged and fitted. A unit that is universal to the type of van is what is required with this project.

I am responsible for implementing a system that will provide the best possible user experience for the end user. I believe that I can achieve this, once I can integrate all of the elements correctly. The hardware devices, for example, Raspberry Pi and sensors, must communicate with each other and other platforms. I would be heavily inclined to integrate the Amazon Web Services (AWS) platform and have the captured data, stored in a database in the cloud. Raspberry Pi’s have great capacity to communicate with platforms such as, AWS. Having the environment monitored and transmitting its data to the cloud isn’t enough though. I will need to develop a platform that the end user will be able to work with. This will mean some sort of application will have to be implemented. AWS will be the glue that holds the system together, because it will exchange data between the unit in the van and the application that the user will be operating. This application will ideally be mobile and therefore, I will create an Android app that should be able to control the system appropriately. Lastly, in order for the

system to be armed and disarmed automatically, I want to use Radio-Frequency Identification (RFID) tags and receivers, to switch the system on when the tag is out of range of the receiver and off when it is within range. The tag will be attached to the owner's keys, so when they are not near the van, the system will be on red alert for any suspicious activity, and won't when they are in close proximity.

Special Resources Required

In order for this project to work, I will need a great amount of hardware components. These components will all work together in a system that detects possible intruders to a defined area (the boot of a van) and acts appropriately when such actions are identified.

I should need the following components:

Raspberry Pi - The Raspberry Pi is a small computer that is around the size of a credit card and also looks similar to one too. Unlike a normal computer, it does not have things such as a keyboard etc, but these devices can be plugged into the Raspberry Pi. They are mainly used for building computer projects because they are small, cheap, portable and capable of doing almost anything with electronics. Fortunately, the college has supplied me with this hardware, as I am specialising in the "Internet of Things" stream.

GrovePi - The GrovePi is a similar device in shape and size to the Raspberry Pi and they are used together. The GrovePi makes it easy to attach and use sensors and other devices with the Raspberry Pi. Thankfully, the college has provided me with this piece of hardware also.

Motion Detector Camera - The motion detector camera will record footage when it senses movement in range.

Alarm - the alarm is a possible device that I could implement into the unit. It could be useful for people that do not have an alarm already fitted into their van.

RFID tag/ receiver - I will need Radio-Frequency Identification to initiate the system. Once the tag comes within range of the receiver, the system will know that the owner is close, therefore it should not attempt to detect unusual behaviour. On the other hand, when the tag is outside of the range, the system will be armed.

Technical Details

Android is one of the two dominant operating systems in the world, the other being iOS. As a result, Android has great demand in the market for mobile application development. This is why the application I will create, will be built with Java in the Android Studio Integrated Development Environment (IDE).

I will be using Python to write the scripts for sensor components attached to the Raspberry Pi. The principal Python modules I expect to use are ones such as, datetime, json and GPIO. PiCamera is another module that I suspect will be needed to interface the Pi with the motion detector camera.

Amazon Web Services offers a managed cloud platform specifically for the Internet of Things, which is fittingly called “AWS IoT”. By connecting the system to this platform, I can securely interact, store and exchange data between all of the required nodes in the system.









































GitHub is a cloud based version control service. I will use this throughout the development of my project, to ensure that none of the code will be lost at any stage. GitHub will also allow you to recover every version of your code, in the case that you make a change and then suddenly realise that it was a mistake and has interrupted other features of your project.

Evaluation

The evaluation and finishing of the project will be a huge task. Integration testing will be carried out to exercise the established interactions between different components of the system. This will examine the different parts of the system and how work together. This will outline any bugs in the software code that will need to be fixed, or show the need for any error handlers in certain areas etc. The system test will then evaluate the system as a whole. The main focus and most important result from these tests are user experience. In order to sign off on the project, I will be sure to experiment the system with people I know. There are plenty of people that I am in close contact with that own a van, so this should not be an issue.

This idea is one that I am very proud of, and it will be a great achievement that I will take with me when I graduate from college. Therefore, I look forward to starting to work on this project for my final year.

5.2 Project Plan

ID	Task Mode	Task Name	Duration	Start	Finish
1		Project Idea	11 days	Mon 18/09/17	Mon 02/10/17
2		Study Feasibility	11 days	Mon 18/09/17	Sun 01/10/17
3		Idea Approval	1 day	Mon 02/10/17	Mon 02/10/17
4		Project Proposal	19 days	Tue 03/10/17	Fri 27/10/17
5		Gather Data	5 days	Tue 03/10/17	Mon 09/10/17
6		Prepare & Upload Document	14 days	Tue 10/10/17	Fri 27/10/17
7		Requirement Specification	21 days	Sat 28/10/17	Fri 24/11/17
8		Preparation	10 days	Sat 28/10/17	Thu 09/11/17
9		Gather User Requirement	4 days	Sat 28/10/17	Wed 01/11/17
10		Gather Resources	3 days	Thu 02/11/17	Mon 06/11/17
11		Detail System Design	3 days	Tue 07/11/17	Thu 09/11/17
12		Gather Data	13 days	Sat 28/10/17	Tue 14/11/17
13		Prepare & Upload Document	8 days	Wed 15/11/17	Fri 24/11/17
14		Project Prototype	1 day	Sat 02/12/17	Sat 02/12/17
15		Mid Point Presentation	4 days	Mon 04/12/17	Thu 07/12/17
16		System Development	123 days	Mon 06/11/17	Wed 25/04/18
17		Initial Hardware	13 days	Mon 06/11/17	Wed 22/11/17
18		Configure Components	11 days	Mon 06/11/17	Sat 18/11/17
19		Unit Testing	4 days	Sun 19/11/17	Wed 22/11/17
20		Software	83 days	Thu 23/11/17	Mon 19/03/18
21		Hardware Components	29 days	Thu 23/11/17	Tue 02/01/18
22		Develop Scripts	26 days	Thu 23/11/17	Thu 28/12/17
23		Unit Testing	3 days	Fri 29/12/17	Tue 02/01/18
24		Cloud Platform Services	21 days	Wed 03/01/18	Wed 31/01/18
25		Establish Connection with	19 days	Wed 03/01/18	Sat 27/01/18
26		Unit testing	4 days	Sun 28/01/18	Wed 31/01/18
27		Build Mobile Application	33 days	Thu 01/02/18	Mon 19/03/18
28		UI Design	8 days	Thu 01/02/18	Mon 12/02/18
29		Development Function	23 days	Tue 13/02/18	Thu 15/03/18
30		Unit Testing	2 days	Fri 16/03/18	Mon 19/03/18
31		RFID Hardware Integration	19 days	Tue 20/03/18	Fri 13/04/18
32		Program Logic	15 days	Tue 20/03/18	Sun 08/04/18
33		Unit Testing	5 days	Mon 09/04/18	Fri 13/04/18
34		Implementation	5 days	Sat 14/04/18	Thu 19/04/18
35		Testing	4 days	Fri 20/04/18	Wed 25/04/18
36		Integration Testing	2 days	Fri 20/04/18	Mon 23/04/18
37		System Testing	2 days	Tue 24/04/18	Wed 25/04/18
38		Technical Report	47 days	Thu 01/03/18	Fri 04/05/18
39		Software & Documentation Up	1 day	Sat 05/05/18	Sat 05/05/18
40		Project Presentation	4 days	Wed 23/05/18	Sat 26/05/18

