Background

The background of my project is security. Given everyday circumstances that compromise the safety of individuals and families it is important to be able to see who is trying to come into your house, place of work or anywhere you choose to utilise a video doorbell. In today's world, where the risk of home invasions has increased, I propose a video doorbell to assist increase security around a household. Many families will want to have a more secure lifestyle – a prime method of this also is security cameras. However, security cameras can be expensive to purchase and install, so even though a video doorbell isn't the best alternative, it is still an adequate start and is still able to serve other purposes the security camera cannot.

Problem Statement

The problem is as mentioned security and feeling secure in your own household. A video doorbell would also be a great solution to incorporate into disability centres, homes, security corporations and defence companies. I aim to make this video doorbell accessible and we can even set the notifications up so that they are sent to two different email addresses, in the context of a disability patient to the patient themselves and their carer.

Materials:

The materials required for my project are:

Raspberry Pi:
-Raspberry Pi model 3B+
-Pi camera module
-PIR motion sensor
-jumper wires
-Keyboard
-Mouse
-Hdmi cable
-power source for raspberry pi

Particle argon:
-Particle Argon
-Breadboard
-MG995 Servo Motor
-Jumper wires
-Laptop

Requirements:

For the sensor to recognise movement and alert the user someone is at the door. The user is able to view the footage of who is at the door and can choose to automatically open the door or ignore.

Design Principles

While there are many video doorbells available in the market, my video doorbell design works by combining the Raspberry Pi (RPi) and Particle Argon. I use the PIR motion sensor as the sensor to sense movement, alternatively I could have also inserted a push button or used a Hc-SR04 ultrasonic sensor. The ultrasonic sensor however is a very flimsy and delicate sensor, speaking from previous and current experience it is very hard to work with it – this time the sensor was not recognising the echo pin even for task 7.3D – when I tried to run it again with the same code that had worked previously, hence I changed it last minute and decided to go with PIR motion sensor.

For the servo motor, the MG995 is a powerful servo motor that would hold enough capacity to be able to open a door and hence why I used a more powerful motor.
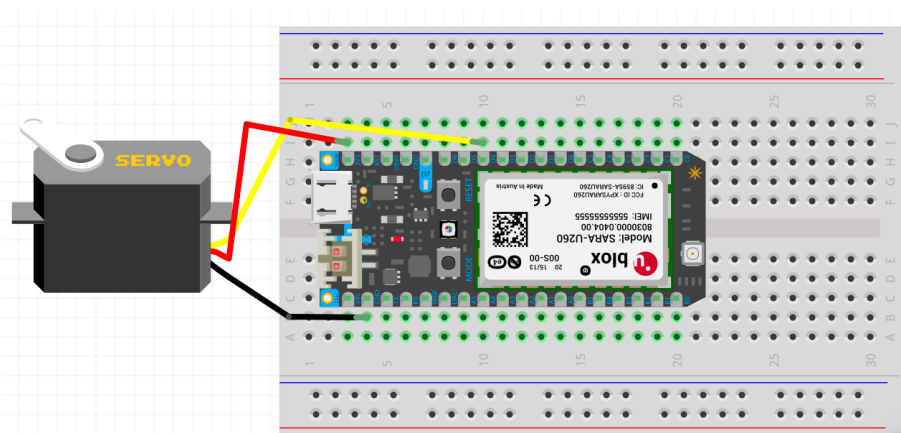
Prototype Architecture

Raspberry Pi:
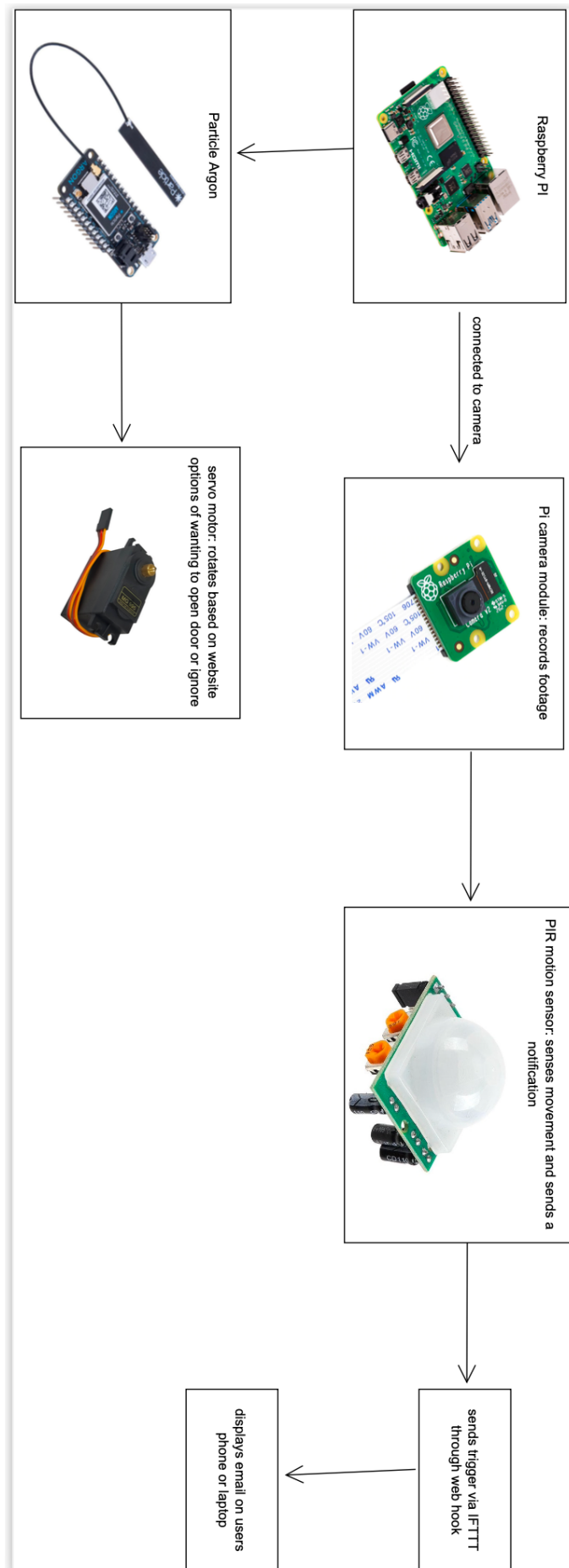PIR motion sensor:
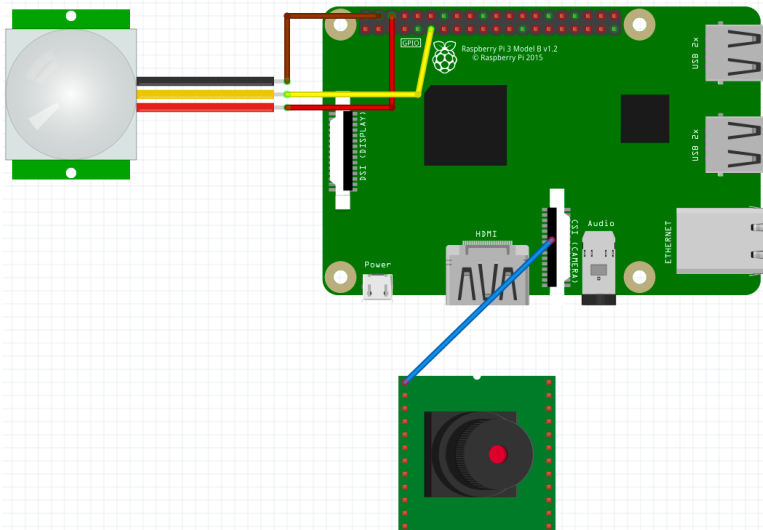
Connections:

PIR motion sensor:
-gnd pin of motion sensor to to gnd RPi (pin 6)
-out pin of motion sensor to RPi gpio17 (pin 11)
-vcc pin of motion sensor to RPi 5V (pin 2)

Connections:

Servo motor:
-gnd pin of servo motor to gnd pin of particle
-vcc pin of servo motor to to 3v3 of particle
-pulse pin of servo motor to D4 (trigger pin) of particle

Particle Argon

Raspberry PI

connected to camera

servo motor: rotates based on website options of wanting to open door or ignore

PI camera module: records footage

PIR motion sensor: senses movement and sends a notification

sends trigger via IFTTT through web hook

displays email on users phone or laptop
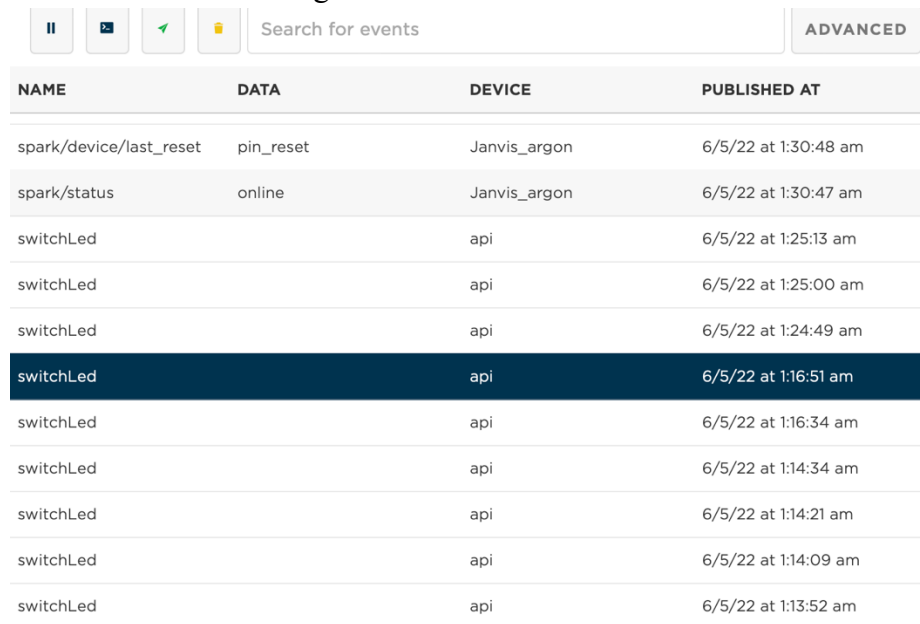
Link to prototype code on Github

https://github.com/janviig/doorbell

Testing approach you have used for evaluating your system

I aimed to have a more integrated approach to my project. My original aim was to have the motion sensor sense movement and send a webhook through RPi to Particle and call a function, which would be to blink an led to let the person know who was sensed by the sensor that the sensor has recognised them. The led light blinking would then send an email through IFTTT to let the owner know someone is at the door, and they can view the camera footage through a website and be able to open a gui mockup on the RPi to click opening the door or ignoring which would be to rotate the servo motor so it is able to open the door. However, I wasn't able to get the RPi and Particle argon IFTTT to work – the RPi was sending a trigger, but the led was unable to blink. Below is a screenshot of the RPi calling a trigger but the

Particle was not working:

NAME	DATA	DEVICE	PUBLISHED AT
spark/device/last_reset	pin_reset	Janvis_argon	6/5/22 at 1:30:48 am
spark/status	online	Janvis_argon	6/5/22 at 1:30:47 am
switchLed		api	6/5/22 at 1:25:13 am
switchLed		api	6/5/22 at 1:25:00 am
switchLed		api	6/5/22 at 1:24:49 am
switchLed		api	6/5/22 at 1:16:51 am
switchLed		api	6/5/22 at 1:16:34 am
switchLed		api	6/5/22 at 1:14:34 am
switchLed		api	6/5/22 at 1:14:21 am
switchLed		api	6/5/22 at 1:14:09 am
switchLed		api	6/5/22 at 1:13:52 am

So, to evaluate my system I changed it up a little bit, and used the sensor to detect motion and then created a website for the 2 buttons for opening or closing the door via a website for the servo motor rotation mechanism – this was done through the particle argon. Originally, I did plan to have the GUI mockup on the RPi to be able to open the door or ignore, I had the code working for this as well, however because the LED wasn't working I just switched the two around.

User manual

Given the customer has the prototype already:

Cloning GitHub repository:

1) *in terminal type git clone https://github.com/janviig/doorbell.git*

To install the camera module on the RPi:

1) *open terminal*
2) *type sudo raspi-congif*
3) *click interface options → camera → yes. This will enable camera module*
4) *to check the camera is working, take a picture on the camera by typing this in terminal; raspistil -o Desktop/image.jpg*
5) *your camera module is now ready to go*

Downloading libraries for the code to run:
The following steps are all commands to be executed in terminal:

For camera viewing footage install:
- *sudo pip install "picamera[array]"*

Installing open cv – installing this package can be a bit more time consuming so please follow these steps to avoid open cv not installing fully:
1) *check if you are using all your memory:*
   -in terminal: df -h, if the numbers don't add up perfectly run:
   sudo raspi-config →advanced →expand filesystem → reboot your pi
2) *update and upgrade your RPi*
   sudo apt-get update
   sudo apt-get upgrade
3) *check your python version*
   python -v. it should be some sort of python 3 version
4) *sudo apt-get install python3-pip python3-virtualenv → y*
5) *create a project directory either on terminal or in documents*
   mkdir doorbell
   cd doorbell
6) *install the following packages in the folder*
   python3 -m pip install virtualenv
   python3 -m virtualenv env
   source env/bin/activate
7) *download the following system packages all in one*
   sudo apt install -y build-essential cmake pkg-config libjpeg-dev libtiff5-dev libpng-dev libavcodec-dev libavformat-dev libswscale-dev libv4l-dev libxvidcore-dev libx264-dev libfontconfig1-dev libcairo2-dev libgdk-pixbuf2.0-dev libpango1.0-dev libgtk2.0-dev libgtk-3-dev libatlas-base-dev gfortran libhdf5-dev libhdf5-serial-dev libhdf5-103 libqt5gui5 libqt5webkit5 libqt5test5 python3-pyqt5 python3-dev
8) *to install opencv*
   pip install opencv-python
9) *testing to see it all downloaded*
   python
   import cv2
   cv2.__version__

open cv has now been installed successfully and you should be able to run the cameramod.py file with no issues.

To initialise webhook on RPi and IFTTT:
1) *click create applet on the ifttt website*
2) *for the if part select webhook → receive a web request*
3) *type motion_detected in the event name to correlate with the code file*
4) *for the then part select email and change the part after* <u>extra data</u> *to someone is at your door*
5) *now update that and your whole applet this will now create an applet for you*
6) *to ensure your webhook is being picked up click on the ifttt icon on the website and search up webhook, now click settings on the top right and you should see a key for your webhook, copy that key*
7) *under the heading – <u>to trigger your event</u>, you will see the website link – in the **{event}** box enter motion_detected*
8) *in line 26 of the motionsensor.py file, change the underlined part below to your webhooks unique key:*
   https://maker.ifttt.com/trigger/motion_detected/with/key/<u>xxxxxxxxxxxxxx</u>
9) *save the code file and run the code, the applet should be working and you should receive a notification*

Steps for Particle argon:
1) *compile and flash the code onto the particle argon*
2) *open the servo.html file for the buttons*
3) *this will allow you to select the options of wanting to open the door or ignore*

Challenges and lessons learnt
The challenged I faced was downloading open CV, I wasn't able to download open CV correctly. I couldn't run the camera code from the very start and this took a long time to figure out how to actually download this package, eventually I did manage to download it successfully be able to view the camera footage and hence why the steps for downloading this package are super detailed above.

The ultrasonic sensor not working all of a sudden was a big step back as I was not able to program it to be able to distance in terms of when it's going to recognise movement, I had planned for that to be in the 5-10cm's range. However since it had suddenly stopped working - it wasn't even working with previous code that it was working with for me so I did have to change my plan and have to use a PIR motion sensor. Luckily I did have one on hand to be able to use one but the ultrasonic sensor is very temperamental and it's very hard to be able to implement it and trust it fully.

The Wi-Fi on the particle argon is very moody and the Wi-Fi kept going off so I was not able to run my code even practice codes I had done or done from previous tasks that had worked completely fine we're not able to be executed because of the Wi-Fi. I did try resetting it a

couple of times and I tried changing the modes I tried everything I could, but it was very hard to still get it working. My aim was to get an LED blinking on it via a web book from the raspberry pie however I was not able to do that at all because again the Wi-Fi was highly unpredictable. Even though I was able to get the LED the blink at one point with the test occurred I tried to test again and the LED wouldn't blink with the same circuit. Again, the Wi-Fi was a huge problem in this whole project and that is why unfortunately I did have to step back a little bit and change my plans instead of the LED blinking I did have to do with the seven motor on the Particle.

Future extensions for project
In the future I will try to do the LED blinking system with an Arduino instead, because an Arduino is easier to use, however it does not have the Wi-Fi or the Bluetooth modules that a particle does so to be able to import those would be an extra step but I do think it will be worth it in the end it is more it is a more reliable than the particle.

In the future if I had a pushbutton on hand I would use that instead of a PIR motion sensor as that motion sensor is very sensitive and that's the reason I did want to use a ultrasonic sensor because I could have programmed it to the centimetres I wanted it to be able to detect movement so it was going to be a little bit more accurate, where as this one is more sensitive so it's harder to use - I would use a pushbutton instead so that it's just easier user knows that someone does everything you want to speak to them and it's not just someone passing bye.

I would also aim to create a website to be able to view the footage and somehow be able to send that website link in the email with the IFTTT trigger notification. However, I was not able to send a link in the email at all and through if I did try to but I had no luck.

Conclusion
Overall, I think the project was still carried out well however it wasn't done to the full standard and capacity that I had hoped and I had planned to due to unforeseen circumstances. This project aims to assist families, and medical centres that are looking for increased security within their life and hope to be able to have a secure environment for their loved ones and the protection of their family.