CEN 5035 Principles of Software Engineering, Fall 2019

Group 5 - The Smart Security System

Adam Corbin (acorbin3@fau.edu) Malia Kency Stincii Dumerjean Jared Peterson

December 2019

Contents

1   Revision History

Description	Date	Revision number
Initial proposal submission	9/27/2019	v1
Final report submission	12/4/2019	v2

2 Executive Summary

Our group project during the Fall 2019 semester in CEN 5035 Software Engineering will be a Smart Security System. We have found there is a lack of flexibility to set up schedules for a security system which lead us down this path. We are looking to solve this problem by having an easily configurable way to schedule when the alarm should be enabled or disabled. This system will also have the standard features of a typical alarm system.

The Smart Security System breaks away from the competition in its ability to connect to and utilize with the owner's digital calendar to automatically create a detection schedule.This system will also have voice controlled activation which can be used to arm and disarm the system through an Alexa or Google home device. To reduce the risk of failure, the system will also have a backup keypad to enable or disable the detection system in the event that voice was not sufficient.

The security system will include cutting edge passive infrared motion sensing technology to be able to detect any movement in the desired location. The alarm warning system will announce that a break in has been detected and the authorities have been notified to ward off intruders. For added security, the system will capture images, video and audio to relay back to the system owner and or authorities should a break-in occur. The system will also have the ability to confirm if an event was real or not using SMS text and/or phone call notifications. Giving the owner a short time frame to disregard the event before the authorities are notified.

Finally the system will have a password secured web interface to be able to view and update the monitoring schedule. The web system will provide the user the ability to activate or disarm the alarm and view a live stream of their camera at any given time.

Our target audience will be for anyone who wants to have a security system that has a busy schedule where they might be traveling often. This will help ensure that the alarm system will always be active when the home owner is away.

3   Competitive Analysis

	ADT Pulse	Vivint	Ring Alarm	FrontPoint	SimpliSafe	Smart Security System
24/7 Monitoring	X	X	X	X	X	X
Live Alerts	X	X	X	X	X	X
Motion Detection	X	X	X	X	X	X
Voice Control	O	X	X	X	X	X
Arm/Disarm Remotely	X	X	X	X	X	X
Schedule sync	O	O	O	X	O	X

Symboal	Meaning
X	Includes
O	Excludes

TODO - Consider beefing this up a little. In the first example given in class they have 2 sections: Competitive relationship and Competitive advantages

An advantage that our Smart Security System is that the system will be able to connect up with the owner's digital schedule. There is only one other system that has that capability, FrontPoint. The advantage our system has over FrontPoint, along with the other systems, is the ability to receive these alerts quick enough in order to determine whether the alert was real or not. Our system will be able to notify authorities quickly enough after confirmation.

4   Data Definition

TODO - Consider going back and seeing we need to add anymore definitions. Also probably need to go into a little more detail

- Enable/arm - Turning the system on to monitor for intruders
- Active - Currently monitoring for intruders
- Disable - The system inactive and not monitoring for motion
- Disarm - The act acknowledging the system of a false alarm
- Notification - sending an alert to the system owner
- System owner - User in control of the system who can enable/disable the system
- Detect - System has found a disturbance
- Monitor - the act of checking the sensors to see if the signals are out of normal range
- Record - capturing video and or pictures
- Sensing - This the period where the system has been Armed and the waiting period has lapsed. During this time if there has been activity spotted by the PIR then the alarm will be activated

5   Overview of Scenarios and Use Cases

Use Case: Enable Alarm

Brief Description : System owner arms the monitoring service

Basic Flow :

1. System owner chooses to enable monitoring service
2. System asks for code to arm monitoring service
   - System verifies that system owner would like to arm monitoring service
   - Once verified, system arms monitoring service

3. System Owner is notified that monitoring service has been armed on user interface

Alternate Flow :

1. System Owner chooses to enable monitoring service
2. System asks for code to arm monitoring service
    - Code provided is incorrect, system allows for another try
    - Code provided is incorrect, system is locked
3. System Owner must verify identity to unlock system

Use Case: Disable Alarm

Brief Description : System owner disarms monitoring service

Basic Flow :

1. System owner chooses to disable monitoring service
2. System asks for code to disarm monitoring service
    - System verifies that system owner would like to disarm monitoring service
    - Once verified, system disarms monitoring service
3. System Owner is notified that monitoring service has been disarmed on user interface

Alternate Flow :

1. System Owner chooses to disable monitoring service
2. System asks for code to disarm monitoring service
    - Code provided is incorrect, system allows for another try
    - Code provided is incorrect, system is locked for 10 minutes

---

Use Case: Disable False Alarm

Brief Description : System owner disarms monitoring service after an incident

Basic Flow :

1. System owner is notified of possible disturbance
2. A video/picture is sent to system owner
    - If disturbance occurs outside of normal range
        i. System owner can choose to dispatch authorities
        ii. Or system owner can choose to disable alarm
3. See **Use Case Disable Alarm**

Use Case: Notify System Owner

Brief Description : System notifies system owner if monitoring service was triggered

Basic Flow:

1. Monitoring service is triggered
2. Monitoring service takes video/picture of surrounding area
3. Monitoring service sends an SMS to system owner that the service was triggered
4. If disturbance occurred outside of range of camera but within range of motion sensor
    - System owner is given option to dispatch authorities or disable alarm
    - If user chooses to dispatch authorities
        i. User then has option to contact authorities
    - If user chooses to disarm the alarm
        i. See **Use Case Disable False Alarm**

6   High-Level functional requirements

6.1   Detection System

- **HLR.REQ-1** - The system shall monitor user designated areas for any signs of movement when the system has been armed. ## Automated Monitoring Calendar
- **HLR.REQ-2** - The system shall provide the user with a calendar that can be utilized to schedule times when the system should arm and disarm automatically. ## Recording System
- **HLR.REQ-3** - The system shall stream video, and audio from an area designated by the user.
- **HLR.REQ-4** - The system shall capture pictures and record audio/video when the system has been armed and triggered. ## User Interface
- **HLR.REQ-5** - The system shall provide the user an interface for viewing pictures and video recordings.
- **HLR.REQ-6** - The interface shall provide the user the ability to view and edit the monitoring calendar.
- **HLR.REQ-7** - The system shall provide the user with the ability to arm and disarm the system. ## Alert System
- **HLR.REQ-8** - The system shall audibly notify the user and the surrounding area when the system has been armed and triggered. ## Reporting System
- **HLR.REQ-9** - The system shall send messages to the user when the system has been armed and triggered.

7   List of non-functional requirements

- **HLR.REQ-10** - The system shall report and handle when any and all components are disconnected or unavailable.
- **HLR.REQ-11** - The system shall handle erroneous inputs from the user.
- **HLR.REQ-12** - The system shall be secure and enforce secure password requirements on the user.
- **HLR.REQ-13** - The system shall be easy to learn, taking no more than 15 minutes for new users to start utilizing all of the system's features.
- **HLR.REQ-14** - The user interface shall be capable of being accessed from any device with internet access.
- **HLR.REQ-15** - The system shall be able to recover from a temporary total power loss.
- **HLR.REQ-16** - The system shall restrict read/write/modify access to video/audio streams, pictures, video/audio recording to the owning user.
- **HLR.REQ-17** - The system shall respond to user reporting feedback within 60 seconds.
- **HLR.REQ-18** - The system shall store 1GB of video recordings and 250MB of pictures.
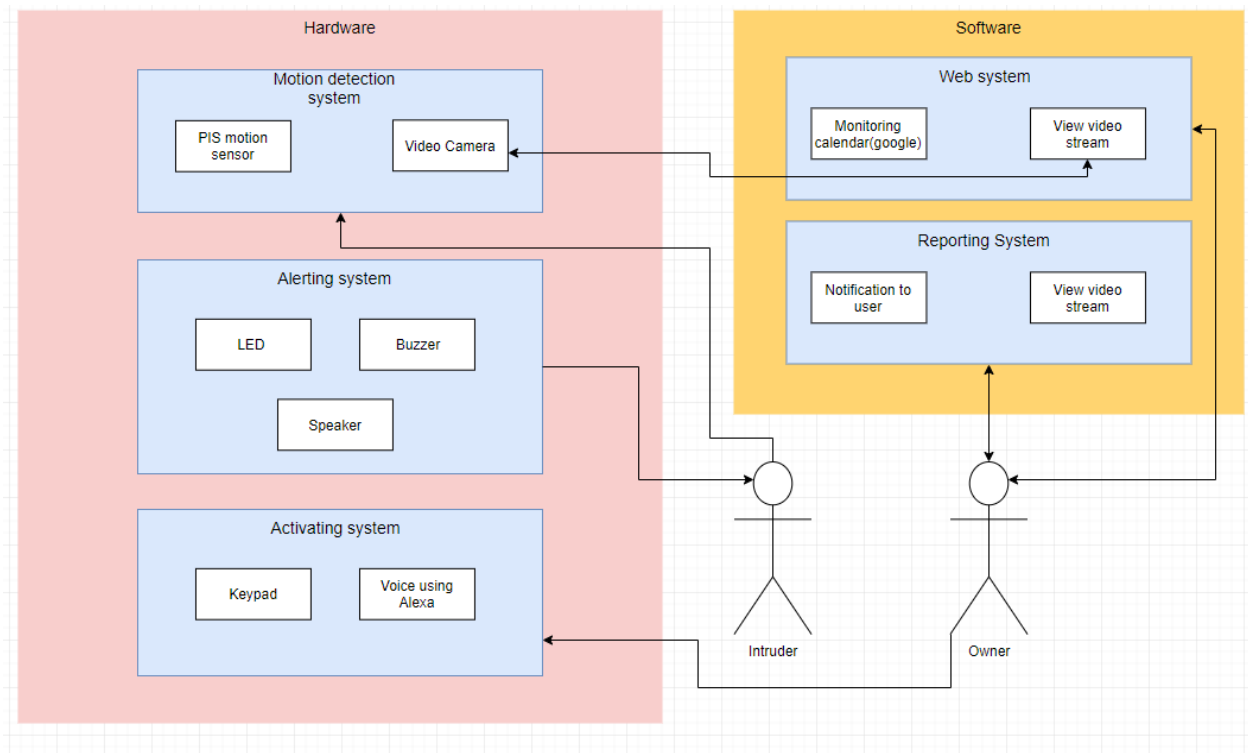
8   High-Level System Architecture

Figure 1: System Architecture

Link:https://www.draw.io/#G1RfKGCpj4uxjit77ksxuVJuZhFB1gMUyB TODO - update the graph to remove the buzzer and speaker and alexa. Add the Ultrasonic sensor in the Motion Detection system TODO - Explain text based on the interactions between the sub-systems
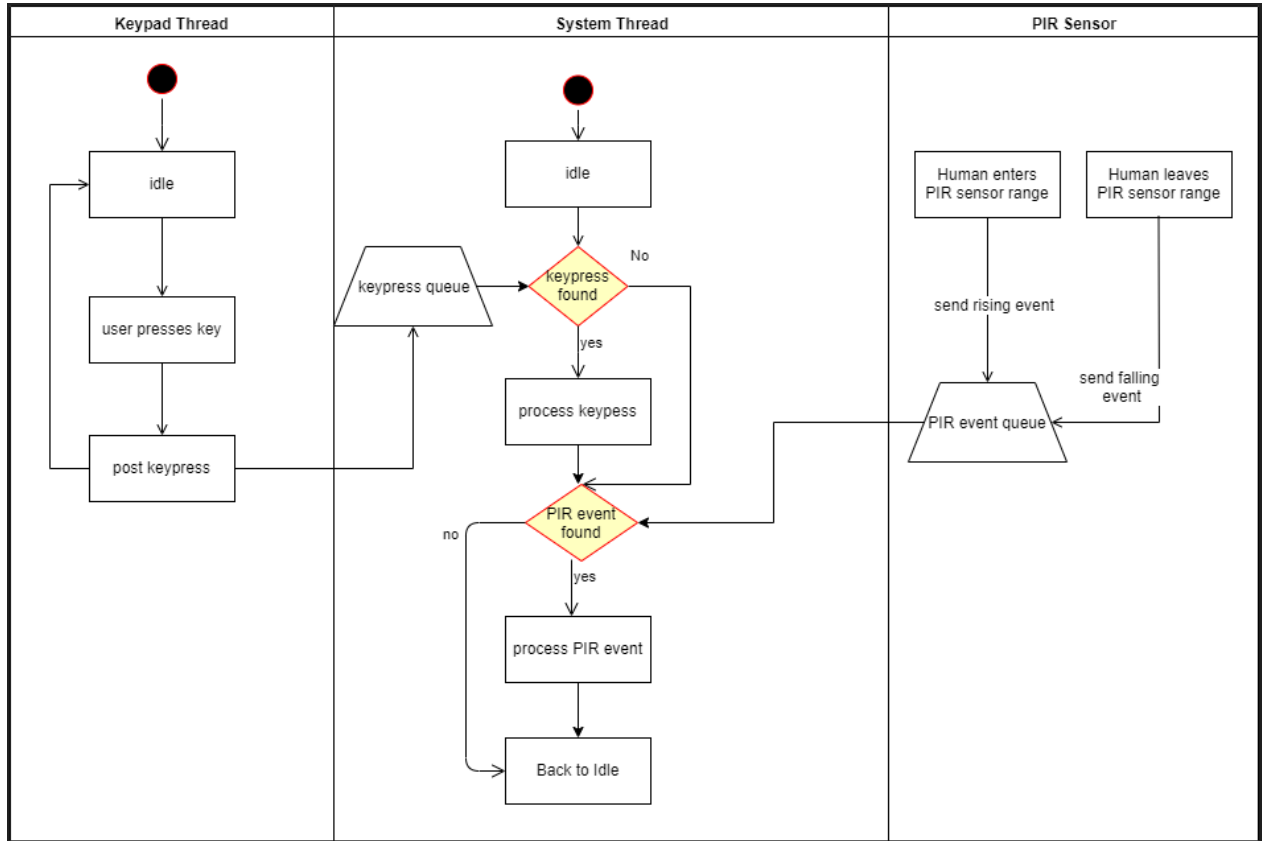
Figure 2: Activity Flow diagram between Keypad, system, and sensor thread

9   Implementation details

9.1   Inputs

9.1.1   PIR Sensor

The PIR Sensor will be use to detect movement within a room. For this project we will be using the HC-SR501 PIR Motion Sensor. PIR stands for Passive Infrared which means it doesnt use any energy to detect signals The PIR sensor consist of a main compost of the pyroelectric sensor which when exposed to heat generates an electric signal. A warm body emits infrared radiation(IR) and this pyroelectric sensor picks up this IR signals. On top of the pyroelectric sensor there is a fresnel lens bounces the IR signals onto the pyroelectric sensor.
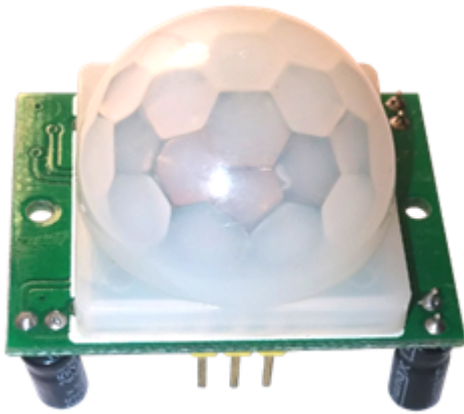
Figure 3: HC-SR501 PIR sensor

9.1.1.1 PIR Sensor pin description   This PIR Sensor has 3 different pins: power, ground, and the output. The output pin will give a 5v or a 0v signal representing if the sensor has detected anything. 5v being that the sensor has detected IR signals and 0v for lack of IR signals. The power is expected to use 5v-20v input with 65mA current consumption.

9.1.1.2 Adjusting the PIR Sensor   The PIR Sensor module has 2 potentiometers. 1. Adjusting the sensitivity of the sensor to reach up tp 7 meters 2. Adjusting the time the output signal stays high when IR signals have been detected The time can be adjusted between 0.3 seconds to 5 minutes.

9.1.1.3 Trigger modes   The PIR Sensor module has a set of 3 pins to select the different types of trigger modes. 1. Non-repeatable trigger mode : This is when the PIR Sensor has been triggered and the output time has lapsed, the output signal will automatically go back to zero 2. Repeatable trigger mode : Similar to mode 1 but once the output time has lapsed and the IR signals still are detected the output would remain high. In a security system the Repeatable trigger mode is what we are looking for. For different applications like backyard light for night where you might only want to have it on for a short period of time.

9.1.1.4 PIR Sensor stabilization   The PIR Sensor when powering up need to have about a minute to stabilize to adjust to the area before any valid outputs will occur. The average IR gets computed during this time and when a warm body comes into range, thats where the spike of IR signal will occur causing the PIR Sensor to signal activity has occurred.

9.1.1.5 PIR Sensor add ons   The HC-SR501 PIR Motion Sensor has the ability to connect up to a few other modules right on this board for different applications. You can add alight sensitive resistor or a photo resistor. This can be used with the PIR Sensor so that it will make it only active in the dark. There is another place to add a thermal sensor which can factor readings into the PIR Motion Sensor.

9.1.1.6 Initial testing   Before connecting up the Raspberry Pi this PIR Sensor can be tested without reading the output values pin. To do this experiment there needs to be an LED connected from the output pin to he ground pin. After the PIR Sensor has stabilized the LED will then turn on when IR signals have triggered the PIR Sensor. This was a good way to test out the PIR Sensor before adding any of the complexity of the Raspberry Pi.

Source for some of this data: https://www.youtube.com/watch?v=ZC_sEW3_694

9.1.2   4x4 Membrane Matrix Switch

In typical security systems the Smart Security System needed a physical device for a user to arm and disarm the security system. The sensor kit that we used had a 4x4 membrane matrix switch that would work perfectly in our prototype.

This keypad has 16 individual keys but our Raspberry Pi only has a limited number of inputs. That means we need to multiplex the keys in order to achieve less inputs into the Raspberry Pi. Here are the following steps in order to achieve our matrix 4x4 for all 16 keys 1. Set the columns as outputs to High 2. Set the rows pints as inputs with pull-up resistors 3. While checking for an individual key has been pressed, 1 at a time the output will be set to Low. 4. While the 1 output is set to low, we will check if any of the inputs as set to Low, this will indicate that a button as been pressed 5. In order to avoid sending multiple keypress events while the button is held down, the system will wait till that Low becomes High again

System design on how to use the keypad class: https://drive.google.com/file/d/1uPLhlLkEPWDTnYCvWuJ4w6HXwep1SSEe/v

The Keypad creates its own thread to capture all the key presses and then sends each key press into the queue. The intention would be there is a separate thread that would process the keys using the queue

Figure 4: 4x4 Matrix membrane switch

9.1.3   Playstation Eye Camera

-TODO fill in details about camera

Figure 5: PS3 Eye

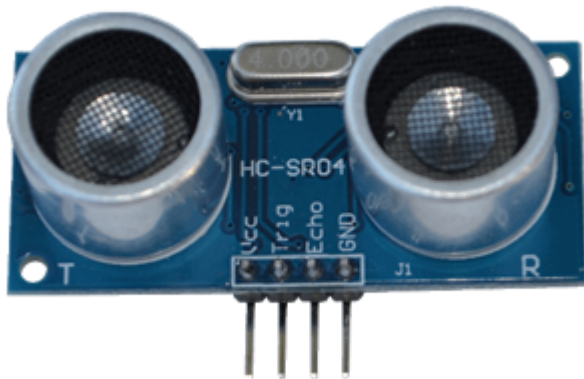### 9.1.4   Ultrasonic Distance Sensor

-TODO fill in details

Figure 6: HC-SR04 Ultrasonic Distance Sensor

9.2 Outputs

9.2.1 LED Module

9.3 Services

9.3.1 Watson Graphs

The IBM Watson service has a ton of different things from processing speech to text, text to speech, searching things online, and many more. While investigating how Watson could be integrated with the Smart Security system it was found that Watson could create time series graphs of events. This complemented the Smart Security system well since we have many different events such as movement detection, arming the system, and when the alarm is sounded.

Integrating the Watson graphs were relatively easy to set up. The events just needed to be created from the Watson web interface and then the system just needed to connect up to Watson and send the defined events.

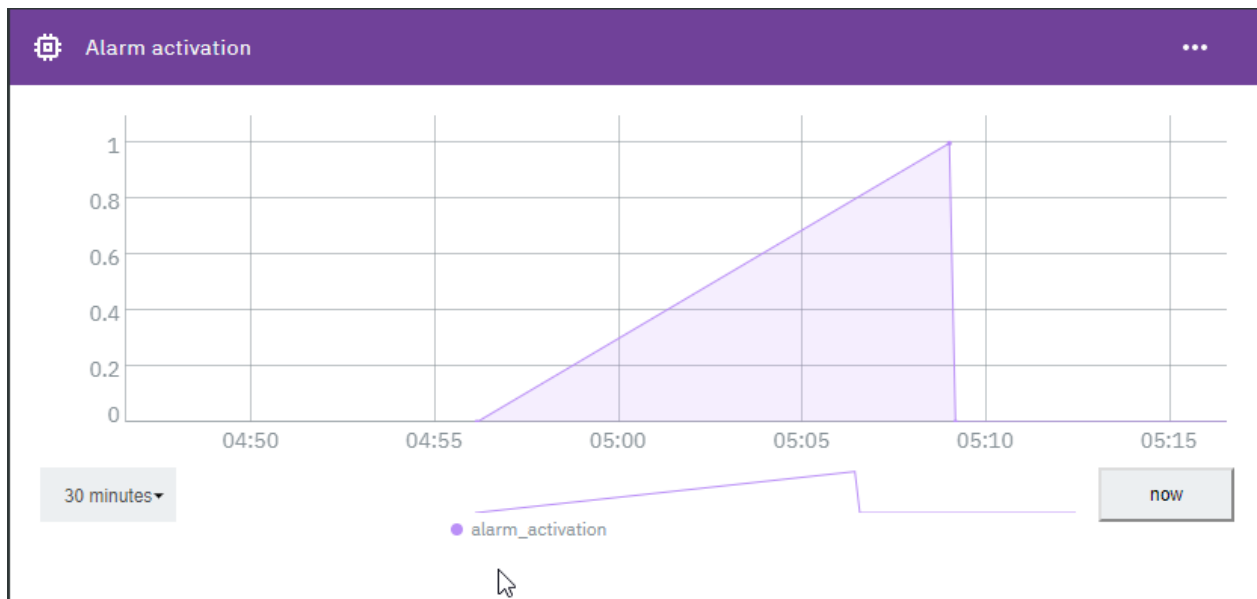Here are some pictures of the graphs that Watson creates
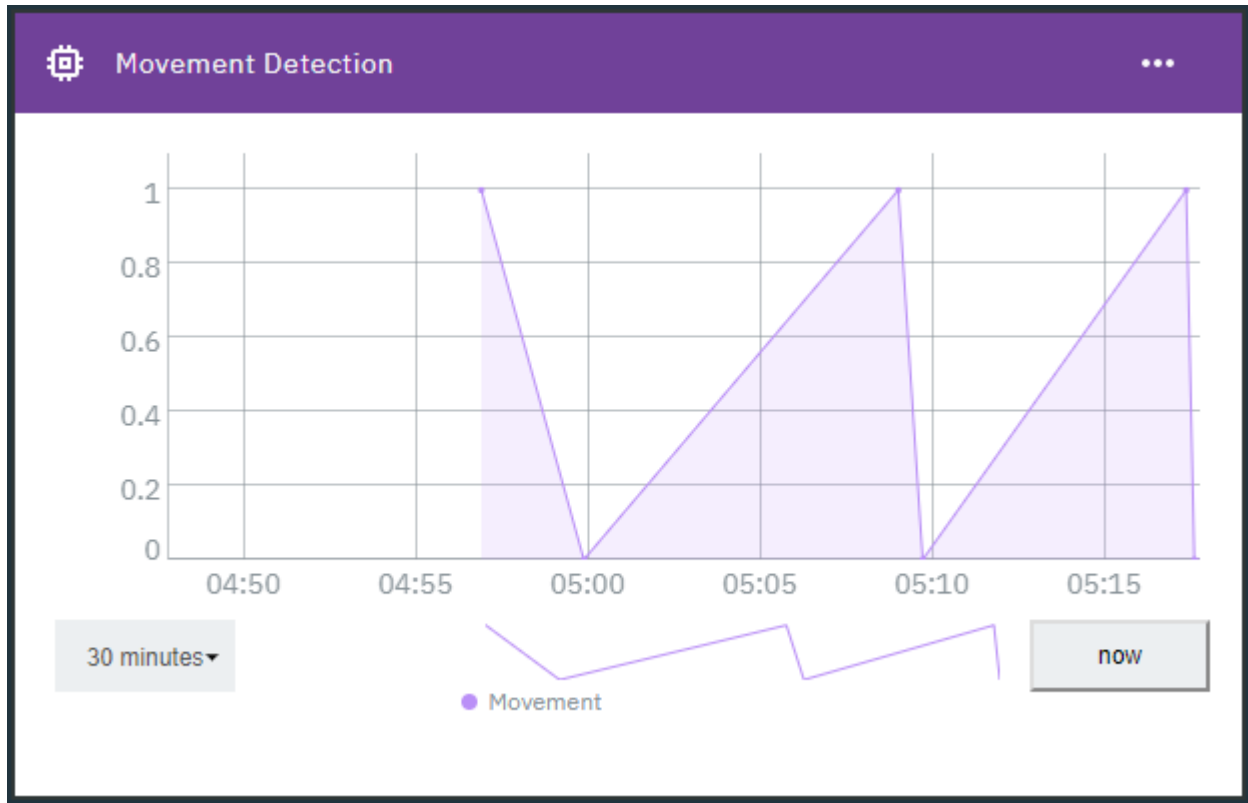

Figure 7: Alarm activation
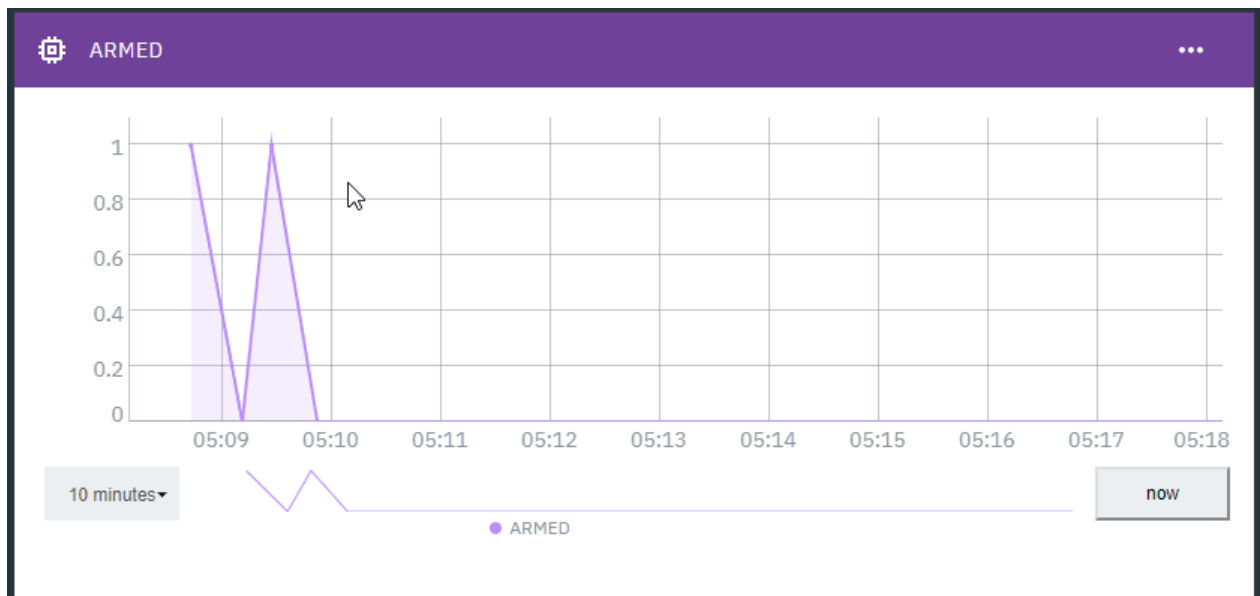
Figure 8: Movement Detection

Figure 9: System Armed

9.3.1.1 Challenges   Ideally it would be nice to embed these graphs in our user interface but it doesnt look like Watson has the capability to share these graphs publicly. The only way you can view them is to be on an authorized email list and you have to login to IBM's Watson system.

9.3.1.2 Considerations   For the Smart Security system we considered using voice activation to arm and disarm the system but in the end it seemed like a security risk in the event that it didnt function properly or if an unauthorized user found out about the verbal pass code. Possibly in the future when Watson can self identify someone based on their void then this could be a great feature to add to this Smart Security System

9.3.2 Google Calendar as smart enable system

The concept of the Smart Security System would to leverage the use of hte Google Calendar to know when the system should be monitoring for intruders. The system will periodically monitory for calendar updates. It will also activate and deactivate the alarm system based on which events are running.

9.3.2.1 Determining when to monitor   There can be a few different ways to determine when the alarm system should be activated.

1. Alarm only on during meetings - With little interactions this will make it simple for the system to only be monitoring when there is a meeting happening. This could be problematic too because its possible that someone might be working from home and they might have a false positive . A way to combat this will be introduced in the next option

2. Labeled meetings - Adding some keyword to signify if during this meeting the system should be be monitor mode. This does leave some room for error if someone forgets to tag the meeting that the system should be monitoring.

3. Complete separate alarm events - Another option would be to completely separate the events in their own type of meeting to signify only for monitoring.

4. First and Last meetings - Automatically create a range where the system will be active the start of the first meeting till the end of the last meeting. This still has holes since some meetings might take place at the home of the system owner.

Due to the time constraints we will be using 1, for our purposes to show that we can interact with the Google Calender

9.3.2.2 Considerations   Along with the alarm events is possible to connect up with the user's phone GPS to allow for the system to automatically activate based on the GPS coordinates. Having some kind of range when someone leaves 0.5miles from home then the system could be activated automatically.

9.3.3 Motion library

Since the semester is a short time frame our team tried to leverage as much open source software as much as possible. Our initial setup of getting the PS3 Eye Camera working used a library called motion. This library is highly configurable and can support many different types of cameras. It was not designed specifically for the Raspberry Pi but has supporting plugins to help communicate to the cameras. Its intended platform is linux based and was also intended to be used as a way to monitor multiple cameras. Project website: https://motion-project.github.io/motion_config.html

TODO_AC - add how it was installed

TODO_AC - Add what configurations needed to be updated in order to get working To start the video stream run "sudo motion -b"

9.3.4 Flask framework

TODO - Add details as to why Flask was chosen TODO - add details how it works & how it interacts with the main system TODO - add any considerations we might have
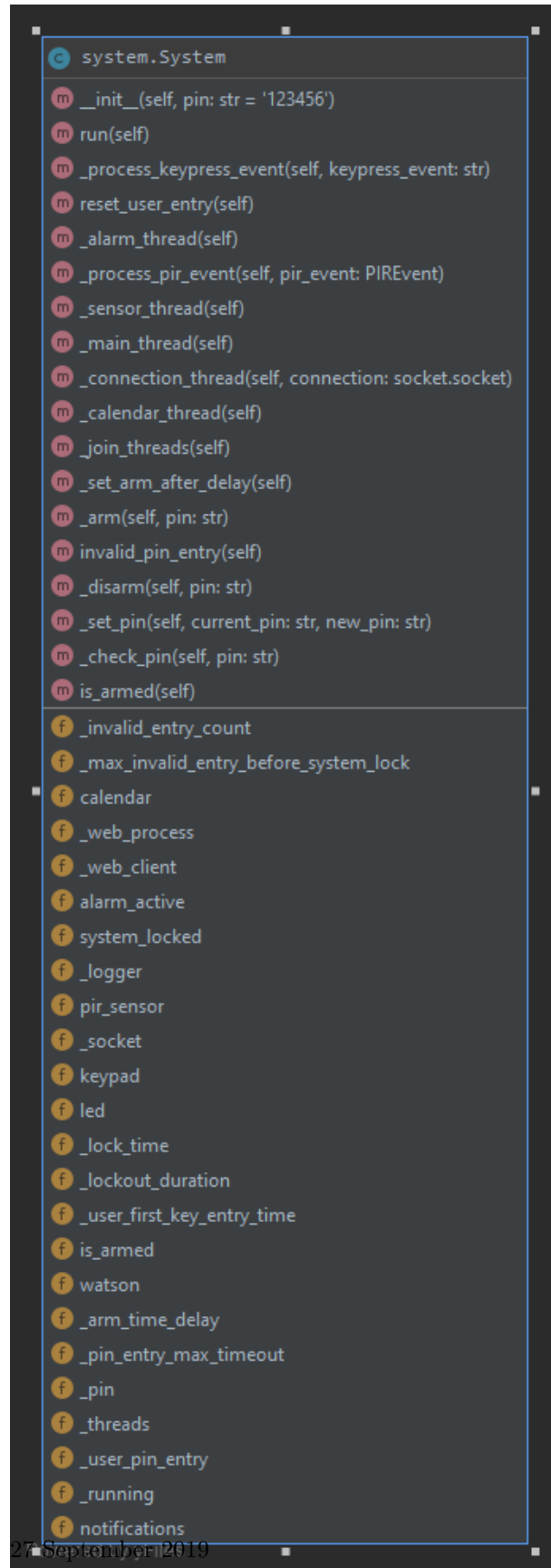
9.3.5 Twilio notifications

TODO - Add details as to why Twilio was choses TODO - add details on how it works and how it was integrated into the main system TODO - add any considerations

9.4   Class Diagrams

9.4.1   System class

```
© system.System

ⓜ __init__(self, pin: str = '123456')
ⓜ run(self)
ⓜ _process_keypress_event(self, keypress_event: str)
ⓜ reset_user_entry(self)
ⓜ _alarm_thread(self)
ⓜ _process_pir_event(self, pir_event: PIREvent)
ⓜ _sensor_thread(self)
ⓜ _main_thread(self)
ⓜ _connection_thread(self, connection: socket.socket)
ⓜ _calendar_thread(self)
ⓜ _join_threads(self)
ⓜ _set_arm_after_delay(self)
ⓜ _arm(self, pin: str)
ⓜ invalid_pin_entry(self)
ⓜ _disarm(self, pin: str)
ⓜ _set_pin(self, current_pin: str, new_pin: str)
ⓜ _check_pin(self, pin: str)
ⓜ is_armed(self)

ⓕ _invalid_entry_count
ⓕ _max_invalid_entry_before_system_lock
ⓕ calendar
ⓕ _web_process
ⓕ _web_client
ⓕ alarm_active
ⓕ system_locked
ⓕ _logger
ⓕ pir_sensor
ⓕ _socket
ⓕ keypad
ⓕ led
ⓕ _lock_time
ⓕ _lockout_duration
ⓕ _user_first_key_entry_time
ⓕ is_armed
ⓕ watson
ⓕ _arm_time_delay
ⓕ _pin_entry_max_timeout
ⓕ _pin
ⓕ _threads
ⓕ _user_pin_entry
ⓕ _running
ⓕ notifications
```

9.5   Development Environment setup

Since our team was made up of people in many different locations we needed a good way to communicate, develop, and test our system. For our source control and task management, we used GitHub. It was a nice way to organize our work and prioritize what needed to get done. In order to use the actual Raspberry Pi remotely there needed to be some channels opened up to allow for file transfer and remote access. FTP/SSH was opened up for that purpose and VNC was enabled for remote access to the Raspberry Pi. ### Tools * Filezilla - File transfer system to move files from a local computer to the remote Raspberry Pi * Putty or any SSH program - This program is used to tunnel into the terminal of the Raspberry Pi which would be used to run scripts * Python file editor like Notepad++ or PyCharm - This is intended to be used to create the python files on your local host. Because some of the scripts need hardware outputs, most might only be able to be tested using the remote Raspberry Pi * Real VNC client - This will be used if someone wants to visually see the Raspberry Pi desktop. In most cases this might not be needed but its good to have this option available ### Port Forwarding on router for SSH In order to get the SSH and FTP working the router need to open up the ports to the outside world. Otherwise the router would just block all requests and it would appear if something isn't working.

The following ports were opened up: * Port 22 for SFTP transfers * Port 2022 for SSH communications * Port 5901 for VNC * Port 8081 for camera streaming using motion ### Workflow to editing and testing out code 1. Locally clone repo from https://github.com/AdamCorbinFAUPhD/CEN5035-Software-Engineering-Course-Project 1. Make changes to the files if necessary 1. Send files using FileZilla * Setup of FileZilla * Screenshot of the settings of the server
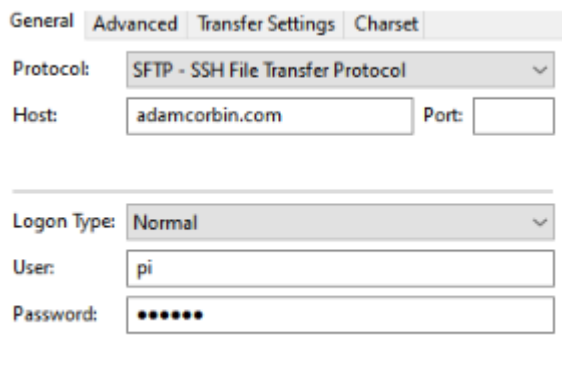
1. From putty you can run the python files by doing: "python3 file.py"

2. If you need to make any changes do it from your local computer and then use FileZila to transfer the file over to retest

9.5.1   Real VNC

In the event that someone would like to remote into the Raspberry Pi desktop RealVNC was installed on the Pi. In order for this to work on someones local computer, their email would need to be added to the approved email list. Once added to the list, all thats needed is to download the local client to communicate to the server on the Raspberry Pi.

10   Verification

TODO go into some test cases

TODO go into the manual testing

TODO go into the auto testing

11   Team Member Contribution

1. Adam Corbin: Product Owner
2. Malia Kency: Individual Contributor
3. Stincy Dumerjean: Individual Contributor
4. Jared Peterson: Scrum Master