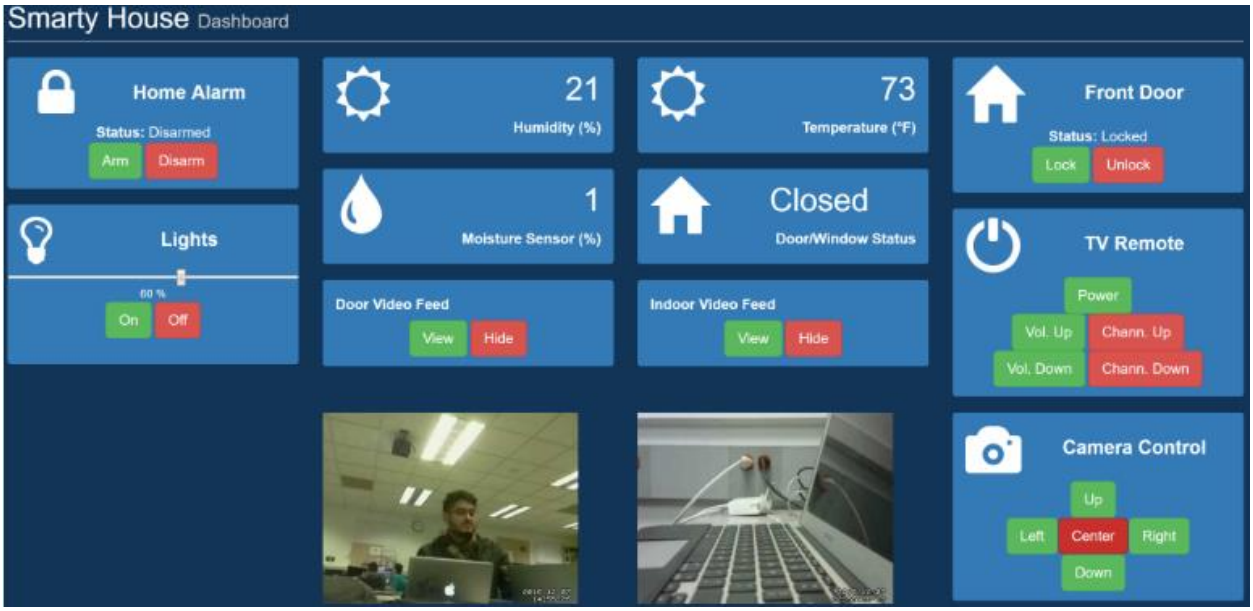
Design Documentation
Fall 2016
Team 6: Smarty House

Group Members: MarcoAntonio Ledesma
Jesus Espinoza
Omar Guzman
Jorge Gonzales

Contents

Abstract

Smarty House is a project developed with home automation and security features for the benefit and comfort of the users. The purpose of this project is to facilitate the interaction with home security and daily tasks. Our system provides our users a reliable home security system with facial recognition and passcode. Moreover, our Smarty House system also provides voice commands using Amazon's Alexa Voice Services to control the lights, television and the door lock. On top of this, Smarty House also provides a web application to display home details such as temperature, humidity, soil moisture, door/window and alarm status. The web application also allows the users to control the lights, television, camera, and door lock. Currently there is a wide variety of home automation systems out on the market. Smarty house provides the ability to interact with all the home security and automation features directly from our web application and Amazon's Alexa.



Introduction

The motivation behind our project is that Home Automation Systems are as of the time of writing this report a new technology available to developers and thus something of interest to everyone not only with programming knowledge, but to anyone who owns even a simple smartphone as many home automation features can be implemented with just that. Currently, the options on the market for people interested in home automation systems are to buy a central system to send their commands such as the Amazon Echo, through the use of a paid-for phone application, or other systems. However, the issue that arises with these systems is that alone, no home automation features can be implemented to use, and instead the consumer needs to purchase separate components depending on what feature that s/he wants to implement.

These options as one can see can get expensive very fast and with new technologies always entering the market, in time they will become obsolete which will cost the consumer more money to keep his automation system functional. Therefore, with the inspiration to improve people's lives and save them some money in the process, our project began as an idea to develop an open source home automation and security system that consumers can implement themselves by downloading our code and purchasing cheap and easily available components. This will allow people to not only take advantage of all the technology that is currently available, but to improve their quality of life by making the worries and the tedious task they once had to do disappear.

In understanding this project there were 3 major tasks to accomplish. First, we need to figure out what hardware to use in order to implement all of the features we want. Second, we need

to figure out what programming languages we will write the programs for each of the features we want to implement. Third, we need to find a method to make all the component's controls accessible through the web and Amazon Echo.

Product Proposals

Team 6 Members:
Espinoza, Jesus; Gonzales, Jorge; Guzman, Omar; Ledesma MarcoAntonio

Purpose and Justification

To design and build a home automation and security system with facial and voice recognition, motion detection, and other wireless controls. This project will mainly focus on user authentication with face recognition as the primary authentication method, secondary method would consist of an access code, and the automation systems of the home will be controlled via a smartphone web application.

Statement of the problem or Need

To make a home secured and prevent unauthorized access. Most homes are currently equipped with motion sensors, window and door sensors. The alarm systems usually have a control unit where the user enters a code to disarm the alarm system within a certain amount of time. Moreover, currently homes have little if any automation meaning that things inside the house cannot be controlled wirelessly which is what we plan to address in this project.

Project Deliverables

- Facial recognition to gain access and disable home security (Primary)
- Access key code to gain access and disable home security (Secondary)
- Voice recognition to enable/disable TV and door lock
- Implementation of "Alexa" using Raspberry Pi: Weather information, control smart devices, set alarms, and much more
- Wireless control of light intensity
- Intelligent Doorbell – Sends an alert to cell phone and enables video streaming to see who is at the door.
- Status of doors/windows (open/closed)
- Home temperature and humidity
- Wireless garden moisture monitor
- Web application to monitor every component mentioned above.

Knowledge Required for Project

- Programming in Python, NodeJS, HTML, CSS
- Network Programming
- Software Design
- Circuit Design
- Application Design
- 3D Modeling and Printing

Project Assumptions and Constraints

Assumptions:

We would have to meet the following requirements to have a fully functioning project by the end of the semester

- Implementation of home security
    - Facial Recognition, Access Key Code
    - Status of doors/windows
    - Intelligent Doorbell
- Implementation of home automation
    - Voice Recognition to control entertainment system
    - Lighting control
    - Window/Door monitor
    - Central system to get live weather and traffic report
    - Home temperature
    - Moisture sensor for irrigation

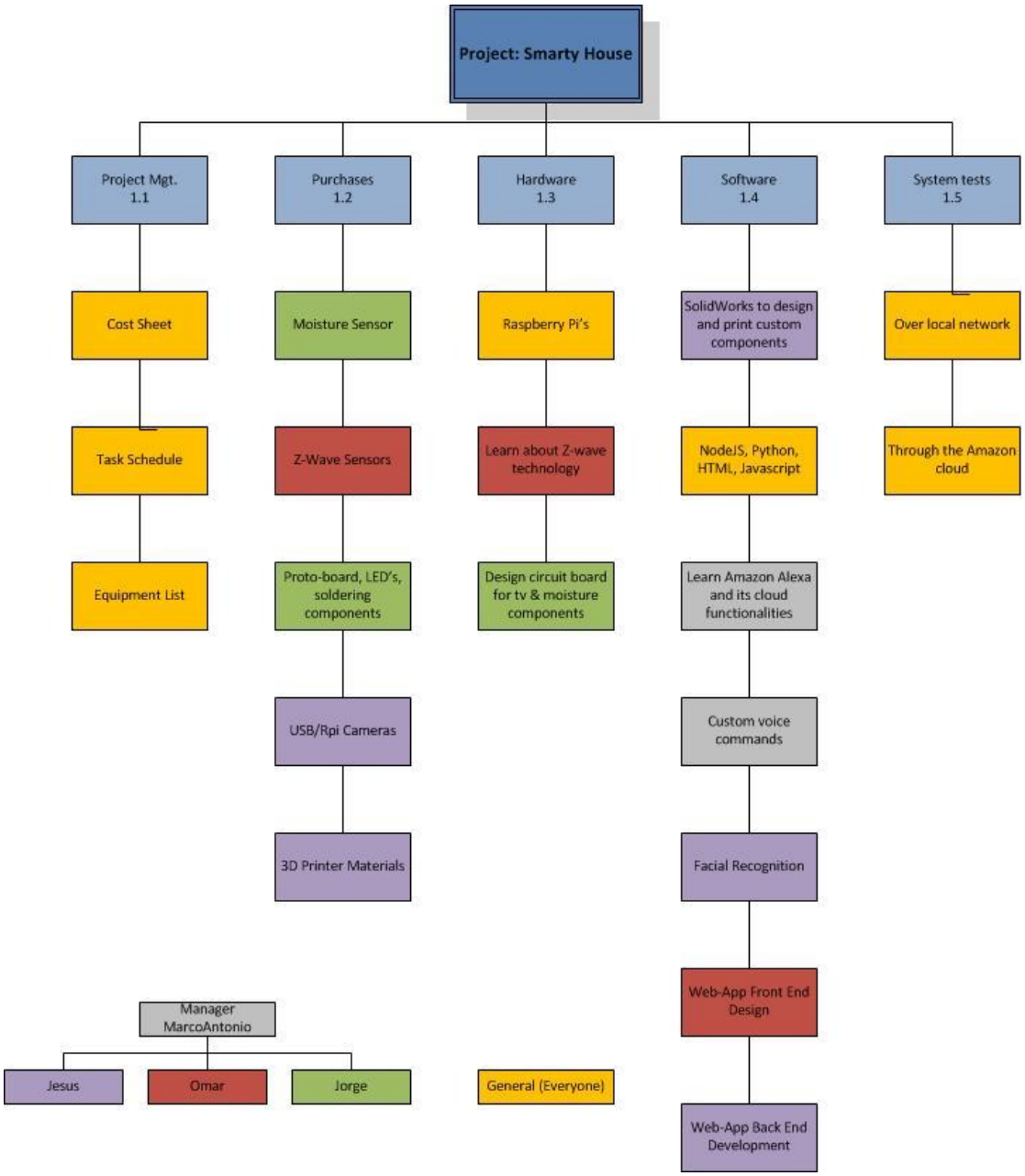
Constraints:

Development of the project would be in a tight schedule that would need the tasks to be done in parallel. Approximately 3 months to completion.

Project Risks

Since there are many features to implement us would all need to work on individual tasks and work together at the end to integrate all parts into a single product.

WBS and Schedule



Above we have our WBS for the fall. It is color coded to highlight which team member is assigned which task. Although each member has been assigned a particular task, the task can

be completed by anyone in the team, however, it is the responsibility of the person to whom a task is assigned to monitor its progress.

Funding Proposals

The goal of our project is to engineer an affordable home automation system that can use all kinds of components from different companies without having to stick to a certain brand. Many of our features are using a raspberry pi in one way or another. We didn't have to spend much money of raspberry Pi's because we all had some available from previous courses. We also had access to a 3D printer which was really helpful when needing custom made parts. We are funding the project ourselves not only to emphasize the affordability, but also to have full ownership of our home automation system.

Below is the total cost of the project broken down by component:

	A	B	C	D
1	Item	Quantity	Cost	Purchased By
2	Raspberry Pi	4	140	
3	Z-wave Dongle	1	45.95	Omar
4	Z-Wave Dimmer switch	1	33.85	Omar
5	Z-Wave Door sensor	1	26.75	Omar
6	Microphone	1	5	Tich
7	LCD Screen	1	70	Jesus
8	Web Camera	2	40	Jesus/Marco
9	Raspberri Pi Cam	1	30	Jorge
10	Amazon Echo Dot	1	40	Jesus
11	Servos	3	10	All
12	Arduino 101	1	30	Jesus
13	Battery and Solar Panel	1	18	Jesus
14	3D Printer Material	1	20	Jesus
15	Temperature Sensor	1	5	Jorge
16	IR Light	1	1	Jorge
17	Total Cost		515.55	

Many of our components were recycled from previous courses and projects so this saved us money but total cost as if we would have purchased is reflected above. We already had our Raspberry Pi's. We worked with 4 during the semester, so each person could be working on a feature, but the project could have worked with 2-3 Pi's. Ideally 3 raspberry Pi's would be best. Since we funded the project ourselves we have full ownership. This is good for us.

Team Member Tasks

Once we determined the work breakdown schedule, we needed to make each member of the group responsible for certain tasks. We split the tasks amongst ourselves by assigning a responsible person as well as a back-up person to a task. The "back-up" was set up in case something happened to the person responsible for the task in order to avoid delays in the development of the project.

We broke down the tasks in the following way:

Task	Assigned to	Backup
Cost Sheet	General Team	N/A
Keeping track of the cost of every component being purchased for the development of the project. Also, keeping track of how much each member spent and on what		
Weekly Report	MarcoAntonio	N/A
Updating the weekly report with the team minutes from each week's minutes. Also, responsible for submitting the report after each member has updated the report with their respective task progress.		
Equipment List	MarcoAntonio	Jesus
Responsible for tracking the amount of components being used for each feature in the automation system		
Purchases of components	General Team	N/A
Everyone in the team is responsible for purchasing the necessary components for their feature unless a request for funds is submitted to the team manager.		
Learn NodeJS, Python, HTML and JS	General Team	N/A
Each member of the team is responsible for learning the appropriate language for the feature there are responsible for. This includes, but is not limited to learning the		

NodeJs, Python and HTML which are used in most of the automation features.		
Temperature & Humidity Sensor	Jorge	Jesus
Responsible for the development of a temperature and moisture sensor to be implemented with the Raspberry Pi		
Alexa Skills	MarcoAntonio	Omar
Responsible for research and development of custom voice commands for Amazon Alexa		
Facial Recognition	Jesus	MarcoAntonio
Responsible for the design and implementation of the facial recognition security system		
Design Irrigation (Moisture) System	Jesus	MarcoAntonio
Responsible for the design and implementation of a wireless system to monitor the moisture levels of vegetation		
TV Remote (IR Sensor)	Jorge	Jesus
Develop a wireless TV remote that can be configured with any TV, and that can transmit command wirelessly over the web		
Z-Wave Sensors	Omar	Jesus
Responsible for research and development of light and door sensors using Z-Wave technology		
Graphic User Interface (Front End)	Omar	Jesus
Responsible for front end design of the user interface (web app)		
Graphic User Interface (Back End)	Jesus	Omar
Responsible for the back end development of the user interface (web app)		
Sensor Testing	Jorge	MarcoAntonio
Responsible for the testing and debugging of the sensors and any other hardware/software that may be showing errors		

User Manual

The home automation systems use 4 raspberry Pi's (Figure 1.1) as the hubs for the different components implemented in the project although less RPi's can be used if the user does not wish to have all the components in different places of his/her home.

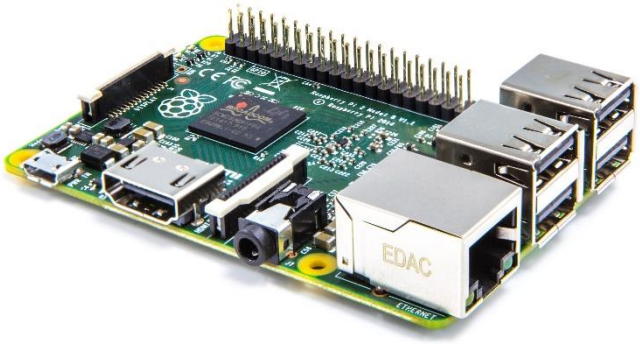


Figure 1.1 Raspberry Pi

Also, the features in the project make use of Z-Wave Sensors for control of the lights and doors/windows security system as well as an infrared LED to control the TV. Furthermore, we also used a Amazon Echo Dot in order to implement voice commands as an option for the user to control the features besides the web application.

Figure 1.2 Amazon Echo Dot & Z-Wave Sensor

Facial Recognition Registration

The first step in setting up the facial recognition software is to create a username for yourself. After creating a username, you need to register a 4 digit pin followed by taking a series of 30 different pictures in order for the program to store your face in its memory. You will need to place yourself relatively close to the camera and make different facial expression in order to allow the program to better recognize you in the future.

TV Remote Control

In order to implement this feature with the TV you own, first you must run the "lirc" program and register the infrared signals from your TV's remote control. Lirc will consequently produce a file with all the commands for your TV which then the python program will read in order to send the right commands to your TV. Also, it must be noted that the Raspberry Pi with the IR LED needs to be placed within 2 feet from the TV as that is as far as the range of the LED will go. Bigger ranges can be supported with stronger IR LED's.

Alexa Commands

In this project we have also implemented Alexan commands in order to control the lights, door, and TV. In order to do this, simply say the words, "Hello Alexa, ask smarty house to..." And use any of the following options:

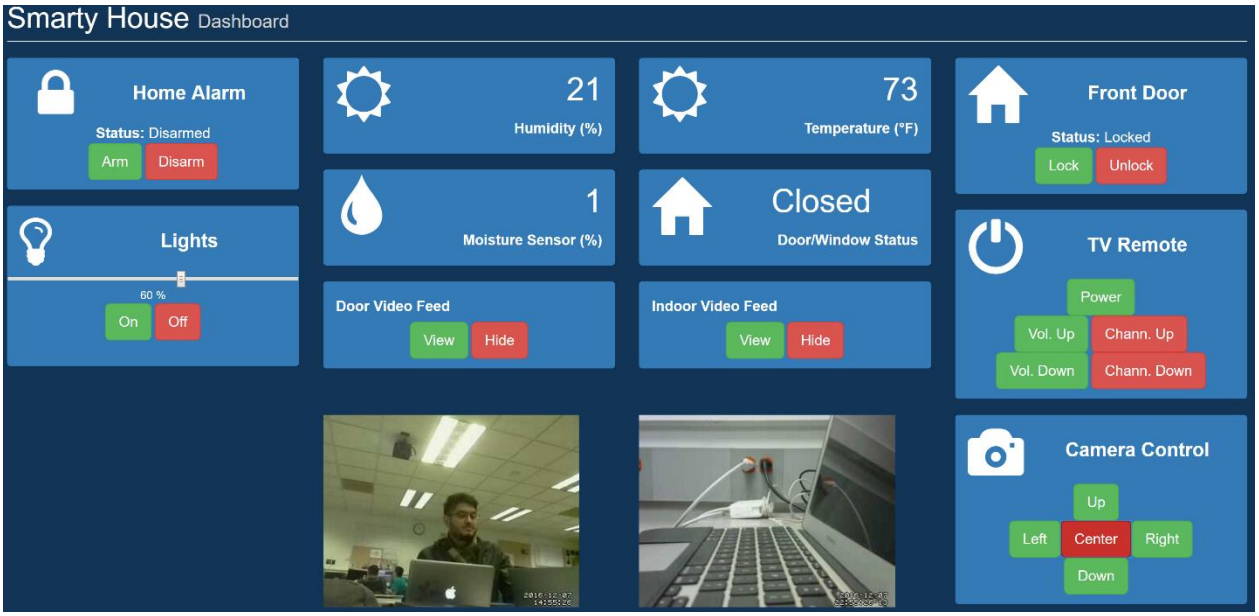
- Turn on/off the light
- Lock/unlock the door
- Turn on/off the TV
- Turn the channel up/down
- Turn the volume up/down

Security Camera

In order to control the security camera, simply open the web application and click on the corresponding buttons. The camera implement in this project can move in 4 directions, up, down, left and right.

Web Application

All features can be controlled and accessed thru our web application. This allows the user to see stats like home temperature and humidity as well as door/window status.



Design Documentation

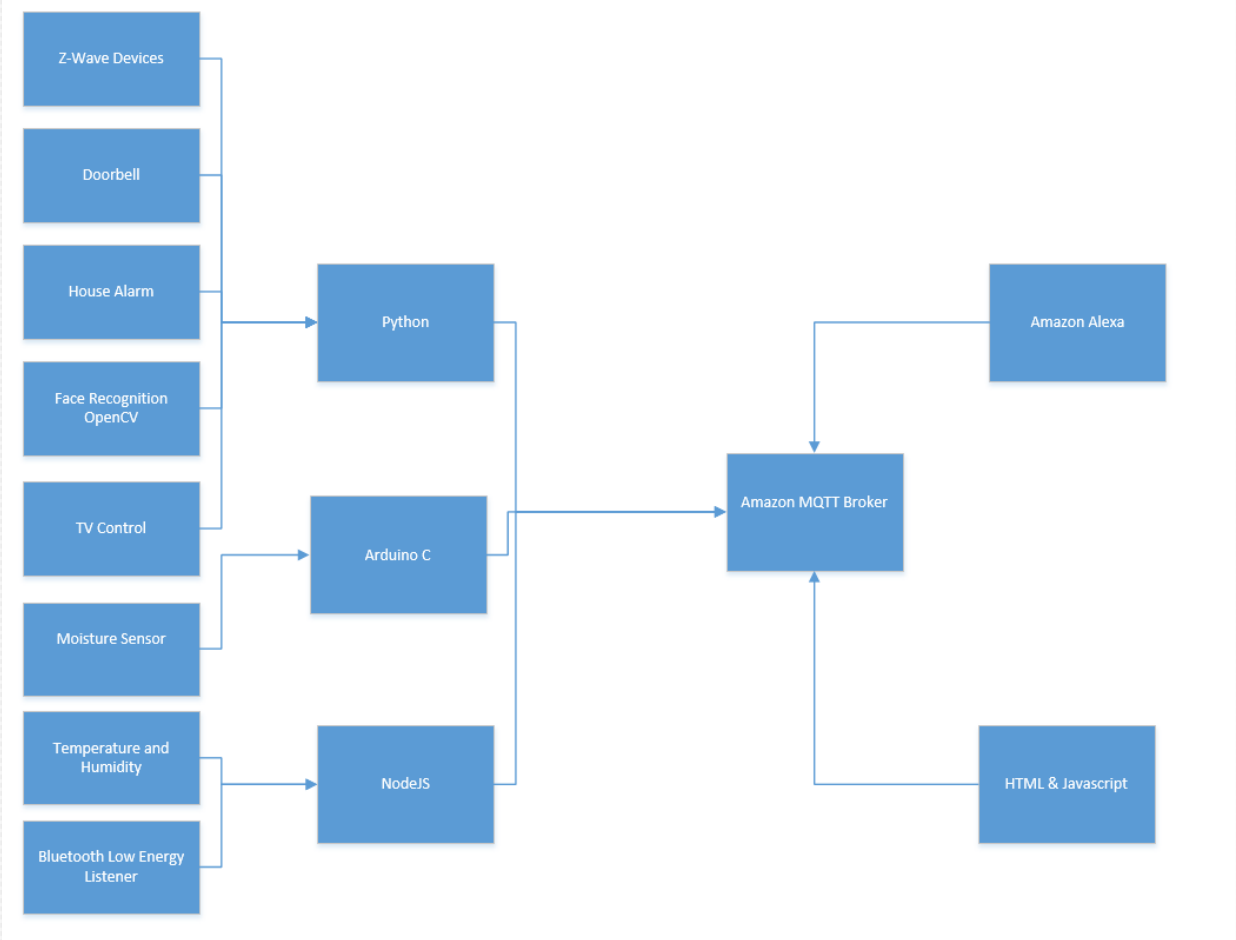
The purpose of this design is to provide people with a better way to control everyday devices wirelessly and take full advantage of the technology available today. Our design will use 4 Raspberry Pi's as the hubs for the automation devices, which will then send messages to the cloud to update the web application. The user will be able to put these devices anywhere around his home and be able to monitor them through the internet.

In the fall semester we were able to design and implement the idea in its entirety. Currently we have four Raspberry Pi's working as the hubs for the features we implemented, Z-Wave sensors for the doors and windows, IR LED to control the TV, and Amazon Echo in order to send voice commands through Alexa. We tested each of the features individually first and once we knew the features were working correctly, we hooked up the entire system and tested it one more time.

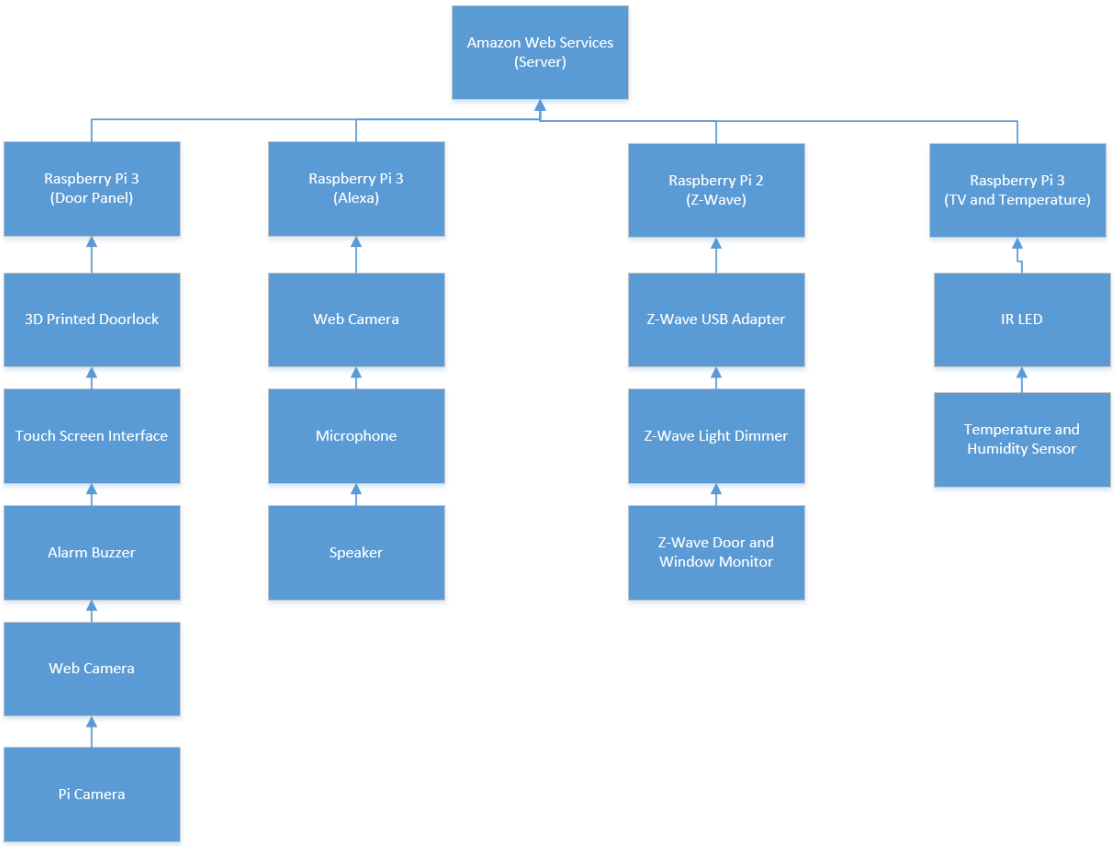
Some of the obstacles we had to face during the semester were the limitations in time we had to implement our project because since we switched projects, we only had one semester to finish it up. Also, another problem that came up is that the Raspberry Pi recently released a new OS and a lot of the documentation online was for a previous version causing us a lot of stress and conflicts with our programs. Furthermore, we ran into some issues with the communications of our Raspberry Pi's with the web application. In the beginning we were using the MQTT Mosquito broker which ran in the local network, but was having issues when connected outside of the network. We solved this by moving our entire design to the cloud by using Amazon Cloud Services.

In the end, working together as a team, and having patience constantly researching helped us overcome all of these obstacles and we were able to successfully implement all of the features we wanted at the beginning of the semester. The web application can successfully monitor and control the home automation devices we programmed, Alexa can also perform the same tasks the web application can through the use of voice commands, and we were successful at implementing our facial recognition security system.

Software Block Diagram



Hardware Block Diagram



Software

This project required a large amount of software implementation. For this project we used various programming languages such as HTML, JavaScript, and Python, we also used different software tools such as Amazon Web Services and Alexa Skills Kit. In the following section, we will be discussing and explaining different parts of the project and how the code was implemented.

The following code is an example of how the web application was developed.

```
<link href="include/css/bootstrap.min.css" rel="stylesheet"

<link href="include/css/sb-admin.css" rel="stylesheet">

<link href="include/css/plugins/morris.css" rel="stylesheet">

<link href="include/font-awesome/css/font-awesome.min.css" rel="stylesheet"
type="text/css">

<script src="include/js/jquery.js"></script>

<script src="include/js/bootstrap.min.js"></script>
```

In the portion of the code shown above, we simply liked to Bootstrap. This helped us make our web application look a lot better. The font, colors, and template appears much better than if you had just used HTML on its own.

```
<script type="text/javascript">

$("#vmoisture").text("1");

var socket = io.connect('192.168.1.217:1991');

socket.on('connect', function () {

socket.on('mqtt', function (msg) {

if(msg.topic === '/temperature') {
```

```
console.log("Temperature: " + msg.payload);
$("#vtemperature").text(msg.payload);
}
</script>
```

In this portion we are showing how our web application communicates with the other devices via MQTT protocol. In this example we are showing how the temperature data got transmitted from the Raspberry Pi to get displayed on the web application.

Now we will explain briefly how the facial recognition system was implemented. This feature was written in python. In the following code we can get a basic understanding of how the facial recognition system works.

```
while (True):
    (_, frame)= video_capture.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in faces:
      cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 255), 4)
      face = gray[y:y + h, x:x + w]
      face_resize = cv2.resize(face, (width, height))
      prediction = model.predict(face_resize)
      cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 255), 4)
      if prediction[1]<500:
        cv2.putText(frame,'%s - %.0f' % (names[prediction[0]],prediction[1]),(x-10, y-10),
cv2.FONT_HERSHEY_PLAIN,2,(0, 255, 255))
```

```
            print("Recognized: " + (names[prediction[0]]))
            result = str(names[prediction[0]])
            detected+=1
        else:
            cv2.putText(frame, 'Not Recognized', (x-10, y-10), cv2.FONT_HERSHEY_PLAIN,2,(0, 255, 255))
            print("unknown")
            unknown+=1
                .
                .
                .
```

Basically, we first have to register a user, this is done by taking 30 pictures of their face. These pictures get stored in a directory. Once the facial recognition button is pressed, the program goes in a while loop. In this loop we compare the current live feed to the pictures that we stored in the directory. If the live feed from the camera matches the pictures then the user is recognized otherwise a "not recognized" message will be displayed.

```
def customCallback(client, userdata, message):
        global positionx
        global positiony
        if(message.topic == "/camera"):
                if(message.payload == "left"):
                        if positionx >= 250:
                                print("limit")
                        else:
                                positionx = positionx + 20
                                wiringpi.pwmWrite(pinx, positionx)
```

```
elif(message.payload == "right"):
        if positionx <= 50:
                print("limit")
        else:
                positionx = positionx - 20
                wiringpi.pwmWrite(pinx, positionx)
                                .
                                .
                                .
```

The above code was used to control the webcam. Here we are show how we control the servos for the camera to move in the X-axis and in the Y-Axis. In the web application we have displayed 4 buttons for the user to use and control the movement of the camera. If the user clicks on "left" the program first checks that the position of the camera is not the most left position, if the camera can't go anymore left, a "limit" will be printed. On the other hand, if the camera is not at its limit, the servo will move 20 degrees to the left. It's the same logic for all the other directions, "right", "up", and "down".

```
network = ZWaveNetwork(options, autostart=False)
network.start()
manager = libopenzwave.PyManager()
manager.create()
for i in range(0,2):
   if network.state>=network.STATE_READY:
      print("Network is ready")
      break
   else:
```

```
        time.sleep(1.0)
def sendDimmer(value):
            network.nodes[2].set_dimmer(dimmer1, value)
def customCallback(client, userdata, msg):
            if msg.topic == "/dimmer":
                        value = int(str(msg.payload))
                        sendDimmer(value)
def callback(args):
    if args:
        if 'valueId' in args:
            v = args['valueId']
            if str(v['id']) == doorSensor:
if str(v['value']) == "True":
myAWSIoTMQTTClient.publish("/door", "Open", 1)
elif str(v['value']) == "False":
myAWSIoTMQTTClient.publish("/door", "Closed", 1)
```

The above code was implemented for the Z-Wave devices and the lock system. We used the same approach as for the other devices, meaning that these devices just like the rest, communicate to the cloud. Here we can see how the function"sendDimmer" is implemented to send the appropriate values to the cloud.

Aside from all these different devices and its code, we also worked with the Alexa Skills Kit. We used the Amazon Development services to create our own Alexa Skills. There was minor coding for this part of the project. Basically we had to create instances to which Alexa will respond accordingly.

Hardware

Raspberry Pi 3 Model B

The Raspberry Pi 3 Model B is the latest version of the Raspberry Pi family. It replaced the Raspberry Pi 2 Model B in February 2016. This powerful device comes with the following specs:

- A 1.2GHz 64-bit quad-core ARMv8 CPU
- 802.11n Wireless LAN
- Bluetooth 4.1
- Bluetooth Low Energy (BLE
- 1GB RAM
- 4 USB ports
- 40 GPIO pins
- Full HDMI port
- Ethernet port
- Combined 3.5mm audio jack and composite video
- Camera interface (CSI)
- Display interface (DSI)
- Micro SD card slot (now push-pull rather than push-push)
- VideoCore IV 3D graphics core

https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

Logitech C270

For the webcams, we used the Logitech C270 that comes HD video calling, Logitech Fluid Crystal Technology, photos up to 3.0 megapixels, build-in mic with noise reduction, and Hi-Speed USB 2.0.

http://www.logitech.com/en-us/product/hd-webcam-c270

Raspberry Pi Camera Module

The Raspberry Pi Camera Module is an add-on for any of the Raspberry Pis. It directly attaches to the Raspberry Pi by one of the sockets that can be found on the board. The CSI bus is very capable of high data rates and carries pixel data.

https://www.adafruit.com/product/1367

Hobby Servos

For the controlled webcam, we used two servos that are mapped using PWM ports from the Raspberry Pi 3. The servos worked perfectly to control the webcam in both X-axis and Y-axis. Each servo has a 0-180 degree rotation movement, this was just enough to monitor a bedroom for example.

Arduino 101

The moisture sensor was implemented using the Arduino 101. The Arduino 101 was perfect and easy to program for this feature of the project. The board comes with a 196 kB Flash Memory, 24 kB SRAM, and 32 MHz Clock Speed. It was powered via a Micro Solar 100mA/5v solar power panel.

https://www.arduino.cc/en/Main/ArduinoBoard101

Amazon Dot

The Amazon Dot is a 360 degree speaker that uses far-field voice recognition and uses Amazon Alexa voice services. This small devices was used to control the various different devices we had. There are many things the Amazon Dot can do, it can be paired to speakers via Bluetooth, and it can be paired to many apps such as Fitbit, Uber, Domino's Pizza and many more.

Z-Wave Devices

We purchased a Z-Wave light switch and a door sensor. Both of these devices communicated to a Z-Wave dongle connected to a Raspberry Pi. All of these signals get sent to the Amazon cloud and that is how all of the devices worked.

Software Test Plan

The Smarty House project consists of various different parts that required software implementation. All of the devices communicate via MQTT protocol. The main broker or main unit was implemented in the Amazon Web Services cloud. In the following section, a brief description of each of the devices and their software implementation will be discussed.

1. The web application was developed using HTML, Bootstrap, and JavaScript. We took advantage of the useful tools that Bootstrap has to offer to make our web application look elegant. JavaScript was used to make our web application dynamic.

2. For the television control implementation, python was the language used to interact between the Raspberry Pi and the television via infrared wavelengths.

3. Python was also used to implement the camera control. We used the GPIO pins of one of the Raspberry Pis to control 2 servos that moved 180 degrees in the X axis and 180 degrees in the Y axis.

4. For the webcams, we installed the Motion software on 2 of our Raspberry Pis. This open-source software is widely used by developers.

5. The moist sensor was implemented using an Arduino 101. We used the Arduino software IDE to write out code for the moist sensor.

6. The facial recognition system was implemented using OpenCV. OpenCV is an open-source library. This library came in handy because is it used for real-time image processing, and is used in applications like gesture mapping, motion tracking, and facial recognition.

7. Along with OpenCV, python code was also written for the facial the security system. On top of the facial recognition, we also implemented a passcode system to verify the user is who he say he is. A basic alarm system was also implemented using python in case an intruder.

8. We used Alexa Skills to implement the voice commands to control the devices. The Alexa Skills Kit can be found in the Amazon Developer website.

9. Aside from the Alexa Skills, we also had to implement AWS (Amazon Web Services) Lambda functions. AWS Lambda is a compute service that runs the code in response to events and automatically manages the underlying compute resources for you. We created our own back-end services so our devices can communicate via MQTT protocol to a central unit that was the Amazon cloud.

Hardware Test Plan

Our Smarty House project consists of many individual components that will in the end be integrated into one. So first we will make sure all parts work individually and then slowly integrates them together.

1. Our first step is to make sure all our microcontrollers and power supplies for those micro controllers are working properly. This includes our many raspberry pies and arduino 101 that we are using.

2. Test raspberry pies and arduino. Make sure the wireless communication is working properly. Test that each port necessary is working properly.

3. Test Z-Wave components: Z-Stick, Z-Dimmer Switch, Z Door Sensor.

4. Test the Z-stick on a regular computer to make sure it can connect and be paired. We used OpenHab to test if the Z-Stick was working properly before trying to implement of a raspberry pi.

5. Test Z-Dimmer Switch: Install switch in apartment to test functionality. Try Pairing with Z-Stick. Test that it works with app and physically.

6. Test Z Door Sensor. Pair the sensor similarly as Dimmer switch. Make sure we can get a reading of when the sensor is opened and closed.

7. After All Z-Wave components have been tested. Build a portable circuit for light bulb with dimmer Switch. Test that the circuit works just as when it was installed in apartment.

8. Test Temperature, Humidity, and IR sensors. These 3 sensors are to be implemented on one Raspberry pi. Test to make sure we can get a reading from all sensors.

9. Test moisture sensor block: This includes testing that solar panel on block powering the arduino. Make sure the solar panel can keep the battery charged enough to get the reading from the sensor.

10. Test Raspberry Pi Touch Screen. Connect and configure raspberry pi to run on the Touch screen that will be used for the door entrance. Make sure the pi can run just as it were to be running on a regular monitor.

11. Test All Cameras. Test to make sure the Facial recognition camera can be powered on and functional. Also check that we can get a camera feed from both the Front door camera and the monitoring camera. All cameras are running must be able to get

enough power from raspberry pies.

12. Design and test a 3D printed gear lock. (real locks are expensive). Test the lock to make sure it can function like a regular lock. Make sure the lock using a servo can get enough power to lock and unlock.

13. Design and 3D print mount for monitoring camera. This will be done using two servos to rotate camera in 4 directions. Up, Down, Left, and Right. Test to make sure servos can rotate the camera in these 4 directions.

14. Test Hobby Servos. Run test on hobby servos to make sure they are fully functional. Test for delay and smooth movement of the servos. No faulty Servos. Test to make sure some don't get too hot. They might be faulty.

15. Test Voice commands using Amazon Alexa. Make sure the mic is working and can pick up voice commands after the skills have been set.

16. Once we have done all components testing. Now integrate parts together and test by adding one component at a time. For this the webpage needs to be finished and cloud services must be ready as well.

17. Once every components is integrated together make sure we can control the features with both the webapp and voice commands. Also test that the webapp is functional on mobile devices.

Integration Plan

1. The first step in our integration plans is to make sure all the hardware and software components are functioning properly individually.

2. Once we have the Amazon cloud services running we will add one raspberry pi at a time with all the functionally assigned to that specific pi. All components should have been configured and working when testing parts individually. Integrating consists of merging the code to the cloud which allows communication between all components.

3. First we will add the Pi that will run openzwave and give us control of the Z-wave Dimmer switch and the Z-Wave Door sensor. Make configurations to turn on and off the light with using the web app. Also make sure we are getting a reading from the Door sensor. The Z-components should have been paired already with the Z-stick in previous steps when testing the hardware individually.

4. Next we integrate the raspberry pi that will use the touch screen for the door access and facial recognition. First we need to integrate two cameras, the touch screen, and the raspberry pi all into one compact component. A 3D printed case will be used to hold the Pi, Screen, and cameras all in one. Once we have all those components together we will merge software to cloud. Next we test to make sure all cameras, screen, and facial recognition is still working. We will test to make sure we can still see the camera feed with web app. Door lock functions will be tested at this point.

5. Our next task is to integrate the arduino 101 moisture sensor box. The arduino with the moisture sensor data can communicate to any of the 3 pies to then send the data to the cloud. Ideally the arduino will use the raspberry pi at the door as the bridge to communicate the data to the cloud. This makes sense since it will be the pi closest to the garden where a moisture sensor can be used. We will then test to make sure we can still get moisture data on our web app dashboard.

6. Next we will add the raspberry pi that has the IR sensor for TV controls and also has the house temperature sensor and humidity sensor. Once the merge to the cloud is complete we will check to see if we can still get temperature and humidity data. TV controls will also be tested. The web app has common controls such as power, volume, and channel operations. We will test that all those functions making sure they still work after the merge to the cloud.

7. The last Pi we will integrate to the cloud will be the Pi with Alexa features and monitoring camera. Alexa is our Amazon smart assistant that will give us the ability to add voice commands for things like turning on lights, locking door, or powering up the TV. The Pi with Alexa will also be running our other camera that monitors whatever you want such as a baby room. Once merged to the could we will test to see the voice commands working and that we can still get a video feed from the monitoring camera.

8. At this point everything is integrated all in one system. All components are connected to the cloud and are able to upload and download data. At this point we will test that all features work with both the web app and voice enable commands as well. If everything is up and running we can consider our project complete.

Accomplishments

This semester we had multiple challenges that we as a team faced and overcame. This project was definitely a journey for our books. We successfully implemented everything from our project proposal. The smarty house can be controlled from a web interface that can be accessed from a computer, tablet or a mobile device. Amazon Alexa was also implemented to work with our home automation and security devices.

The web interface includes home stats like temperature, humidity, door status, alarm status, and more. We are able to control the door lock, lights, and web cameras. From our web application we can also arm and disarm the alarm system. The back end of the web interface is a very great accomplishment, we have websockets running that is bridging the HTML and nodeJS software that hosts the website and sends/recieves data from the cloud. The NodeJS creates a server where the HTML is hosted connects the Amazon Web Services (AWS) where we have an MQTT Broker running.

The Amazon Web Services is very complex and it was definitely a great learning accomplishment. All of our devices are secured and the communication between the devices and the broker are encrypted. The devices need a public/private keys and certificate of authentication in order to communicate via the AWS. The MQTT Broker running on the AWS is a publish and subscribe system. Unlike sockets the communication is always going thru the broker and if the listening device is not open it will simply be unread.

The face recognition is another accomplishment that we are very proud of. We used openCV which is an image processing library for Linux. We registered users by taking 31 pictures, name and secret passcode. Then the live web camera feed is scanned for a face, once it detects a face

it will compare the face with the 31 pictures previously stored. If a match is found it will notify the percent of accuracy, we then used a high threshold to filter out false positives.

Having access to a 3D printer allowed us to create custom parts and components to piece our project together. Jesus was familiar with SolidWorks so most of the 3D printed parts were designed by him. Jesus designed, Door lock system, Pan/Tilt camera system, and camera holders.

To communicate with the light system we used a Z-Wave dongle that allowed us to scan and register standard z-wave devices. This allows our project for further expansion since there are multiple z-wave compatible devices in the market that can easily be implemented to our project. The z-wave devices that we implemented is a dimmer light switch and a door/window sensor. Once the device had an update the z-wave dongle would receive it and be sent up to the cloud for our website to display.

We are very proud of this project especially after receiving great feedback from the visitors at the showcase. The people we talked to were very impressed at the way we approached our communication between our devices and the way we utilized Amazon Web Services.

Summary

Our goal is to design and build a home automation system that can be more affordable due to the variety of components that can be added from many manufacturers. Many systems out there are expensive and make you use their proprietary components. Our system will allow you to decide what components you may want to add to your home automation system. Our Home Automation System consists of two parts. The first part is the home Security part which includes the face recognition to enter door and lock with camera to let you know who is at your door. We also have implemented a monitoring camera which can be used for security or monitoring a baby room or any other ideas the user might find a camera useful for. The second part is the home automation that integrates many smart home features. Those features include light controls, TV controls, and a bunch of sensors for different kinds of data. We have temperature sensors, moisture sensors for a garden, humidity, and door/window sensors.

We have successfully implemented all our smart home features. We have control of all features via the web app we created or voice commands using the Amazon Alexa. This was all successful by migrating most of our project to the Amazon cloud services. Most of our features are running on either a raspberry pi or Arduino. Our components wirelessly connect to the Amazon cloud where the can send or pick up signals to control features like the lights or locks. The web app is the only part hosted locally but does have connecting to the cloud to send and receive data from the could. This allows us to see sensor information on our web app and control smart features like light control. Our project currently stands at 100 percent of what we said we would implement. There is lots of room to grow and add more features. Overall it was a very successful project.

References

OpenCV Image Processing

http://opencv.org/

Amazon Web Services

https://aws.amazon.com/documentation/

Raspberry Pi

https://www.raspberrypi.org/documentation/

Open Z-Wave

https://github.com/openzwave/

BootStrap

http://getbootstrap.com/css/

MQTT Broker

http://mqtt.org/